Design and validation of a methodology to implement data-driven predictive maintenance in industrial environments

Oscar Serradilla Casado

Supervisors:

Urko Zurutuza Ortega

and

Ekhi Zugasti Uriguen

A thesis submitted to Mondragon Unibertsitatea
for the degree of Doctor of Philosophy

Department of Electronics and Computer Science
Mondragon Goi Eskola Politeknikoa
Mondragon Unibertsitatea
November 2021

To my parents Mila and Pedro, my brother Iker, Jessica and Esther. For your accompanying and unconditional support ♡

Abstract

New trends in manufacturing and industry lead to digitalise all processes, machines and communicate them forming Cyber Physical Systems (CPS), facilitating process monitoring and data acquisition. The analysis of that amount of data provides new insights in product quality, process optimisation and Predictive Maintenance (PdM). PdM analyses industrial assets to perform maintenance actions that extend their life and anticipate their failures to prevent them, optimising maintenance costs with respect to time-based and corrective maintenance strategies. PdM systems aim to monitor industrial assets to detect anomalies, diagnose their root cause, predict their degradation and propose mitigation actions.

Research on data-driven PdM systems has increased in the last six years due to their capability to model complex industrial systems by learning from the large amount of data collected from industrial assets. However, they are rarely transferred to industrial production scenarios due to they fail incorporating domain expert knowledge to the system. In addition, most data-driven works do not address industrial requirements such as interpretability, real time execution, novelty detection or uncertainty modelling. No methodology to guide the life-cycle of data-driven PdM models in industrial environments exist, which could facilitate the implementation of PdM systems in real use-cases to reduce maintenance costs and avoid production breakdowns.

The main contribution of this thesis is the design and validation of the methodology for data-driven techniques and expert knowledge combination for predictive maintenance MEthodology for DAta-Driven techniques and Expert Knowledge combination for Predictive Maintenance (MEDADEK-PdM). It defines the stages, steps and tasks to guide the design, development and implementation of data-driven PdM systems according to business and process characteristics. It defines the required working profiles to facilitate their collaboration, and includes a list of deliverables. The methodology is designed in a flexible and iterative way, combining standards, state-of-the-art methodologies and related works of the field.

The methodology has been validated empirically by its application in

three industrial use-cases, where industrial requirements are addressed. The first use-case consists of modelling correct working engine data to detect anomalies in run-to-failure aviation engine data, addressing novelty detection with data-driven PdM systems in a simulated environment. The second use-case consists of estimating and explaining the Remaining Useful Life (RUL) of experiments in a bushing testbed, by combining data-driven PdM systems with eXplainable Artificial Intelligence (XAI) techniques and domain knowledge. The third use-case implements an adaptable data-driven PdM system for semi-supervised anomaly detection and diagnosis in press machine process data. The system detects novel anomalies and performs their diagnosis combining XAI, clustering and projection techniques. The adaptability of the system to changing Environmental and Operational Conditions (EOC) is addressed with transfer learning.

The application of the proposed methodology guides the life-cycle of data-driven PdM systems, integrating Human-In-The-Loop (HITL) to include domain knowledge. As a result, the obtained PdM systems tackle the specific industrial requirements of the addressed use-cases, obtaining a trade-off between accuracy and explainability.

Resumen

Las nuevas tendencias en la fabricación y la industria llevan a digitalizar todos los procesos y máquinas y a comunicarlos formando sistemas ciberfísicos (CPS), facilitando la monitorización de los procesos y la adquisición de datos. El análisis de esa cantidad de datos proporciona nuevos conocimientos sobre la calidad de los productos, la optimización de los procesos y el mantenimiento predictivo (PdM). El PdM analiza los activos industriales para llevar a cabo acciones de mantenimiento que prolonguen su vida útil y se anticipen a sus fallos para prevenirlos, optimizando los costes de mantenimiento respecto a las estrategias de mantenimiento correctivo y mantenimiento predeterminado. Los sistemas PdM tienen como objetivo monitorizar los activos industriales para detectar anomalías, diagnosticar su causa raíz, predecir su degradación y proponer acciones de mitigación.

La investigación sobre los sistemas PdM basados en datos ha aumentado en los últimos seis años debido a su capacidad para modelar sistemas industriales complejos aprendiendo de la gran cantidad de datos recogidos de los activos industriales. Sin embargo, rara vez son transferidos a escenarios de producción industrial debido a que no logran incorporar el conocimiento de los expertos del dominio al sistema. Además, la mayoría de los trabajos basados en datos no abordan requisitos industriales como la interpretabilidad, la ejecución en tiempo real, la detección de nuevos patrones de funcionamiento o el modelado de la incertidumbre. No existe ninguna metodología que guíe el ciclo de vida de los modelos PdM basados en datos en entornos industriales, lo que facilitaría la implementación de los sistemas de PdM en casos de uso reales para reducir sus costes de mantenimiento y evitar paradas de producción.

La principal aportación de esta tesis es el diseño y la validación de la metodología para combinar de técnicas basadas en datos y conocimiento experto en la aplicación de mantenimiento predictivo (MEDADEK-PdM). La metodología define las etapas, los pasos y las tareas necesarios para guiar el diseño, el desarrollo y la implementación de sistemas de PdM basados en datos según las características del proceso y su negocio. Además, define los

perfiles de trabajo necesarios para facilitar su colaboración e incluye una lista de entregables resultantes de su implementación. El diseño de la metodología es flexible e iterativo, combinando estándares, metodologías del estado del arte y trabajos relacionados del campo de investigación.

La metodología ha sido validada empíricamente mediante su aplicación en tres casos de uso industriales. El primer caso de uso consiste en detectar anomalías en motores de aviación mediante el modelado de datos de funcionamiento correcto, abordando la detección de nuevas anomalías con sistemas de PdM basados en datos en un entorno de simulación. El segundo caso de uso consiste en estimar y explicar la vida útil restante (RUL) de los experimentos en un banco de pruebas de casquillos, combinando sistemas de PdM basados en datos con técnicas de inteligencia artificial explicable (XAI) y conocimiento del dominio. El tercer caso de uso implementa un sistema de PdM adaptable basado en datos para la detección y el diagnóstico de anomalías de forma semisupervisada en datos de proceso de máquinas de prensa. El sistema detecta nuevas anomalías y realiza su diagnóstico mediante la combinación de técnicas de XAI, clustering y proyección. La adaptabilidad del sistema a cambios en condiciones ambientales y operacionales (EOC) se aborda mediante el aprendizaje por transferencia.

La aplicación de la metodología propuesta guía el ciclo de vida de los sistemas PdM basados en datos, integrando al humano en el bucle (HITL) para incluir el conocimiento del dominio. Como resultado, los sistemas PdM obtenidos abordan los requisitos industriales específicos de cada caso de uso, obteniendo un equilibrio entre precisión y explicabilidad.

Laburpena

Fabrikazioaren eta industriaren joera berriek prozesu eta makina guztiak digitalizatzera eta komunikatzera eramaten dute sistema ziberfisikoak (CPS) eratuz, prozesuen monitorizazioa eta datuen eskurapena erraztuz. Datu kopuru horren analisiak ezagutza berriak ekartzen ditu produktuen kalitateari, prozesuen optimizazioari eta mantentze prediktiboari (PdM) buruz. PdMk aktibo industrialak aztertzen ditu hauen balio-bizitza luzatzen duten mantentze-ekintzak gauzatzeko eta hauen akatsei aurreratuz hauek prebenitzeko, mantentze-kostuak optimizatzeko mantenketa zuzengarria eta aurrez zehaztutako mantentze estrategiekin alderatuz. PdM sistemen helburua aktibo industrialak monitorizatzea da, anomaliak detektatzeko, horien kausa diagnostikatzeko, degradazioa aurreikusteko eta arintze-ekintzak proposatzeko.

Datuetan oinarritutako PdM sistemei buruzko ikerketak gora egin du azken sei urteotan industria-sistema konplexuak modelatzeko duten gaitasunagatik, industria-aktiboetatik jasotako datu kopuru handitik ikasiz. Hala ere, gutxitan transferitzen dira industria-produkzioko inguruneetara, ez baitute adituen ezagutza sisteman txertatzen. Gainera, datuetan oinarritutako lan gehienek ez dituzte baldintza industrialak jorratzen, hala nola interpretagarritasuna, denbora errealean gauzatzea, funtzionamendu-patroi berriak detektatzea edo ziurgabetasuna modelatzea. Ez dago industria-inguruneetan datuetan oinarritutako PdM ereduen bizi-zikloa gidatuko duen metodologiarik, eta horrek PdM sistemak ezartzea erraztuko luke erabilerakasu errealetan, mantentze-kostuak murrizteko eta produkzio-geldialdiak saihesteko.

Tesi honen ekarpen nagusia datuetan oinarritutako teknikak eta adituen ezagutza integratzen duten mantentze prediktiboaren aplikazioak gidatzeko metodologiaren (MEDADEK-PdM) diseinua eta balidazioa da. Metodologian, prozesuko eta negozioko ezaugarrien arabera datuetan oinarritutako PdM sistemen diseinua, garapena eta inplementazioa gidatzeko beharrezkoak diren etapak, urratsak eta zereginak zehazten dira. Gainera, lankidetza errazteko beharrezkoak diren lan-profilak zehazten ditu, eta metodologia

inplementatzearen ondorioz sortutako entregagaien zerrenda jasotzen du. Metodologiaren diseinua malgua eta iteratiboa da, estandarrak, puntako metodologiak eta ikerketa arloarekin lotutako lanak konbinatuz.

Metodologia enpirikoki baliozkotu da, hiru erabilera-kasu industrialetan aplikatuz. Lehenengo erabilera-kasua abiazio-motorretan anomaliak detektatzean datza, funtzionamendu egokiko datuak modelatuz, simulazio-ingurune batean datuetan oinarritutako PdM sistemekin anomalia berriak detektatuz. Bigarren erabilera-kasua zorro bankuan egindako esperimentuen gainerako bizitza erabilgarria (RUL) kalkulatzean eta azaltzean datza, datuetan oinarritutako PdM sistemak adimen artifizial azalgarriko (XAI) teknikekin eta domeinuaren ezagutzarekin konbinatuz. Hirugarren erabilera-kasuak datuetan oinarritutako PdM sistema moldagarri bat inplementatzen du, prentsa-makinen prozesu-datuetan anomaliak modu erdi-superbisatuan detektatzeko eta diagnostikatzeko. Sistemak anomalia berriak detektatzen ditu eta horien diagnostikoa egiten du XAI, multzokatze eta proiekzio teknikak konbinatuz. Sistemak ingurumen-baldintzen eta baldintza operatiboen (EOC) aldaketetara egokitzeko gaitasuna lortzeko transferentzia bidezko ikaskuntza erabiltzen du.

Proposatutako metodologiaren aplikazioak datuetan oinarritutako PdM sistemen bizi-zikloa gidatzen du, gizakia buklean (HITL) integratuz domeinuaren ezagutza sartzeko. Horren ondorioz, lortutako PdM sistemek erabilera-kasu bakoitzaren baldintza industrial espezifikoak jorratzen dituzte, zehaztasunaren eta azalgarritasunaren arteko oreka lortuz.

Eskertza

This thesis is the result of years of research and study, where many people participated to make it possible and facilitate the journey. I would like to express my gratitude to all of them, including the ones that are not explicitly mentioned.

Mila esker Mondragon Unibertsitateari. Txikitatik unibertsitatera joateko ilusioa neukan, eta Informatikako gradua egiteko MU aukeratu nuen. Hemen, ikasitakoa praktikara eramatea ere garrantzitsua zela ikasi nuen. Unibertsitateko ikasketa metodologia hainbeste gustatu zitzaidan berriro aukeratu nuela tesia egiteko. Tesia egiteko behar izan dudan prestakuntza eta errekurtsoak eskaini dizkidate.

Aipamen berezia merezi dute nire tesi zuzendariak Urko Zurutuza eta Ekhi Zugasti. Zuen orientazioa ezinbestekoa izan da momentu zailenetan, arlo tekniko eta kudeaketak errazteko. Neurri handian zuei zor dizuet ikerlari bihurtu izana.

Bidean zehar egon zareten tesiko kideei ere eskerrak eman nahi dizkizuet, Unai, Julen, Javi, Iñigo, Markel, Xabi Etxezarreta eta Xabi Gandiaga. Zuekin egindako atxedenak, deskonektatzeko eta lanera itzultzeko indarrak eman dizkidalako.

I am also grateful for the opportunity of collaboration provided by the research center Koniker and its director David Chico. Its been a pleasure to work with Jon Rodriguez and Julian Ramired de Okariz, who have provided me with the resources to ellaborate the research. I do not forget Markel, Xabi, Aitor eta Mikele, zuekin oso eroso sentitu naiz eta behar izan dudanean, laguntza eskaini didazue.

This thesis has been possible by the funding received from the project MEANER, which has been funded by the Provincial Council of Gipuzkoa, and the European project Q4LITY of Horizon 2020.

A mis padres Mila y Pedro, por educarme, apoyarme y enseñarme que trabajando todo termina saliendo. A mi hermano Iker por su apoyo y ayudarme a desconectar. A Jessica por acompañarme y por esos momentos de relax disfrutando de las pequeñas cosas. A Esther por escucharme y apoyarme. Y al resto de familia y amigos por estar ahí y amenizar el camino.

Contents

List of Figures

List of Tables

List of abbreviations

2D-CNN-AE Two Dimensional CNN-based AutoEncoder

AD Anomaly Detection

AE AutoEncoder

ANN Artificial Neural Network

AI Artificial Intelligence

ARIMA AutoRegressive Integrated Moving Average

BN Bayesian Network

CBM Condition Based Maintenance

CM Condition Monitoring

CNN Convolutional Neural Nework

CNN-AE CNN-based Autoencoder

CNN-VAE CNN-based Variational AutoEncoder

CNN-AE-ELM CNN-based AutoEncoder combined with ELM in the latent space

CPS Cyber Physical Systems

DAE Denoising AutoEncoder

DBN Deep Belief Network

DL Deep Learning

DT Decision Tree

ELI5 Explain Like I'm 5

ELM Extreme Learning Machine

EM Expectation Maximisation

EMA Exponential Moving Average

EOC Environmental and Operational Conditions

EWMA Exponentially Weighted Moving Average

FE Feature Engineering

FFNN Feed Forward Neural Network

FMEA Failure Modes and Effects Analysis

FMECA Failure Mode Effects and Criticality Analysis

FN False Negative

FP False Positive

GAN Generative Adversarial Network

GMM Gaussian Mixture Models

GRU Gated Rectified Unit

HBOS Histogram-Based Outlier Detection

HI Health Index

HITL Human-In-The-Loop

HMI Human-Machine Interface

HMM Hidden Markov Models

I4.0 Industry 4.0

ICT Information & Communication Technology

IF Isolation Forest

IIoT Industrial Internet of Things

ISO International Standarization Organization

k-NN K Nearest Neighbors

KDE Kernel Density Estimation

LIME Local Interpretable Model-agnostic Explanations

LOCI LOcal Correlation Integral

LOF Local Outlier Factor

LRP Layer-wise Relevance Propagation

LSTM Long-Short Term Memory

MAE Mean Absolute Error

MCE Motor Current Evaluation

MEDADEK-PdM MEthodology for DAta-Driven techniques and Expert Knowledge combination for Predictive Maintenance

ML Machine Learning

MLP Multi Layer Perceptron

MM Maintenance Management

MOM Manufacturing Operation Management

MSCA Motor Signal Current Analysis

mRMR minimum Redundancy Maximum Relevance

N/A Not Available

NN Neural Network

NDT NonDestructive Testing

OCC One Class Classification

OC-SVM One-Class Support Vector Machine

OEE Overall Equipment Effectiveness

OPTICS Ordering Points To Identify the Clustering Structure

OSA-CBM Open System Architecture for Bondition Based Maintenance

PAM Partitioning Around Medoids

PCA Principal Component Analysis

PdM Predictive Maintenance

PHM Prognosis and Health Management

PLC Programmable Logic Controler

P-F Potential Failure

RBDA Rank Based Detection Algorithm

RBM Restricted Boltzmann Machine

RCA Root Cause Analysis

reLU Rectified Linear Unit

RF Random Forest

RFE Recursive Feature Elimination

RMSE Root Mean Square Error

RNN Recurrent Neural Network

RUL Remaining Useful Life

RVR Relevance Vector Regression

SAE Sparse AutoEncoder

SHAP SHapley Additive exPlanations

SotA State-of-the-Art

SOM Self Organising Map

SpRAY Sprectral Relevance AnalYsis

SVM Support Vector Machines

SVR Support Vector Regressor

t-SNE t-distributed Stochastic Neighbor Embedding

TP True Negative

TN True Positive

VAE Variational AutoEnconder

XAI eXplainable Artificial Intelligence

XGBoost eXtra Gradient Boosting

Chapter 1

Introduction

This chapter introduces maintenance background, basic predictive maintenance concepts, and author's motivation for research to provide solutions to observed problems. Accordingly, the research hypothesis, objectives and thesis statements are defined, complemented with the main thesis contributions.

1.1 Maintenance in industrial environments

The evolution of industry since the first machine was created has been classified into different revolutions. The first brought mechanisation with water power and steam power, the second brought mass manufacturing by product lines and electric energy, and the third brought the production automation by digital revolution based on Information & Communication Technology (ICT) and electronics.

Nowadays, the fourth revolution denominated as Industry 4.0 (I4.0) is underway, being supported on Cyber Physical Systems (CPS) and Industrial Internet of Things (IIoT). Its objective is the digitalisation of industry to improve industrial processes and address their requirements using software, sensors and intelligent control units [20].

The heterogeneous data collected from I4.0 platforms creates opportunities for process optimisation, product quality and Predictive Maintenance (PdM), among others. One of the main opportunities identified in I4.0 is maintenance optimisation by applying data-driven techniques to massive amount of collected process data, making predictive and proactive maintenance strategies more accessible than ever before.

The norm EN 13306 [21] defines **maintenance** as *the combination of all technical, administrative and managerial actions during the life cycle of an item intended to retain it in, or restore it to, a state in which it can perform the required function*, **failure** as

the loss of the ability of an item to perform a required function and **fault** as *a state of an item where it is unable to perform the required function, excluding time for maintenance actions.*

Furthermore, it defines 3 main maintenance strategies:

- **Improvement maintenance**: to improve reliability, maintainability and safety while keeping the original function.
- **Preventive maintenance**: before failure occurs; it can be classified in two sub-types according to the performing way. One sub-type is **predetermined maintenance**, where the components are replaced periodically without observing their degradation. The other sub-type is **Condition Based Maintenance (CBM) supported on PdM**, in which diagnosis and prognosis are performed to anticipate maintenance requirements based on assets' condition.
- **Corrective Maintenance**: replaces the broken parts of a machine when it stops working or when its users detect defects on it. This maintenance strategy is expensive and leads to environmental and human security problems.

There is another commonly mentioned maintenance strategy called proactive maintenance, which *strives to identify and address the problems that can lead to breakdowns in the first place, addressing the root causes* [22].

Industrial companies consider maintenance a key process to be tracked and for that, Maintenance Management (MM) and Manufacturing Operation Management (MOM) approaches are used.

1.2 Introduction to predictive maintenance

PdM aims to keep assets working correctly and only apply maintenance actions once they start degrading, anticipating their maintenance requirements. This maintenance strategy extends components' working life with respect to predetermined maintenance, and prevents damages by intervening before failures occur: before corrective maintenance, as shown in Figure 1.1.

Even though the concept of PdM is a step further from the CBM, which is the application

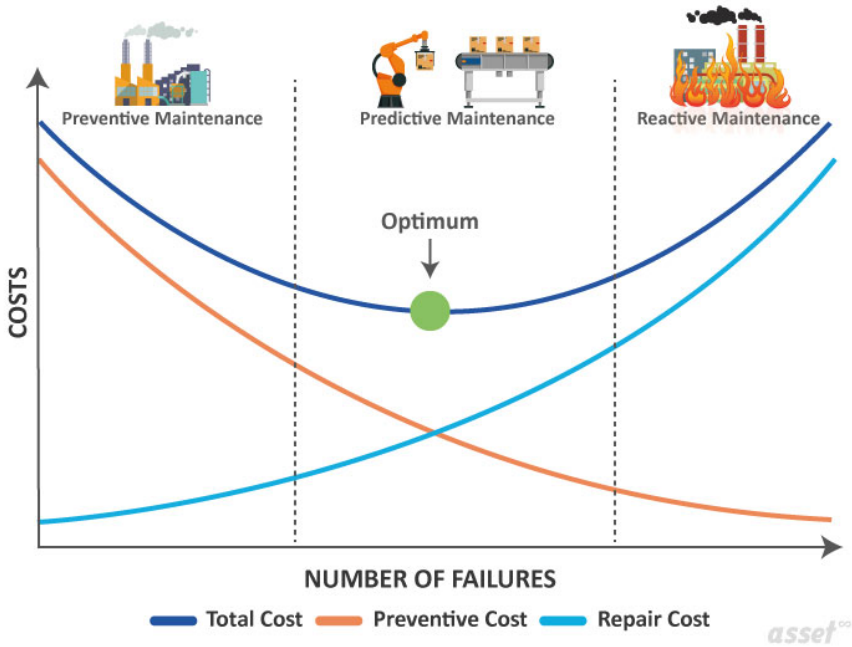


Figure 1.1: PdM optimisation cost by Asset Infinity [1].

of Prognosis and Health Management (PHM) in the maintenance field that arose in 1940's [23], nowadays its automation is even more feasible thanks to the standardisation of technologies.

PdM is achieved by a system that monitors machine/component conditions and detects early signs of failure to apply maintenance in a proactive way, supported on diagnosis and prognosis techniques. This health monitoring system can provide data-driven maintenance information and recommendations to domain technicians, who may anticipate maintenance needs.

These systems have applications on many fields such as manufacturing, automotive, energy, water, service and other industry domains [24, 25]. They anticipate and attend failures to ensure smooth operation which enhances total productive maintenance, improves Overall Equipment Effectiveness (OEE), safety, and protects the environment. As stated by Vorne in [26], OEE is a metric that measures the total productive time in range between 0% and 100%; it is described in Equation 1.1. According to Vorne [26], *it identifies all losses of time in availability, performance and quality which helps benchmarking progress and improving the productivity of manufacturing equipment.*

$$OEE = Availability \times Performance \times Quality \tag{1.1}$$

By using PdM, companies can achieve a OEE over 90% [22] and obtain a 10 times return on investment [27]. It is the most advanced level in maintenance, and often the most efficient by allowing strategic decision making, as introduced in Figure 1.2

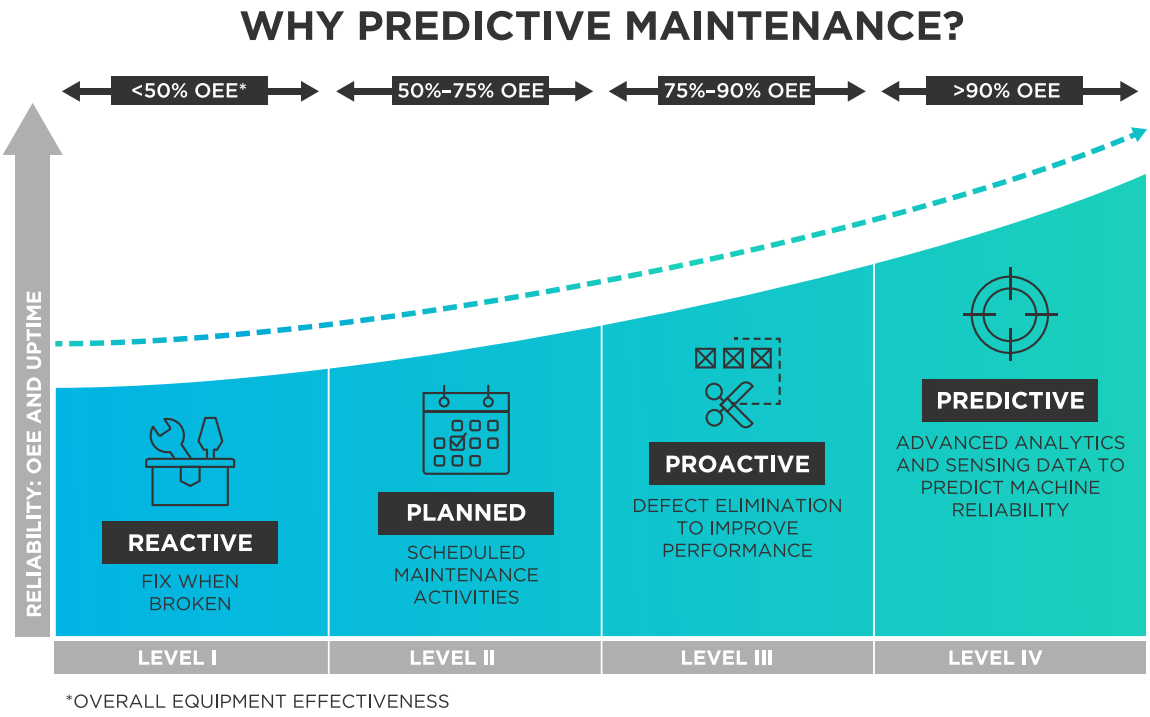


Figure 1.2: OEE of 4 maintenance strategies: corrective, predetermined, proactive and predictive respectively. Image by TIBCO [2].

1.3 Motivation

Effective maintenance can reduce industrial companies' costs up to 50% on machines, systems and people [28]. Nowadays most industrial companies are using the following maintenance approaches: either preventive/time-based maintenance, applying periodical interventions to avoid failures; or corrective maintenance, waiting until failures occur to apply interventions.

These maintenance strategies have a big economical optimisation potential. On the one hand, components' working life can be extended by taking advantage of their untapped

working time before failure, which reduces downtime and replacement costs. On the other hand, replacing components before failures occur can prevent expensive breakdowns whose reparation cost is much higher than the components, given production time loss and maintenance task cost.

For these reasons, recent trends are advancing towards data-driven predictive and proactive maintenance strategies, which have shown to be the most cost-optimal ones [22, 29]. This is best time for industrial companies to apply data-driven techniques to improve their processes, since many of them are monitoring their machines using sensors and uploading it to online platforms. However, there is a gap between the models presented in state-of-the-art publications, which often use controlled and simulated datasets to test the algorithms that do not contain industrial companies requirements. For that reason, several aspects required to deploy models to industrial plants are not addressed, such as modelling correct working data, diagnosis of novel anomalies or real time execution, among others.

Industrial companies require a systematic form or methodology that guides them in the development of data-driven PdM models supported on domain knowledge from scratch, given their lack of expertise and limited resources and time to take these systems to production. Additionally, 80% of Machine Learning (ML) projects finish in pre-production stages before reaching production [30], thus failing to deliver their potential value. A systematic methodology for data-driven PdM life-cycle management may help them prioritise, define use-case requirements and guide them throughout the process: thus reducing uncertainty, and optimising costs and time.

1.4 Hypotheses and research objectives

This thesis wants to demonstrate that **a modular methodology for data-driven predictive maintenance that integrates domain knowledge can guide the process of the life-cycle development of predictive maintenance in industrial environments**. This main hypothesis is completed with two additional hypotheses:

- H1: this methodology can adapt to different use-cases to meet their requirements, addressing their limitations and adapting to their diverse resources thanks to its flexibility.

- H2: given the guidance provided by Human-In-The-Loop (HITL) during the PdM life-cycle, the methodology enables to obtain explainable results.

Inasmuch as industrial companies are nowadays starting to develop data acquisition and data infrastructure, a modular methodology that guides them in the implementation of predictive maintenance life-cycle can facilitate companies addressing different use-cases to meet their requirements. Methodology's adaptability refers to its capability to accommodate to industrial characteristics such as data variability or the lack of failure data given that industrial companies work hard to avoid machine breakage. Moreover, the methodology should facilitate the integration of domain knowledge and the guidance of business perspective in the process, in order to achieve accurate as well as explainable models that meet industrial use-case requirements.

To find answer to these hypotheses, the thesis' **main objective** is defined to: **design and validate a modular methodology to systematise the application of data-driven predictive maintenance in industrial assets, guided by domain knowledge and adaptable to industrial requirements**. Two specific objectives are defined to fulfill the main objective:

- O1: design the methodology that enables the systematisation of data-driven predictive maintenance in industrial environments. It will be supported on existing data-driven PdM methodologies, standards, and architectures, and based on maturity level of digitalisation processes.
- O2: validate the methodology by its implementation in three industrial use-cases: aircraft engine run-to-failure simulation, fatigue experiments of a bushing testbed and press machine of a production line with no known failure. The methodology will systematically guide the data-driven PdM life-cycle on each use-case, being adaptable to fulfill their requirements and industrial needs.
  - O2.1: Demonstrate that the integration of domain knowledge in PdM life-cycle permits to obtain accurate and interpretable data-driven models.

After completing the objectives and validating the hypotheses, the developed and validated methodology will be ready for application in other industrial use-cases.

1.5 Contributions

The main contributions of this thesis are the following:

- Provides a state of the art review of data-driven predictive maintenance techniques, covering statistical, machine learning and deep learning architectures designed to address industrial requirements. It will help data-scientists in the selection of the most adequate algorithms that meet their use-case requirements according to the available resources and data characteristics.

- Presents a methodology for data-driven predictive maintenance application guided by expert knowledge in industrial environments. This methodology differs from state of the art works in covering the whole life-cycle required to design the use-case, analyse the resources, prepare the data, train and validate the model and deploy it into production. Thus, the methodology can guide industrial companies in the implementation of these technologies, helping them to overcome challenges. Moreover, the methodology proposes a roadmap to increase information levels on industrial data, which can be used to refine data collection systems and define data classification systems that will enable more advanced PdM implementation.

- Data-driven systems for three different use-cases based on simulation, test and production data have been implemented. Each use-case had its requirements and resources, so they have been analysed, implemented and compared State-of-the-Art (SotA) algorithms, adapting them to address industrial companies' requirements, which are sometimes overlooked in state-of-the art-works.

- The validation use-cases may be used as showcase of how to implement predictive maintenance for companies that are starting with it. These will help them spot possible applications and benefits of PdM, learn how to adapt the methodology to their requirements, identify the required resources and profiles, and gain knowledge on how data-driven models can be selected, adapted, compared and validated. The stated tasks may facilitate project planning.

- Presents how explainable artificial intelligence, clustering and projection techniques can help domain technicians in the diagnosis of novel anomalies detected by data-driven models. This may encourage collaboration of different profiles and increase trust of stakeholders in the technology, promoting the adoption of predictive

maintenance in industrial companies.

- Demonstrates that a PdM data-driven model can be reused over time with data of similar environmental and Environmental and Operational Conditions (EOC) by adaptation using transfer learning. However, model execution with different EOC data may require a new model.

- Disseminates the results of the research performed during this thesis:

  – O. Serradilla, E. Zugasti, C. Cernuda, A. Aranburu, J. R. de Okariz and U. Zurutuza, "Interpreting Remaining Useful Life estimations combining Explainable Artificial Intelligence and domain knowledge in industrial machinery," in *2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2020, pp. 1-8, doi: 10.1109/FUZZ48607.2020.9177537.

  – O. Serradilla, E. Zugasti, J. Ramirez de Okariz, J. Rodriguez and U. Zurutuza, "Adaptable and explainable predictive maintenance: Semi-supervised deep learning for anomaly detection and diagnosis in press machine data". *Applied Sciences*, 2021, 11(16), 7376, doi: 10.3390/app11167376

  – O. Serradilla, E. Zugasti, J. Rodriguez, and U. Zurutuza, "Deep learning models for predictive maintenance: a survey, comparison, challenges and prospects,". *Applied Intelligence*. Accepted for publication.

  – O. Serradilla, E. Zugasti, J. Rodriguez, and U. Zurutuza, "Methodology for data-driven predictive maintenance models design, development and implementation on manufacturing guided by domain knowledge". *International Journal of Computer Integrated Manufacturing*. Major revision.

1.6 Research methodology

Before deepening into the research undertaken in this thesis, the research methodology that guided its elaboration is briefly presented. Given the research is focused on designing and validating a methodology to manage data-driven PdM life-cycle, the author initially reviewed state-of-the-art data driven PdM works, and analysed methodologies related to PdM and data-driven life-cycle management. In addition, several experiments were performed to practice acquired knowledge.

Afterwards, the hypothesis that this thesis aims to demonstrate was specified, including secondary hypotheses. Moreover, the objectives to test the defined hypothesis were defined, addressing research questions.

Then, the data-driven PdM methodology was designed, combining state-of-the-art theoretical knowledge and author's experience. It is the main contribution of the thesis: a methodology for data-driven PdM life-cycle management including domain knowledge, to guide industrial companies in the design and implementation of their PdM use-cases according to their needs. It includes use-case definition, design and analysis of required resources, and PdM system development, validation, deployment and monitoring. This methodology and thesis hypothesis have been empirically validated in three diverse use-cases with different requirements. Moreover, theoretical foundations and domain expertise have been used in many PdM steps of these use-cases for tasks like data preparation or model design and validation.

The first use-case for methodology validation consists of detecting faulty engine conditions on simulated run-to-failure turbofan data, where PdM models learn from correct working conditions. The second use-case models remaining useful life of fatigue experiments performed in a bushing test bed, targeting accuracy and also explainability. Finally, the third use-case models press machine data of a stamping production line by learning from correct working conditions. Its objective is to perform anomaly detection and provide tools for diagnosis supported on a deep learning model. Fault diagnosis is addressed combining eXplainable Artificial Intelligence (XAI), clustering and projection techniques with domain knowledge. The adaptability of the anomaly detection model to different time and operational conditions has also been evaluated.

1.7 Organisation of the work

This work is organised in six chapters. The first and introductory chapter contextualises and introduces predictive maintenance, and explains the main motivations of the thesis. It also states the hypotheses tested during the thesis and its main objectives, in addition to limiting its scope, explaining the main contributions and defining research methodology.

The second chapter discusses the techniques and theoretical background required to

understand the main data-driven techniques used in experimentation. Moreover, it introduces and explains each predictive maintenance stage.

The third chapter reviews state-of-the-art predictive maintenance works based on expert-knowledge and data-driven techniques, including reference benchmark datasets and their results. Furthermore, it reviews standards, norms and state-of-the-art methodologies related to data-driven predictive maintenance.

The fourth chapter defines the methodology proposed in this thesis, which guides the life-cycle of data-driven predictive maintenance systems including their design, development and implementation combined with domain knowledge. It contains the main stages, steps and tasks for their elaboration, complemented by required profiles and resulting deliverables.

The fifth chapter presents the methodology validation in three industrial use-cases: simulated aircraft engine, bushing testbed and production press machine. Each use-case has different objectives and addresses different PdM stages, so each one implements the methodology according to its requirements.

The sixth and last chapter gathers the conclusions of the work and presents future research lines.

Chapter 2

Theoretical background

This chapter describes the theoretical background of data-analysis and predictive maintenance used for the elaboration of this thesis. Section 2.1 describes the background of the implemented statistical, machine learning and deep learning techniques for data-driven PdM, and scoring metrics to evaluate their performance. Furthermore, it describes transfer learning, its applications and main techniques. In addition, it overviews the XAI background, presenting the main techniques that have been used for model interpretation. Moreover, Section 2.2 sets the background of PdM, defines concepts related to it, and introduces the PdM roadmap by describing its stages.

2.1 Data analysis

This section presents the background of the statistical, machine learning and deep learning models related to the thesis classified by the ML task they address. Moreover, it reviews the topics transfer learning and explainable artificial intelligence.

2.1.1 Statistical and traditional machine learning models

This section explains how the statistical and traditional ML models used in the use-cases work, and the metrics used to evaluate them.

2.1.1.1 Supervised

Classification

Principal Components Analysis

The Principal Component Analysis (PCA) [31] is a commonly used technique for multivariate linear dimensionality reduction that takes advantage of the correlations among the input variables.

PCA works in the following way: it extracts the *eigenvectors* and *eigenvalues* of the input data matrix to project it into a new space where the new variables, named principal components, are orthogonal among them and hence they are linearly uncorrelated. The first principal component holds the largest variance of the original data, the second holds the second largest variance while being perpendicular to the first, and so on and so forth.

The application of PCA for dimensionality reduction is done by selecting the k first principal components to project the original data from d dimensions to k dimensions in the new space, where $k \leq d$. Another way to select the k number of components is by choosing an a-priory percentage of variance to be represented in the new space, and then, k is derived by selecting the least number of first principal components that represent the selected variance.

PCA has also applications on semi-supervised anomaly detection, by applying the dimensionality reduction with following the methodology presented in the previous paragraph, but then the projected data is reconstructed to the original space by only using the chosen k first principal components. Therefore, there will be a difference between the original and its reconstructed data if its explained variance is less than 100%. The Anomaly Detection (AD) is performed by calculating the distance between the reconstructed and the original data in a positive value that is denominated as *reconstruction error*, which is usually calculated using a distance function like *mahalanobis* or *euclidean* distances. The higher the reconstruction error is, the further or more different the reconstructed data is from the original data. Finally, a threshold is set in the reconstruction error to categorise as anomalous the instances with reconstruction errors higher than this threshold. This threshold can be calculated from the data if it is labelled, or be inferred in semi-supervised ways based on outlier detection or clustering techniques.

The logic behind this AD method is that the PCA model is trained to model the linear relations of the input variables by mapping them to the principal components' space named latent space. When the model is fed with data whose variables do not share the same correlations, the model will not be able to reconstruct the input data properly and

therefore the reconstruction error will increase and surpass the established threshold, categorising it as anomalous.

One-class Support Vector Machines

The One-Class Support Vector Machine (OC-SVM) [32] is a semi-supervised model with applications on outlier detection and anomaly detection. It is trained with data instances of one class, adjusting an hyperplane that surrounds them aiming at isolating the data distribution of the observed class. Then, this model can be used to classify novel instances into normal or anomalous/outliers classes when they are inside or outside the hyperplane respectively. In addition, the anomalous magnitude of the instances outside the hyperplane increases with their distance regarding the hyperplane.

The OC-SVM can use different kernels that enable modelling linear and also non-linear relations. Figure 2.1 shows a two dimensional space where the OC-SVM isolates the normal class with radial hyperplanes.

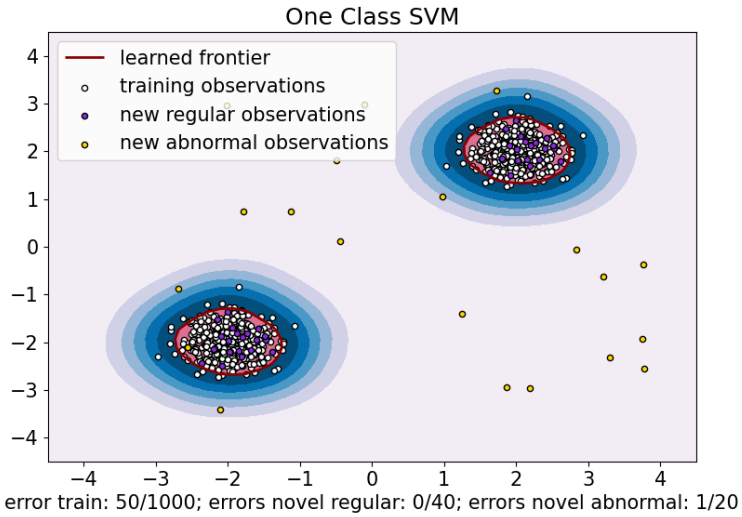


Figure 2.1: Schematic of a OC-SVM that models the normal class and establishes the hyperplane frontier to differentiate normal and anomalous data observations [3].

Regression

Random Forest Regression

The random forest regression [33] is a supervised model used for regression problems. It is a *bagging* ensemble method that underlies on the combination of less precise uncorrelated models to improve the precision and generalisation in an assembled model.

The random forest aggregates many random *regression trees* trained with different features and data subsets, forming a *forest*. The regressions are done by averaging the regressions of the trees.

eXtra Gradient Boosting regression

eXtra Gradient Boosting (XGBoost) [34] is an implementation of gradient boosting framework presented by Friedman [35] and Friedman et al. [36]. The XGBoost regression is a supervised model used for regression problems. It is a *boosting* ensemble method that aggregates weak regressors to fix the errors generated by existing ones using the gradient boosting method.

The difference between bagging and boosting algorithms is that the first aggregates random weak classifiers whereas the latter aggregates weak classifiers that fix the errors of existing classifiers. These differences are visualised in Figure 2.2.

2.1.1.2 Unsupervised

Agglomerative Clustering

The agglomerative clustering is an unsupervised technique to group the data into the selected k number of clusters. It uses a distance function to calculate the distance among observations in the feature space. The algorithm first assigns a cluster to each observation and then it recursively merges the clusters that are near given their observations, reducing the number of clusters while augmenting their size. This procedure is continued until there only remain the selected k number of clusters. One advantage of this algorithm is its visualisation of the distance among observations and clusters in a hierarchical scheme named dendrogram. In addition, it can select the number of clusters

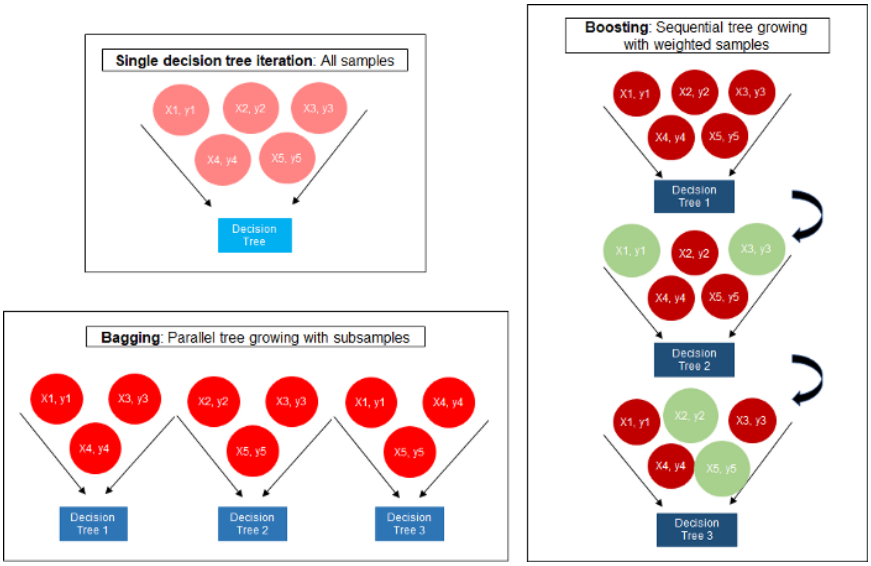


Figure 2.2: Difference between bagging and boosting, by Nikulski [4].

automatically by specifying the distance limit to stop merging the clusters.

Gaussian Mixture Models

The gaussian mixture models [37] model the data in a probabilistic way assuming that data observations belong to a finite number of gaussian distributions. The model uses the expectation maximisation algorithm [38] to estimate the parameters of the gaussian distributions from the data and cluster observations accordingly based on probabilities. Figure 2.3 shows the estimated probability distribution function of a gaussian mixture model with two gaussian distributions.

2.1.2 Neural networks and deep learning techniques

This section introduces deep learning background and presents the common architectures applied to the field of PdM, classified by the ML task they address. Nowadays, deep learning models outperform statistical and traditional ML models in many fields including PdM, when enough historical data exists. The deep learning term refers to Artificial Neural Network (ANN) that *go beyond shallow 1- and 2-hidden layer networks* [39], and ANNs are a machine learning technique inspired on how brains work.

A deep learning network is formed by layers of neurons that compute linear regressions

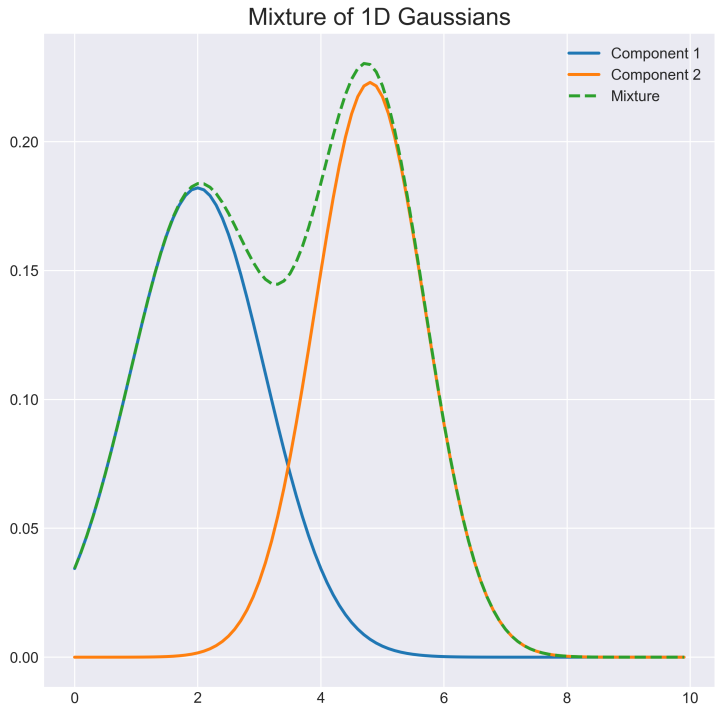


Figure 2.3: Plot of the probability distribution estimated from data using gaussian mixture models, where two gaussian distributions are combined. Image by Turner [5].

of inputs with weights plus a bias, and they may contain non-linear activation functions such as sigmoid, Rectified Linear Unit (reLU) or tan-h to produce non-linear outputs. The networks' parameters are commonly initialised randomly and afterwards they are adjusted to map input data to the selected output data given a training dataset. This learning process takes place by running the gradient descend algorithm combined with the backpropagation algorithm. These calculate the weight adjustments of each neuron required to reduce the error produced by the network, which is calculated by a cost function. The article by Hornik [40] justifies that networks of at least two hidden layers with enough training data are capable of modelling any function or behaviour, creating the universal approximator.

The book by Goodfellow et al. [41] provides exhaustive background on DL and it is considered a reference book by many researchers in the field. Concretely, the book in-

troduces machine learning and deep learning mathematical background. Afterwards, it focuses on DL optimisation, regularisation, presents different architecture types, introduces their mathematical definition and presents common applications. A simpler yet powerful overview of the field is done in the survey of DL applied to medicine by Litjens et al. [42], which is complemented with a visual scheme collecting the main architectures. Pouyanfar et al. [43] present another survey specifically focused on DL architectures, applications, frameworks, SotA and historical works, trends and challenges. Additionally, a reference book of practical DL applications is presented by Geron [44], which is based on Scikit-Learn [45], Keras [46], and TensorFlow [47] tools.

2.1.2.1 Supervised

Classification

Feed-forward network

Feed Forward Neural Network (FFNN) or Multi Layer Perceptron (MLP) [48] is the first, most common and simplest architecture. It is formed by stacked neurons creating layers, where all the neurons of a layer are connected to all the neurons of the next layer by feeding their output to others' input. However, there are no connections to neurons of previous layers or among neurons of the same layer. The first layer is named input layer, the last layer is denominated as output layer, and each intermediate layer is named hidden layer. The neural network is fed with observations pairing input features and target features, which are used to learn their relation by minimising the error produced by the network when mapping input data to the output. The majority of DL structures are based on the feed-forward scheme but each has its own characteristics.

Convolutional neural network

Convolutional Neural Nework (CNN) [49] is a type of feedforward network that processes neurons' neighborhoods by applying convolutional filters. It is inspired by the animal visual cortex and it has applications on image and signal recognition, recommendation systems and natural language processing among others. The convolutional layer is linear and it is usually followed by an activation function to produce non-linear outputs. After that, a max or average pooling layers may be used to reduce the dimension while

keeping relevant information. Finally, most architectures have a flatten step to obtain representative features of input data that can be used with other ML or DL networks to perform typical ML tasks. The weights of convolutions are shared, which makes them them easier to train.

Extreme Learning Machine

Extreme Learning Machine (ELM) [50] is a feed-forward neural network used for classification, clustering, regression and feature extraction [51]. It can be linear or use different activation functions such as sigmoid to model non-linear relations. It has a single or multiple hidden layers whose parameters are randomly assigned and they are not tuned. The output layer's weights are assigned by calculating the pseudoinverse on the projection of the training data into the network up to the last layer.

The work by Huang et al. [52] has proved that ELMs have the capability of universal approximator. Another advantage of ELMs is their high speed training and their little requirement of training resources. Figure 2.4 contains the scheme of a simple ELM.

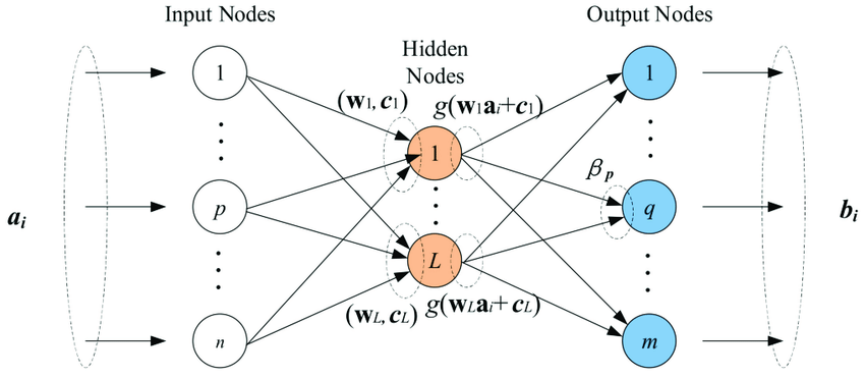


Figure 2.4: Schematic of the extreme learning machine of one hidden layer. Image by Zhang et al. [6].

Regression

Recurrent neural network

Recurrent Neural Network (RNN) [53] models temporal data by saving the state derived from previous inputs of the network. The back-propagation through time algorithm [54] is an adaptation of the traditional backpropagation for temporal data, which is used to propagate network's error to previous time instances. However, this propagation can result into vanishing or exploding gradient problem [55], making this networks forget long-term relations. To solve this problem, specific RNN architectures were created based on forget gates, like the Long-Short Term Memory (LSTM) [56] and Gated Rectified Unit (GRU) [57].

2.1.2.2 Unsupervised

Autoencoder

AutoEncoder (AE) [58] is based on the singular value decomposition concept [59] to extract the non-linear features that best represent the input data in a smaller space. It consists of two parts: an encoder that maps input data to the encoded, latent space, and the decoder, which projects latent space data to the reconstructed space that has the same dimension as input data. The network is trained to minimise the reconstruction error, which is the loss between the input and the output. Autoencoders can be classified according to their latent space dimensionality in undercomplete and overcomplete, which respectively correspond to a latent space smaller, and bigger or equal to the input dimension. These simple architectures are extended and adapted to address different tasks and problems. It can be trained in unsupervised way to detect patterns in its latent space, or it can be used for semi-supervised anomaly detection.

Vanilla autoencoders are the simplest autoencoders, which belong to the undercomplete type. There are different types of autoencoders obtained by applying regularisation and modifying the explained types. One adaptation is the Denoising AutoEncoder (DAE) [60], used for corrupt data reconstruction. It is a version of the overcomplete AE where learning is controlled to avoid the *identity function*. It is fed with data pairs of noisy input and its denoised output, and it is trained to reduce the loss between them. Another modification is the Sparse AutoEncoder (SAE) [61], an AE restricted in the learning phase that uses a sparse penalty constraint, which is based on the concept of KL-Divergence. This algorithm aims to make each neuron sparse, discovering the structure information from the data easier than vanilla AE, which may be useful for practical

applications [62].

Variational AutoEnconder (VAE) [63] is a generative and therefore non-deterministic modification of the vanilla AE where the latent space is continuous. Usually, its latent space distribution is gaussian, from where the decoder reconstructs the original data based on random sampling and interpolation. It has applications on estimating the data distribution, learning a representation of data samples and generating synthetic data samples, among others.

Generative adversarial network

Generative Adversarial Network (GAN) [64] is designed to work in unsupervised way. GAN is another type of generative neural network that consists of two parts: the generator and the discriminator. The generator is trained to generate an output that belongs to a specific data distribution using as input a representation vector. The discriminator is trained to classify its input data into the learned data distribution or not belonging to this distribution. The generator's output is connected to the discriminator's input and they are trained together, adversarially. The generator's objective is to bias the discriminator by generating outputs from a random input and trying to make the discriminator classify it as it belongs to the specific trained distribution. The role of the discriminator is to distinguish between the synthetic/generated data from the non-synthetic/real class data from trained distribution. They are trained together so that each part learns from the other, competing to bias the other part, similarly to game theory. GANs can be extended to other ML tasks such as supervised learning or reinforcement learning.

Self organising map

Self Organising Map (SOM) [65] is a neural network-based unsupervised way to organise the internal representations of the data. It is based on competitive learning, in contrast to typical neural networks that use backpropagation and gradient-descend for training. The SOM creates a new space called map that is typically 2 dimensional, where input data is projected and represented in cells. It is based on neighborhood functions to preserve the topological properties of the input space into the new space. It has applications on clustering and visualisation, among others.

2.1.3 Evaluation/scoring metrics

Measuring the performance of the ML models in any task such as classification, regression, clustering or anomaly detection is required. This will show the models' performance on the selected task, enabling comparisons among them and facilitating model optimisation by parameter setting and architecture selection. The models are first trained with a subset of the available data called train data, and then they are evaluated with scoring metrics on a different data subset denominated as test data. An additional data subset denominated as validation data can be used to track the training using the score metric, which can be used to stop the training in an optimal moment, preventing underfitting and overfitting.

There exist different metrics to evaluate ML models, but the most adequate ones should be selected according to the addressed problem. In binary anomaly classification, the confusion matrix is a widely used tool that enables results' evaluation and model comparison by counting the number of instances that are correctly and wrongly classified by the model. Figure 2.5 shows a confusion matrix containing the number of instances correctly classified by the model as normal and anomaly labelled *True Positive (TN)* and *True Negative (TP)*, whereas the number of misclassified normal and anomaly instances are denominated as *False Positive (FP)* and *False Negative (FN)* respectively [44].

Figure 2.5 also introduces three common evaluation metrics that facilitate model evaluation, which are defined by Exsilio solutions [66] as: *accuracy is the ratio of correctly predicted observations to the total observations, it is an intuitive metric despite only being useful for balanced classes*; *precision is the ratio of correctly predicted positive observations to the total predicted positive observations*; and *recall is the ratio of correctly predicted positive observations to the all observations in actual class.*

In addition, the *F-score* metric defined in Equation 2.1 is the harmonic mean that combines precision and recall, which contains a β parameter to weight their importance. When β equals 1, both metrics are evenly weighted; when β is higher than 1, more importance is given to the recall and as a result, to false negative errors; conversely, when beta takes values lower than 1, the precision is more valued than recall and therefore, the importance of False Positive errors is higher.

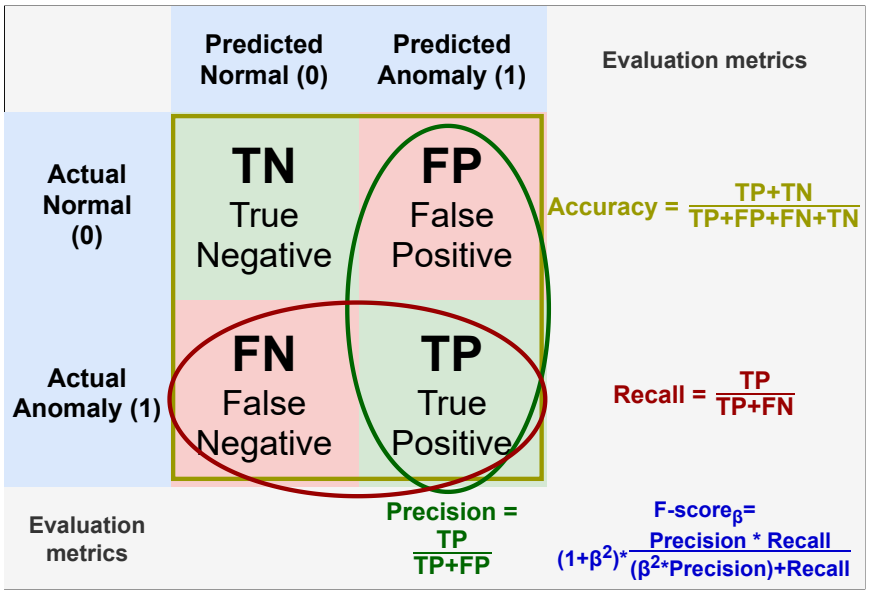


Figure 2.5: Confusion matrix and definition of the main evaluation metrics for binary classification, applied to anomaly detection.

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall} \tag{2.1}$$

Regarding the evaluation of regression models, the Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) are two widely used metrics. MAE is the average of all the absolute errors between the real and predicted values, presented in Equation 2.2; and RMSE is the square root of the average of all the squared errors between real and predicted values, presented in Equation 2.3. In these equations, i indicates the number of observation, y_i is the real target value in the observation i and $\hat{y}_i$ is the predicted target value in the observation i.

$$MAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n} \tag{2.2}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n}} \tag{2.3}$$

The MAE metric is more intuitive to understand than RMSE because it is the absolute error performed by the model on average in each observation. In contrast, RMSE is

more sensible to outliers than MAE, making it interesting for some use-cases [67].

2.1.4 Transfer learning

Transfer learning aims at transferring the knowledge acquired in the source domain to improve the performance on a related target domain, according to the survey by Zhuang et al. [7]. The use of transfer learning may reduce the required number of target domain data samples, which simplifies the data collection and model training processes. Transfer learning can also be beneficial for scenarios where data labelling is difficult, expensive or even impossible; for instance, when domain expertise is required or when only data of one class is collected.

The published research works about transfer learning have applications on developing new algorithms, improving existing transfer learning algorithms, and developing algorithms in new application domains, as indicated by Weiss et al. [68]. In addition, its authors classify transfer learning techniques into two groups: homogeneous, when the source and target domains are represented in the same feature space; and heterogeneous, when these domains are represented into different feature spaces.

Moreover, Zhuang et al. [7] classify the transfer learning techniques for models according to their strategy and objectives. The main categories of transfer learning techniques defined by strategies are model control, parameter control, model ensemble and deep learning techniques. Additionally, the categories of transfer learning techniques according to their objective are prediction making, domain adaptation and pseudo label generation. A scheme of these categories and strategies is presented in Figure 2.6.

2.1.5 eXplainable Artificial Intelligence

Complex ML and DL models can achieve high accuracy, but there is the need of understanding the decisions taken by these models so that humans can understand and trust them. These explanations may also be used for decision justification, control, improvement and discovery of model behaviour.

The objective of explainable artificial intelligence is to make Artificial Intelligence (AI) more transparent while maintaining high performance levels. Gunning et al. present on DARPA [69] the questions that XAI aims at answering from ML models: *why did you*

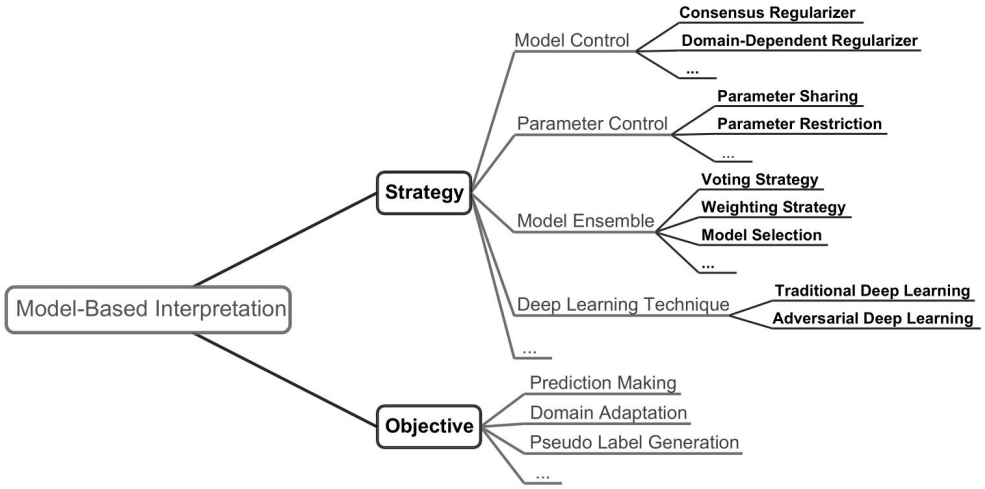


Figure 2.6: Tree-based classification of transfer learning methods according to their strategies and objectives. Image by Zhuang et al. [7].

do that? why not something else? when do you succeed? when do you fail? when can I trust you? how do I correct an error?.

The publication of XAI for anomaly detection made by Amarasinghe et al. [70] argues that explainability can be classified into two categories: model functionality, which focuses on explaining the reasons behind the model's learned concepts; and transparency, which can be analysed though three parameters: decomposability, simulatability and algorithm transparency. Decomposability is an intuitive explanations analysis, in simulatability a human reproduces model's calculations to predict the output given the input data, and algorithmic transparency focused on the inner workings of the learning algorithm.

Moreover, a detailed description of the XAI techniques is provided by Adadi et al. [71], who summarise their trends and explain their applications on explaining black-box models. XAI can have a big impact on the fields of transportation, healthcare, legal, finance and military among others. In addition, Adadi et al. [71] propose a classification of XAI techniques based on three characteristics:

- **Complexity**: the more complex the model is, the more difficult it is to interpret.
- **Scoop**: where *global interpretability* techniques analyse the model's overall logic and reasoning, and *local interpretability* techniques analyse the model's decision

based on individual data observations.

- **Level of dependency** where 2 types of interpretability techniques are distinguished: *model-specific* techniques take advantage of the particularities of the model to hande it, for example being white-box, and *model-agnostic* techniques are generally applicable to different models because they treat models as black-box.

There exist different XAI techniques available, which the book by Samek et al. [72] classifies into four groups regarding their underlying technique. Approximation using local surrogate functions samples neighbours of an observation to perform local explainability, like Local Interpretable Model-agnostic Explanations (LIME) [73] does. Local perturbations analyse how perturbations on an observation change model's output, like the sensitivity analysis feature importance technique proposed for Random Forests [33] works, and it was was generalised afterwards with ELI5 [9]. Propagation-based approaches integrate in the model structure using local redistribution rules based on methods as Layer-wise Relevance Propagation (LRP) [74]. Finally, meta-explanations use techniques like Sprectral Relevance AnalYsis (SpRAY) [75] and SHapley Additive exPlanations (SHAP) [76] that aggregate local explanations to obtain global explanations.

2.1.5.1 LIME

LIME [73] is a library that enables to explain individual predictions of black-box models. Concretely, it perturbs the data points near to the target observation, uses the model to predict the target values of these data points, and then it creates a sparse linear regression model using the data points and target values to approximate the black-box model's behaviour on the target observation. Finally, the linear model is analysed, interpreting this white-box approximation of the original model's behaviour on the target observation. Figure 2.7 presents a local linear approximation of the black-box model, and Figure 2.8 shows an example of the interpretability information provided by the linear model on the target observation.

2.1.5.2 ELI5

Explain Like I'm 5 (ELI5) [9] is a python library that helps to debug machine learning models and explain their predictions. ELI5 provides visualisation techniques for white-

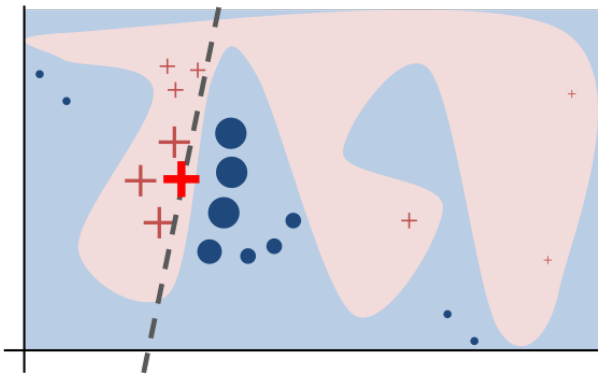


Figure 2.7: Representation of LIME perturbing the observations next to the target observation and approximating a linear model to enable the interpretation of a black-box classifier. Image by Tulio [8].

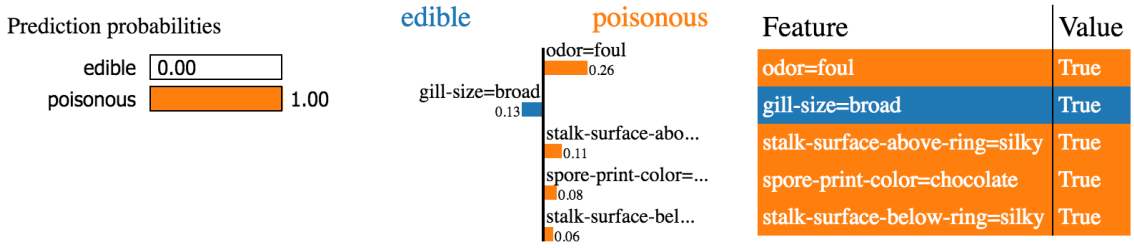


Figure 2.8: The information provided by LIME using linear models to interpret a local prediction of a model. It includes the features' contribution to the prediction. Image by Tulio [8]

box models by implementing local and global interpretation. It also enables global interpretation of black-box models by implementing permutation importance. Permutation importance shuffles the values of each feature of the dataset individually and compares the model's original accuracy with the accuracy decrease of each feature shuffle. Figure 2.9 shows the contribution of each feature to an individual prediction based on ELI5.

2.1.5.3 SHAP

SHAP [76] is a local interpretation library that enables explaining the output of any machine learning model [10]. It is supported on a game theoretic approach, using shapley values [77] and their related extensions to explain the contribution of features to the model's predictions. The library creates an approximated prediction model for all the possible feature subsets to estimate the features' marginal contribution to the prediction

y=1 (probability **0.566**, score **0.264**) top features

Contribution[?]	Feature	Value
+1.673	Sex=female	1.000
+0.479	Embarked=S	Missing
+0.070	Fare	7.879
-0.004	Cabin=	1.000
-0.006	Parch	0.000
-0.009	Pclass=2	Missing
-0.009	Ticket=1601	Missing
-0.012	Embarked=C	Missing
-0.071	SibSp	0.000
-0.073	Pclass=1	Missing
-0.147	Age	19.000
-0.528	<BIAS>	1.000
-1.100	Pclass=3	1.000

Figure 2.9: ELI5's local interpretation of a probabilistic classification model [9]. It contains the contribution of each feature on the model's prediction.

of the original model. The addition of the expected value with the features' marginal contribution of an observation equals to the model's prediction for that observation. In Figure 2.10, SHAP has been used to interpret a black-box probabilistic classifier.

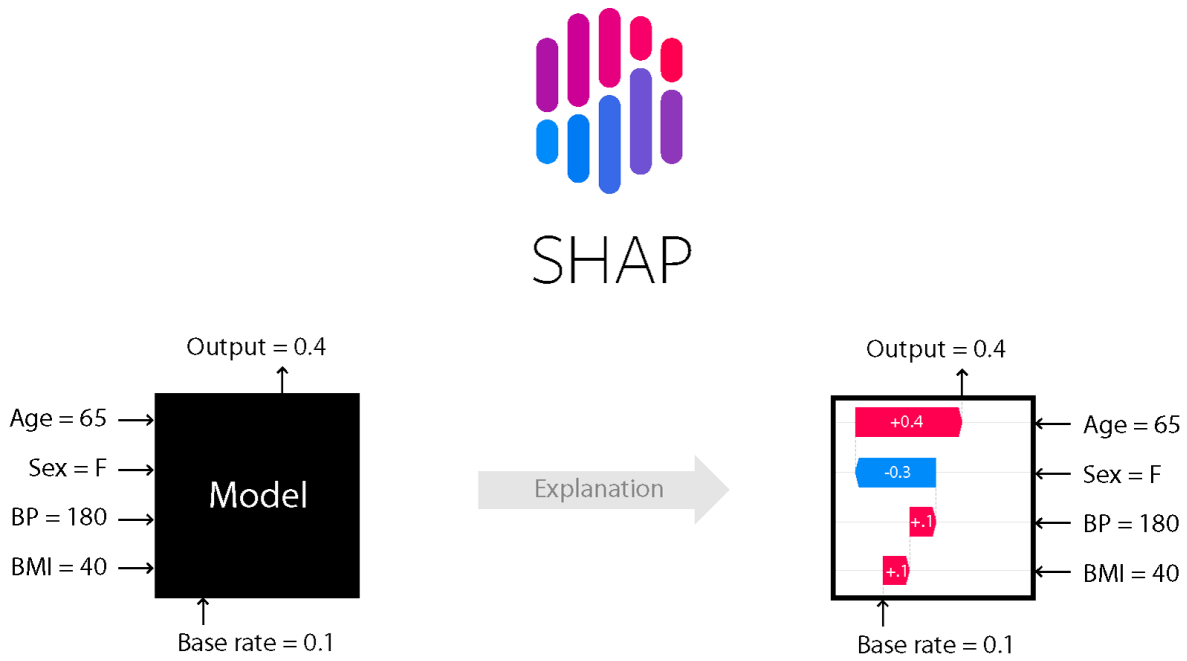


Figure 2.10: Diagram of SHAP used to interpret a black-box model, where each feature's contribution to the prediction with respect to the expected value is indicated [10].

2.2 Predictive maintenance

This section introduces the background of predictive maintenance by explaining different ways to perform PdM, explaining different failure types, related industrial data and monitoring techniques. In addition, the PdM life-cycle is explained in stages, overviewing the possibilities of PdM application and establishing relations among these stages.

2.2.1 Predictive maintenance background

There exist different techniques to create PdM systems. These techniques can be classified according to their underlying methodology, as stated by Liao et al. [78]:

- **Expert knowledge/model-based** methods use the knowledge of the system's failure mechanisms to build a mathematical description of the system's degradation. The result is in a white-box system whose results are easily linked to their physical meaning, but it is difficult to implement in complex systems.
- **Data-driven** methods estimate the state of the machine based on its sensors' data. They are composed of statistical methods, reliability functions and artificial intelligence methods, adopting grey-box and black-box approaches where there is no need to understand the complex system's physics. These systems are more precise than expert knowledge-based ones but their results are difficult to relate to their physical meaning.
- **Hybrid** approach combines the model-based and data-driven approaches, resulting in grey-box systems that can be interpreted.

Failures can be classified into three types regarding their life stage according to Aguiar et al. [79]: in the *initial* working period, they are caused by improper mounting or defects; once the machine functioning is stabilized, *random* failures arise due to inadequate EOC or working conditions, accidents and incorrect maintenance among other reasons; and *wear* failures, which occur due to natural wear of the components.

The term EOC refers to the conditions under which a machine is working [80]. On the one hand, environmental conditions refer to external conditions that affect the machine such as ambient temperature or vibrations produced by trucks driving in the surroundings of the machine. On the other hand, operational conditions are the conditions under

which the machine is working to perform the industrial process, such as working speed, the type of component that is being manufactured or machine adjustments. Some of these conditions are specified by operation technicians while others can be monitored by sensors.

The variables collected from industrial assets are usually classified according to their origin in the process: *Process variables* are the desired operating values for a machine that are set by its operators. *Identifier variables* are the settings under which the process is being executed. *Sensor variables* are all the variables read by sensors placed in the machine, and this data is usually analysed to estimate components' health. *Control variables* are used to specify an action when a sensor variables reach specific actions, for example start the refrigerator when the temperature of a component reaches a threshold. *Derived variables* are sensor variables that have been modified or combined to create new variables that are easy to understand by technicians, which usually imply unit conversions and the application of theoretical formulas. Regardless the origin of the variables, recording their values through time creates a dataset of *time-series* type.

Predictive maintenance models analyse the state of industrial assets by comparing patterns and trends with regard to historical data and domain knowledge, analysing the cause and prognosticating degradation. This comparison and modelling is possible in the data follows patterns such as: *trends* and *seasonality* as indicated by Brownlee [81]. Some algorithms extract features from the data to facilitate modelling, whereas others use the sliding window technique to process the data in chunks. In addition, analysing the variables together adds more context and complementary information, enabling to detect contextual anomalies. However, this multivariate approach adds complexity to the analysis.

The selection and implementation of suitable condition monitoring techniques has a critical impact on the performance of data-driven PdM models, which analyse the data provided by the Condition Monitoring (CM) techniques to estimate the assets' health. The Potential Failure (P-F) curve contained in Figure 2.11 provides a performance degradation plot of a damaged component through time from the beginning of the failure until the asset failure. The plot places the main CM techniques along the degradation time-lapse where they are able to detect the failure. Detecting the damage sooner is preferred to have more time to plan and perform maintenance until the machine stops.

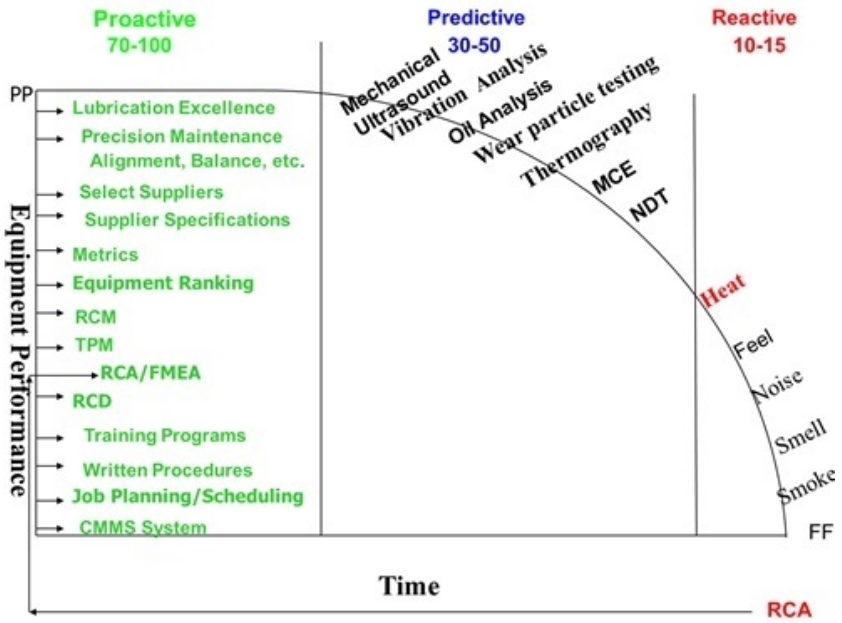


Figure 2.11: P-F curve of a damaged component and techniques to detect failure. Image from UE Systems [11].

In addition, Figure 2.11 contains the main CM techniques: mechanical ultrasound, vibration analysis, wear particle testing, thermography, Motor Current Evaluation (MCE) or Motor Signal Current Analysis (MSCA) and NonDestructive Testing (NDT). Moreover, there exist many other CM techniques as torque, voltage and envelopes [82] or acoustic emission [83]. The articles by Selcuk et al. [24] and Marquez et al. [84] also dive into these techniques and cover the types of failures they can detect, together with their applications.

There are different data collection methods, which can be classified by their origin into real machine and simulation. In the real machine's case, the data is retrieved either manually by an operator or automatically through a CPS-based platform, like the one proposed in the Mantis project by Albano et al. [85]. On contrary, simulations create data aimed at simulating the behaviour of a real machine under different conditions. Simulations can be based on testbeds that emulate a physical machine like Civerchia et al. [86], or Digital Twins such as Borodulin et al. [87], which are software simulations based on theoretical domain knowledge, finite element method, data mining or statistics, among others. A comparison between the simulated and real behaviour of the industrial asset can be used to detect anomalies, and the results of the simulation

can be combined with real data for analysis, like in the work by Borodulin et al. [87]. However, the deployment of data-driven models trained with artificial data to real machines is not straightforward; this is the reason why many research works first validate their methodology on simulated data and then they apply it to real machines, training the models with real data or fine-tuning the simulated models [88].

Several commonly monitored key components in PdM are presented by Zhang et al. in [89]. The work includes bearings, blades, engines, valves, gears and cutting tools, among others. Given that failure modes can be different for each component, two techniques were presented to facilitate their identification: the Failure Modes and Effects Analysis (FMEA) presented by the NASA in 1963 [90], and its evolution by adding criticality analysis Failure Mode Effects and Criticality Analysis (FMECA) [91]. These techniques are still relevant nowadays to prioritise which assets to monitor in search for the most relevant failures.

The work by Li et al. [92] classifies the possible failure types regarding their source as: component failure, environmental impact, human mistakes and procedure handling. Some of the most common failure types are analysed by Selcuk in [24], including imbalance cracks, fatigue, abrasive and corrosion wear, rubbing, defects and leak detection, among others.

2.2.2 Predictive maintenance stages

The predictive maintenance systems are composed by methods that are applied in incremental steps, which can be classified into different stages [93] (see Figure 2.12) according to their purpose. These PdM stages are: the first *anomaly detection*, the second *diagnosis*, the third *prognosis* and the final stage *mitigation*. There are two additional stages previous to these stages that prepare the data to enhance their performance as Khan et al. [16] state, which are preprocessing and feature engineering.

The preprocessing stage cleans and prepares the data for data-driven algorithms. Feature engineering extracts a relevant feature subset representative for the problem to facilitate data-driven algorithms address PdM stages. Anomaly detection is the first stage of PdM roadmap, and consists of analysing if the data collected from industrial assets belongs to a correct or incorrect condition. Most industrial companies have difficulties in collecting faulty data and therefore, the use of semi-supervised models for AD

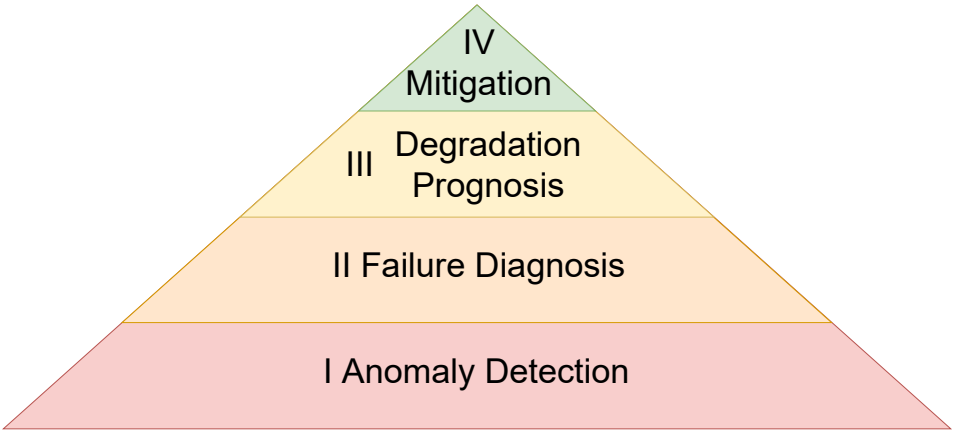


Figure 2.12: Pyramid representing the stages of the predictive maintenance roadmap.

is common to model normal working data and detect novel behaviours.

After detecting that an asset is working improperly, the next PdM stage is to perform diagnosis. Its objective is to perform a Root Cause Analysis (RCA) of the anomaly, evaluating if it belongs to a faulty component, a known failure type or, in contrast, determine that the AD model is not working correctly and needs to be adapted. The diagnosis of an anomaly can be complemented with Health Index (HI) and damage indexes to estimate the damage of an asset.

The third stage of the PdM roadmap is prognosis, which estimates the future degradation of the asset by evaluating the machine's working conditions of that moment, the asset's degradation state and the asset's historical evolution data. When there is a database of run-to-failure observations containing the identified failure type, a supervised regression model can be trained to estimate the Remaining Useful Life (RUL) of the asset. Conversely, if this database does not exist, a semi-supervised prognosis model can be used to estimate the evolution of the asset's health or damage indexes based on the anomaly detection model. The further these metrics are from normal behaviour values, the lower remaining time to failure the asset is expected to have.

The last stage of the PdM roadmap is mitigation. Once an anomaly has been detected, its cause has been diagnosed and its degradation has been prognosticated, there is enough information to mitigate the anomaly in its initial phases and this way avoid failures. The mitigation stage of PdM restores the proper working condition of the machine while reducing the implementation and downtime costs. Mitigation is usually

performed by maintenance technicians who create and implement a mitigation plan. These technicians should combine their experience with relevant information obtained from the PdM system to facilitate mitigation. Integrating the mitigation stage with the industrial MM and MOM processes is essential to ensure an unified maintenance approach that addresses process requirements.

Chapter 3

Review on predictive maintenance

This chapter reviews state-of-the-art predictive maintenance works, introducing relevant works related to this thesis.

On the one hand, PdM applications of physical, knowledge, hybrid, and data-driven techniques are reviewed. Data-driven models include statistical, traditional machine learning and deep learning techniques for PdM applications. Moreover, state-of-the-art data-driven model results are compared in a reference dataset. On the other hand, predictive maintenance standards, norms, and state-of-the-art methodological works are reviewed.

3.1 Predictive maintenance works

This section explains relevant physical, knowledge and hybrid models application for PdM. Moreover, data-driven applications for PdM are reviewed, which are classified into statistical, traditional machine learning, and deep learning models. The review of data-driven models covers all PdM stages and data preparation, including: preprocessing, feature engineering, anomaly detection, diagnosis, prognosis and mitigation. In addition, it contains ways to combine deep learning models with other data-driven techniques, reviews works related to deep learning for PdM, and compares data-driven results on a PdM benchmark dataset.

To perform this review of PdM works, the following research methodology has been defined. The information is gathered from various electronic database-search engines, including Scopus, Engineering Village, Springer Link, Science Direct, IEEE-Xplore, ACM Digital Library and Google Scholar. These resources have provided access to different types of works, including high-impact journals and conference papers. The research

focuses on studies published between 2016 and 2021, although it includes relevant works preceding 2016.

Given the high number of publications in the field, the research space has been limited by defining keywords and research queries. For the physical, knowledge and hybrid models, the following research terms have been used: *("physical models"OR("knowledge-based"AND"models")OR"hybrid") AND "predictive maintenance"*. In addition, the terms *("data-driven"OR"deep learning") AND "predictive maintenance"* are the primary descriptors used to research data-driven PdM techniques. This query has been complemented with the following terms to extend the research to predictive maintenance stages: *"anomaly detection", "diagnosis", "prognosis", "mitigation"* and their preparatory *"preprocessing"* and *"feature engineering"* stages, as presented in Figure 3.1. Finally, the following complementary terms related to industrial requirements were also grouped with the *"deep learning" AND "predictive maintenance"* query (see Figure 3.2): *"transfer learning", "ensemble learning", "reinforcement learning" and "uncertainty modelling"*.

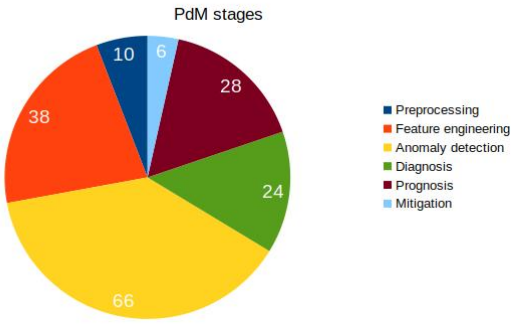


Figure 3.1: The number of data-driven by predictive maintenance stages.

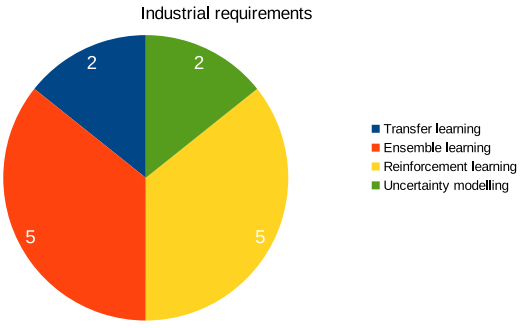


Figure 3.2: The number of deep learning techniques that address industrial requirements by category.

As a result, this section reviews 9 publications that address predictive maintenance using physical, knowledge-based or hybrid models, 92 works that address PdM using statistical and traditional machine learning, 80 publications that address PdM stages using deep learning techniques, and 19 works that combine deep learning and non deep learning data-driven algorithms to create architectures that better address PdM stages.

3.1.1 Physical, knowledge and hybrid models for predictive maintenance

There are two types of models to apply PdM based on domain knowledge: *physical models* and *knowledge-based models*. The literature aims to tackle the stages presented in Figure 2.12.

The review made by Peng et al. [94] summarises that physical models usually employ mathematical models directly tied to physical processes and are developed by domain experts, requiring specific mechanistic knowledge and theories relevant to the monitored systems. Common physical models are based on *first principle modeling* and *parameter estimation*.

Conversely, knowledge-based models embed the knowledge that domain technicians have acquired about the physical systems. Common knowledge-based models rely on *expert systems*, which are based on IF-ELSEs, and fuzzy-logic techniques.

The following papers explain two model-based approaches: stochastic prognostics for RUL prediction on rolling element bearings by Li et al. [95], and physics-based approach for diagnostics and prognostics of cracked rotor shafts by Oppenheimer et al. [96]. The paper by Venkatasubramanian et al. [97] presents a methodology for AD and diagnosis using RCA based on residuals. They also present the 10 desirable characteristics for a PdM system: *quick detection and diagnosis, isolability (distinguish among different failure types), robustness, novelty identifiability, classification error estimation, adaptability, explanation facility, minimal modelling requirements, real-time computation and storage handling, multiple fault identifiability.*

One of the few recent publications is by Blancke et al. [98], who use a multi-state Petri Net to diagnose the type of failure occurring in the system and its evolution to other failure types, contained in Figure 3.3. They first predict intermediate states of

degradation, then integrate expert knowledge into a dynamic model and finally suggest maintenance tasks according to the detected active failure.

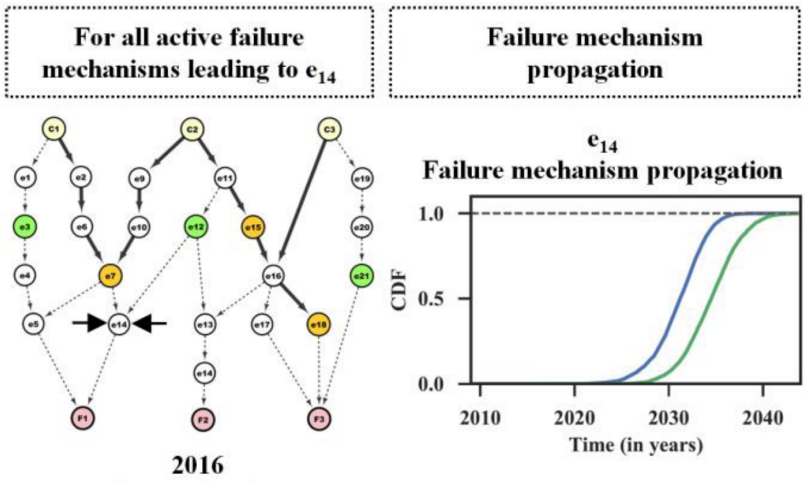


Figure 3.3: Failure mechanism propagation algorithm of Blancke et al. [98].

Li et al. use in [99] discrete stress-strength interference theory to quantify strength and stress probabilities with the objective of modelling expert knowledge for maintenance task assignment.

The other publications tend to use hybrid approaches as the following two studies: a framework that combines model-based and data-driven methods for RUL prediction on lithium-ion batteries by Liao et al. [78], and an integrated prognostics method composed by a hybrid model that qualificates uncertainty for gear remaining life prediction by Zhao et al. [100].

Another recent hybrid approach is by Li et al. [101], whose authors use expert knowledge technicians to review water pipe data and create a model for failure detection and prediction. For that, they first preprocess and visualise the data and then use *Factor Analysis* based on statistical tools, having multiple rounds of discussions and reinforced analysis to reach conclusions of the reasons for failures. To develop prediction models, they use the statistical approach because, as they present, *physical models are usually designed to capture the mechanisms of failures due to certain reasons. But they have significant limitations and statistical models capture hidden patterns of the data and need fewer resources.* They use the divide and conquer method, splitting the data based on attributes. They estimate failure rate with bayesian non-parametric model and use

domain knowledge for inference.

Nowadays there are few models that only rely on expert knowledge due to the difficulty or impossibility to model complex systems mathematically. In addition, expert knowledge elicitation is a bottleneck. Therefore, the research trend is moving towards data-driven models, which are reviewed in the following sections.

3.1.2 Statistical and traditional machine learning models for predictive maintenance

The majority of data-driven PdM models are based on the same principles, no matter if used techniques are statistical, traditional machine learning, or deep learning. Most data-driven methods follow the incremental steps presented in the roadmap of Figure 2.12, based on the articles [25, 93] and Open System Architecture for Bondition Based Maintenance (OSA-CBM) standard [102]: 1st anomaly detection, 2nd diagnosis, 3rd prognosis and lastly mitigation.

Commonly two additional steps are performed before the stated ones to prepare the data for PdM, as general data analytic life-cycle, Khan et al. [16] and other PdM authors present. These steps are preprocessing and Feature Engineering (FE), which are key to enhance model accuracy on PdM stages by creating a representative dataset for the problem. All PdM stages have to be designed, adapted and implemented to fit use-cases' requirements and their data characteristics. In addition, the PdM systems development is incremental and therefore, techniques, algorithms and decisions taken in each stage will influence the following ones. This section overviews the most common data-driven methods to address each PdM stage, excluding deep learning models, which are explained in Section 3.1.3.

3.1.2.1 Preprocessing for data-driven systems

The initial step of PdM is to preprocess the data, preparing it for data-driven models. Each PdM model has different requirements and these must be taken into consideration when choosing adequate preprocessing techniques to boost model performance.

The most common preprocessing techniques are briefly explained and referenced below: sensor data validation [103] makes sure the collected data is correct; feature synchronisa-

tion [104] is used to gather signals sampled at different timestamps to create a time-series or cycle-based data that is easier to handle; data cleaning removes or interpolates not available and missing values [105, 106]; imbalance data handling [105, 107] is applied to boost accuracy on commonly scarce failure data class or to deal with small datasets; encoding and discretisation [108] change features' type by projection to a new space where they are easier to handle by the model; segmentation splits data in chunks to analyse big datasets and enable parallelisation [109]; feature scaling like normalisation [110] (Equation 3.1) or standardisation [111] (Equation 3.2) scales all features to the same or similar space that enables comparisons; noise handling [104] facilitates noisy data modelling. Complementary information of preprocessing techniques can be found in the article by Cernuda et al. [112] on preprocessing for predictive maintenance.

$$X_i^S = \frac{X_i - X_{min}}{X_{max} - X_{min}} \tag{3.1}$$

$$X_i^S = \frac{X_i - mean(X_i)}{var(X_i)} \tag{3.2}$$

3.1.2.2 Feature engineering for data-driven systems

This step consists of extracting a relevant feature subset to be used as input for models in further stages. It can boost statistical and machine learning model performance, despite not being compulsory for deep learning models given these can extract new representative features that fit the problem automatically. The most common techniques can be grouped into next groups: feature extraction as statistical features in time [89] and frequency [89, 113, 114] domains that extract time and frequency relations of features; based on projection to new space like principal component analysis [115, 116] which reduce dimensionality while keeping relevant information; concatenation and fusion methods [117] create new features by combining available ones; feature selection [118] reduces dimensionality discarding features of low variance, redundant and uncorrelated to target, given these increase complexity while not supplying additional information.

3.1.2.3 Anomaly detection

Anomaly detection aims to detect whether the asset is working under normal condition or not. There are three ways to address this step using data-driven models, classified by their underlying machine learning task: classification, one-class classification and clustering. Respectively, these can be used when labeled data of different classes is available in the training phase, when only one class data exist (commonly non-failure data), and when the data is unlabelled. FMEA as explained by [90], and its evolution by adding criticality analysis FMECA [91] are useful to gain vision on the possible types of failures based on expert knowledge, which helps in the design of the data analysis life-cycle by prioritising the failure types or anomalies to be detected.

The anomaly detection methods need preprocessed and some also depend on feature engineered data to work. Once worked on features, the next step is to select, train and optimise the right model for the use-case. Following PdM stages will be influenced and constrained by the selected AD method and use-case's data. Table 3.1 classifies and summarises the main data-driven anomaly detection techniques based on referenced state-of-the-art articles and the following review works [89, 119, 120, 121]. Besides, two or more of these techniques can be combined to create an anomaly detection system that compensates the disadvantages of a single model.

3.1.2.4 Diagnosis

Once an anomaly has been detected, the next stage consists of diagnosing whether this anomaly belongs to a faulty working condition and can evolve into a future failure or, in contrary, there is no risk of failure.

The diagnosis algorithm has to be suitable for the problem being addressed. There are several approaches to tackle this step, which depend on the implemented AD method and training data characteristics: multi-class classification, binary classification, one-class classification and clustering. Concretely, these are chosen if the dataset has multiple failure types, failure and non failure observations, only observations of one class or unsupervised, respectively. There is another technique that commonly complements RCA: anomaly deviation quantification based on health index. It aims to measure assets' damage by comparing current working data with historical data in a supervised or unsupervised way. It can either indicate a percentage of deviation with regard to

Table 3.1: Summary of data-driven anomaly detection models classified by prevailing techniques. In the first column, Unsup refers to unsupervised, All refers to supervised, semi-supervised and unsupervised and Combination refers to a combination of models respectively.

Based on and Type	What analyses	Normal data	Anomalies	Most common algorithms and categorised
Density Unsup	Density in features space dimension	In high density	In low density	K Nearest Neighbors (k-NN) [106, 107, 122, 123], Local Outlier Factor (LOF) [105, 124], LOcal Correlation Integral (LOCI) [2], relative density factor, density-based outlier score, reliability functions [125, 126]
Distance Unsup	Distance among data-points	Near from neighbors	Far from neighbors	Traditional threshold distance on mahalanobis [110] or euclidean [127], Rank Based Detection Algorithm (RBDA), randomization and pruning based, data streams based.
Statistics All	Relation to distribution models fit to training data	Near to distribution models	Far from distribution models	*Parametric*: Gaussian Mixture Models (GMM) with Expectation Maximisation (EM) [128], control charts as Exponentially Weighted Moving Average (EWMA) [129, 130]. *Non-Parametric*: Kernel Density Estimation (KDE): gaussian or KL-divergence [130, 131, 132], Histogram-Based Outlier Detection (HBOS) [133], boxplot analysis [105], 3σ [134]. *Entropy-based* permutation entropy [135, 136], fuzzy entropy [137] and K-S test [138].
Clustering Unsup	Relation to clusters created by unsupervised ML models	Belong to a large cluster or near one	Belong to a small cluster and far from large clusters	*Partitioning clustering*: Partitioning Around Medoids (PAM), K-means [124, 128, 139]. *Hierarchical clustering*: DB-Scan, agglomerative [105], attribute oriented induction [140]. *Grid-based*: Dcluster. *For high dimensional*: D-Stream, fuzzy-rules based [124]
Ensemble Combination	Combines dissimilar models. Robust	Combination of models	Combination of models	Bagging or boosting based as Random Forest (RF) [106, 115, 116, 123], eXtra Gradient Boosting (XGBoost) [105], adaboost [123] and Isolation Forest (IF) [107], greedy ensemble, score normalization.
Learning All	Relation to models learned with training data	Near the known classes of the model	Far from the known classes of the model	*Active learning. Transfer learning. Reinforcement learning. Projection-based*: Subspace and compression reconstruction error measuring like PCA [128] and AE [141], correlation [142, 143] and tensor-based. *State-space based* (hidden state of observed data and time evolution): kalman filter [144], Hidden Markov Models (HMM) [107], Bayesian Network (BN) [145] (dynamic BN, belief network), attention-based NN and RNN (GRU, LSTM). *Graph-based*: capture interdependiencies. *OCC*: OC-SVM [113], BN. *Prediction error-based regression*: measure deviation (AutoRegressive Integrated Moving Average (ARIMA) [110], RNN as LSTM [146]). *Classification*: normal and abnormal data in training using interpretable models: linear regression [123], logistic regression [106, 123], Decision Tree (DT) [106, 147]. ML classification techniques as SVM [106, 122, 147] and feedforward NN [148]. *Generative methods*: GAN [149], VAE [150].

normal working data, or show degradation level in a numerical scale, where the higher the value the more damaged the component is, minimum value means no damage, maximum is fully damaged or failure, and intermediate values indicate different degrees of degradation [151].

The diagnosis step is easier when there is more information about the dataset and its labels. The main statistical and machine learning techniques for diagnosis are described in the following list, ordered by increasing difficulty. They are divided according to the anomaly detection technique used in the previous stage, which depends on data characteristics.

- After multi-class classification for anomaly detection: diagnosis is performed based on previous failure data knowledge of the estimated class, so the link of data to failure type is directly obtained from model [152, 153]. Once the possible failure type has been detected, semi-quantitative and qualitative approaches can be used by harnessing expert knowledge to evaluate its potential consequences, using tools such as FMEA [154] or ishikawa diagram [155]. In addition, interpreting directly explainable models [156, 157] or using explainability on less interpretable models such as Support Vector Machines (SVM) [158] can also help to perform this task.

- After binary classification for anomaly detection: clustering with extracted features can be performed to group data by similarity and try to differentiate unlabeled failure types [159]. These diagnosis techniques can also be based on statistical performance analysis [160], supported on trend analysis and definition of thresholds to differentiate failure types by similarity or distance.

- After one-class classification or clustering for anomaly detection: these techniques use a threshold in distance to the classified class or clusters density respectively to categorise anomalies. Diagnosis for these models usually consists of precomputing metrics from data like health index and monitoring their evolution, instead of monitoring input data evolution. The diagnosis can be performed using a clustering algorithm in these metrics to analyse the intra-cluster and inter-cluster relations. Domain knowledge is essential to tie unsupervised and semi-supervisedly discovered relations to physical meaning of monitored assets. This novel knowledge is

[2]Methods that have been applied for AD in general but not specifically for PdM are mentioned but not referenced

useful for interpreting unsupervised and semi-supervised models' output to discover novel failure types, using models as K-means [161] or HMM with if-else rules [162]. Log data can also be used for this clustering purpose and tag maintenance data [163] to perform RCA.

3.1.2.5 Prognosis

Once an anomaly is detected and diagnosed, the degradation evolution can be monitored based on that moment's working conditions and machine state, focusing on the most influential features for AD and diagnosis stages that can track failures.

This step is carried out by remaining useful life models when degradation data is available, and prognosis of HI and damage indexes when it is not. These techniques can also provide a confidence bound. The data-driven models for prognosis can be classified into 4 groups regarding their underlying method. The following list summarises the most common techniques categorised by groups to prognosticate degradation:

- Similarity-based: compare current behavior with past run-to-failure behavior for prognosis [161, 164].
- Statistical: rely on historical statistics to estimate degradation, for example monitoring life usage in combination with mean-time-to-failure [165] or survival models [166] to estimate the expected duration.
- Time series analysis: ARIMA [161, 165, 167] based on previous values, kalman filter to model hidden state of time-related noisy data [144], and fourier and genetic programming to generate a polynomial function by optimising a fitness function [114].
- Learning: learn patterns from data. Divided in two types: classification and regression.
  - Classification: diagnosis relating the data to a known failure type or similar working data and then prognosticate a degradation according to the historical data of this class. Despite any classifier can be used for this purpose, the following ones are widely used in literature: feed-forward NN [148], SVM [148], BN [156, 168, 169], HMM [170], fuzzy logic based [171] and RF [165, 172].

– Regression: directly estimate HI, anomaly deviation or RUL from the input data. Common state-of-the-art algorithms include: linear function as the simplest method [163]; nonlinear functions [166, 173] can model non-linear relations; Support Vector Regressor (SVR) [165, 174] works like SVM adapted for regression; Relevance Vector Regression (RVR) is based on bayesian regression [161]; CNN models features' time-based relationships [175]; wiener processes model degradation by a real valued continuous-time stochastic processes [176]; recurrent neural networks like LSTM and GRU [177] retain relevant past information for prognosis at each observation.

3.1.2.6 Mitigation

After detecting an anomaly, diagnosing its cause and prognosticating its evolution, there is enough information to design and implement a maintenance plan. The implementation of this plan includes the necessary steps to restore assets to their correct working condition before failures occur, which reduces implementation and downtime costs.

Data-driven PdM models should generate assistance information, providing domain technicians with statistics [160] and prescriptions [156]. Therefore, a more advanced mitigation is accomplished by the combination of domain knowledge and data-driven information about assets' health and expected degradation [178].

3.1.3 Deep Learning-based predictive maintenance

This section collects, summarises, classifies and compares the reference DL techniques for PdM, analysing the most relevant works and applications. It contains accurate DL models that achieve state-of-the-art results on reviewed articles, surveys and reviews of the field. Even though most articles combine several techniques and perform more than one PdM stage in the same architecture, this section classifies the works by their principal DL technique to perform each stage of Section 2.2; including how to perform feature engineering to prepare the data for PdM stages. This classification enables the analysis and comparison of DL techniques by stages. This section also presents works that successfully combine the aforementioned techniques to create more complete architectures that fulfil one or more PdM stages, giving examples of ways to combine techniques that can be infinite. Finally, the most relevant information of similar works is discussed.

The state-of-the-art works can be further classified regarding their underlying ML task and algorithms used to address it, which are directly related to the use-case and its data requirements. Binary classification is selected when training data contains labelled failure and non-failure observations. Multi-class classification is used in the same case as binary classification, but there is more than one type of failure classified and therefore there are at least three classes: one represents non-failure and then one for each type of failure. One Class Classification (OCC) semi-supervised approach is applied when the training dataset only contains non-failure data, which usually consists of collecting machine data in early working states or when technicians assure the asset is working correctly. Finally, unsupervised techniques are used when training dataset is unlabelled and therefore there observations' failure or non-failure labels are unknown. Additionally, there are a few works on other machine learning and deep learning topics such as active learning, reinforcement learning and transfer learning.

3.1.3.1 Feature engineering with deep learning

The deep learning algorithms used in PdM are capable of performing feature engineering automatically, obtaining a subset of derived features that fit specifically for the task. A common technique is using feed-forward by adding deep layers with less dimensions. Restricted Boltzmann Machine (RBM) also provide automatic feature extraction by modelling data probability with contrastive divergence minimisation, based on one-way training and reconstructing input from output. Likewise, Deep Belief Network (DBN) enable automatic feature extraction using stacked RBMs with greedy training, which can also be used for HI construction. Moreover, SOMs map data to a specified dimension, and AEs reduce dimensionality in latent space while keeping maximum input data variance, providing non-linear FE and HI calculation. In addition, CNNs automatically extract features by univariate or multivariate convolutions of input, thus modelling sequential data with sliding windows. CNNs are usually combined with pooling methods to reduce dimension. Finally, RNNs use regression to model time-series and sequential data by propagating state information through time.

These feature engineering techniques remove the dependence on manual and feature engineering process. Table 3.2 shows strengths, limitations and referenced applications of common deep learning techniques used for feature engineering. These techniques are integrated with machine learning and deep learning models to create architectures that

perform PdM stages.

Feed-forward networks are unable to model the temporal relations of industrial sensor data for feature extraction, but they can fuse nontemporal features to reduce the dimensionality of the feature set when used inside an AE. AEs have the ability to extract features automatically; therefore, they are suitable for extracting representative features to perform semisupervised and unsupervised predictive maintenance. However, like the feed-forward models and RBM, DBN and SOM, AEs depend on the use of CNN and RNN layers to extract time-based relations.

RBMs are simpler and faster to train than feed-forward networks, but they have difficulty in modelling complex industrial data because they are composed of a single layer. DBNs address this issue by stacking RBM layers; thus, they achieve SotA results in industrial data by modelling temporal relations with sliding windows. However, the use of sliding windows limits the long-term modelling capabilities of RBMs.

CNNs are suitable for modelling individual sensor relations with one-dimensional filters and can also model time-based relations among sensors by using two-dimensional filters. Their main advantage is that by weight sharing, they reduce the required training resources and model complexity, but they have limited memory. RNNs with specific architectures can extract longer temporal data relations among sensors, but their memory is still limited by the vanishing gradient problem. In addition, they add complexity and therefore increase the explanation difficulty of the network. Explanation difficulty is a challenge that PdM models must overcome before being deployed to production.

3.1.3.2 Anomaly detection with deep learning

The deep learning-based AD algorithms can be classified in three groups according to training data characteristics, as stated in the introduction of this section. The main architectures have been summarised in Figure 3.4.

Those algorithms are summarised, compared and their main applications are referenced in the following tables. On one hand, the anomaly detection algorithms based on binary and multi-class classification approaches [148, 180] rely on training data classified as correct and failure. These commonly used feature extraction techniques either traditional or deep learning followed by a flatten process, and then have several fully-connected layers of decreasing dimension until the output layer. In the case of binary classifi-

Table 3.2: Deep learning techniques for automatic feature engineering and projection. These techniques are based on input signal relations and temporal context.

Algorithm	Advantages	Disadvantages	Applications and refs
Feed-forward	- Reduce dimension to lower feature space - Simplest NN architecture	- Not model the features by neighborhood - Not model temporal relations	Engine health monitoring [148, 179], bearing fault diagnosis [180]
RBM	- Keep spatial representation in new space - Not much training time	- Not keeping data variance in new space - Difficulty on modelling complex data since only one layer	Bearing degradation [181], factory PLC sensors [182]
DBN	- Competitive SotA results - Can model time-dependencies using sliding windows	- Slow training - Not modelling long-term dependencies.	Vibration analysis [183], bearing prognosis [184], engines [185, 186], wind turbine [187]
SOM	- Non-linear mapping of complex data to a lower dimension - Maintains feature distribution in the new space - Can be combined with other techniques for RCA (i.e. 5-whys [188])	- Difficult to link latent variables with physical meaning - More complex than other techniques - Fixed number of clusters	Turbofan [189], pneumatic actuator [190], thermal power plant [188], bearing degradation [181]
AEs	- Automatic FE of raw sensor data achieve similar results to traditional features [1] - Traditional features can also be input - No need of classification or failure data - Allows online CM.	- Extracted features not specific for the task - Needs more resources: computational and training data - Loses temporal relations if input data are raw sensors data - Can lead to overfitting	Bearing vibration [141, 191, 192], satellite data [193], CAN vehicles [107]
CNN	- Simple yet effective - Faster than traditional ML models in production - Takes advantage of neighborhoods - Less training time and data by weight-sharing - Can outperform LSTM - Dropout can prevent overfitting	- Slower training due to high number of weights - Data analysis in chuncks, not modelling long-term dependencies.	Bearing diagnosis [194, 195], electric motor [196], gearbox [197], turbofan [175, 198], Numenta Anomaly Benchmark [199], blade [200]
RNNs	- Model temporal relationships of EOC data - Special architectures as LSTM and GRU can model medium-term dependencies	- RNNs suffer vanishing gradient problem, even special architectures cannot model very long-term dependencies - Need more resources	turbofan [177, 201, 202], hydropower plant [146]

[1] In this section, the term traditional features refers to handcrafted and automatic feature extraction techniques such as statistical or ML-based features, excluding DL-based features.

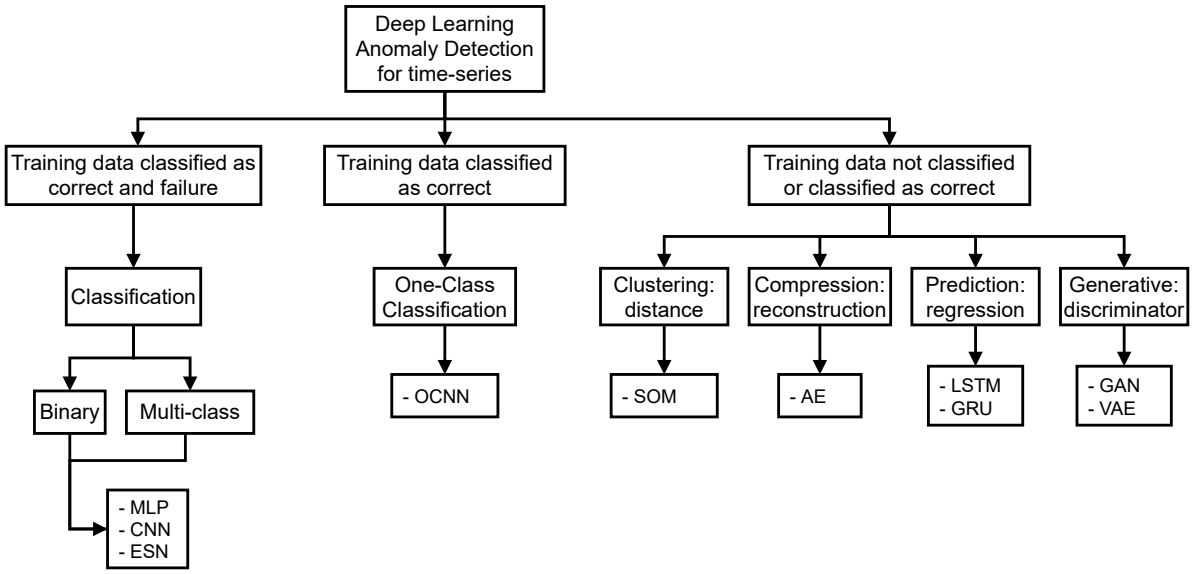


Figure 3.4: Diagram of main deep learning techniques for anomaly detection in predictive maintenance [3].

cation, there are one or two neurons indicating the probability of failure and normal working condition. Similarly, in multi-class classification there are N+1 number of neurons, where there is one neuron to indicate the probability of not failure and each of remaining N indicate the probability of each type of failure.

Another type of algorithms address the AD problem based on one-class classification or unsupervised approaches, using only training data classified as correct or not classified. Autoencoder structures are widely used for this propose, where vanilla AEs uses a threshold in reconstruction error to classify as anomalous data that surpasses it. Stacking more than one AE after another is denominated as stacked AE, whereas SAEs constrained training with sparsity to keeping neurons' activations low, and DAEs are AEs designed for noisy data. The generative VAE is an AE that maps input data to posterior distribution, and GANs are used for data augmentation and AD in 2 ways: using discriminator and using residuals.

One additional one-class technique is OC-NN, which trains AE and freezes encoder for one-class classification that is similar to OC-SVM loss function. Vanilla RNNs are also used for AD in the tracking error between predicted and real behavior using regression,

[3]In this section, the term traditional features refers to handcrafted and automatic feature extraction techniques such as statistical or ML-based features, excluding DL-based features.

and measuring HI difference. Similarly, LSTM and GRU neural networks are used replacing neurons architecture by LSTM and GRU neurons respectively. A comparison on strenghts and limitations, together with applications and references of these techniques are displayed in Table 3.3.

Autoencoders are trained to detect anomalies in industrial data using unsupervised or one-class data; a vanilla AE is the simplest version. Stacked AEs achieve better performances but at the cost of increased complexity and additional resources. SAEs penalise the weights of the autoencoder to limit complexity, which can be used to prevent overfitting of anomaly detection algorithms, and DAEs are more complex and robust to noisy data, making them suitable for addressing vibration data. An OC-NN works as a one-class neural network that can be trained in a semisupervised way. While it cannot extract time-based relations, this ability can be achieved by combining an OC-NN with CNN and RNN layers.

Regarding generative models, a VAE learns the posterior distribution of the sensor data, but the random component can make model interpretability difficult. GANs additionally enable data generation, which can be useful for generating synthetic failure data when only a few failure observations have been collected, and they can achieve SotA results in semisupervised anomaly detection. However, GANs have difficulties handling datasets with high imbalance ratios, their complexity makes them difficult for industrial stakeholders to interpret, and sometimes they are outperformed by simpler methods. RNNs are widely used to evaluate the evolution of industrial asset signals over time and detect anomalies, but the vanilla version cannot model long-term dependencies. LSTMs and GRUs fix this vanishing gradient problem, so they have currently replaced vanilla RNNs. The choice of one model type over the other depends on the specific use case being addressed.

3.1.3.3 Diagnosis with deep learning

The diagnosis steps depends on the information and type of AD model used for the previous stage, given PdM is an incremental process where each stage is complemented by previous stages. In the case of multi-class classifier, the type of failure related to the detected anomaly is already known, which enables a straightforward diagnosis and comparison with historical data [148, 180]. Nonetheless, most PdM architectures implement binary classifier, one-class classifier or unsupervised models, which lack of fail-

Table 3.3: Anomaly detection methods that use training data classified as correct or not classified: one-class classification and unsupervised.

Algorithm	Advantages	Disadvantages	Applications and refs
Autoencoders			
Vanilla AE	- Automatic feature engineering of raw sensor data or traditional features - Minimise variance loss in latent space - No need of classification or failure data - Allows online CM	- Extracted features not specific for the task - Needs more resources: computational and training data - Loses temporal relations if input data are raw sensors data - Can lead to overfitting	Bearing vibration [141, 191], satellite data [193], flight data [203], CAN vehicles [107], marine autonomous systems [204]
Stacked AE	- Perform slightly better than vanilla AE	- Needs more resources than vanilla AE	Bearing vibration [205, 206], generator turbine vibration [207]
SAE	- Same as AE plus prevent overfitting by forcing all neurons to learn	- More complex networks and need more resources than vanilla AE	Bearing vibration, turbine vibration [208], [209], [207], [192]
DAE	- Outperform vanilla AE with noisy data - Works slightly better stacking several DAEs	- More complex networks and need more resources than vanilla AE - stacked DAE needs even more	Bearing vibration [208, 210]
Generative			
VAE	- Learns posterior distribution from noisy distribution, generating data non-deterministically	- Difficulty on implementation - Loses temporal relations if input data are raw sensors data.	Ball screw [132], electrostatic coalescer [211], web traffic [150], aircraft data [212]
GAN	- Good data augmentation with small imbalance ratio - AD outperform unsupervised SotA methods	- Not working well with big imbalance ratio - complex and need more resources - Outperformed by simpler methods as CNN [194]	Induction motor [149], bearing multisensor [194]
One-Class Classifiers			
OC-NN	- Automatic feature extraction	- Slower than traditional OCCs - Extracted features are not focused on the problem	General AD [213]
Recurrent Neural Networks			
Vanilla RNN	- Model temporal relationships of time-series data - Self-learning.	- Suffers vanishing gradient problem; therefore cannot model medium and long-term dependencies - Need more resources than feedforward AE or CNN for training.	Activity recognition [214]
LSTM	- Same as vanilla RNN, however these can model longer time dependencies than vanilla	- Even if handle better the vanishing gradient problem than vanilla, have difficulty on modelling long-term dependencies - Long training and computational requirements	Aircraft data [215], activity recognition [214], nuclear power machinery [216]
GRU	- Comparable to LSTMs plus easier to train	- Comparable to LSTMs	Aircraft data [215], activity recognition [214]

ure type information. Therefore, these can only perform diagnosis by grouping the detected anomalies among them by similarity, which is done using clustering models [212, 217, 218, 219, 220] and SOM [221, 222, 223, 224]. The features used for this stage are similar to the ones for AD, which can be based either on traditional or deep learning techniques.

3.1.3.4 Prognosis with deep learning

The most common deep learning algorithms for prognosis are summarised and compared in Table 3.4. The Vanilla RNNs, Gate based RNN networks (LSTM or GRU) can be used for Regression, predicting features' and Heath Index evolution or predicting remaining cycles or time. Their input can be the information generated in previous stages and traditional or deep learning features. This section focuses on the most common and simple state-of-the-art techniques that only use DL for prognosis, whereas prognosis works that combine DL with traditional features are presented in the combination of models and remarkable works part of this section.

Table 3.4: Summary of DL-based prognosis works for PdM. The terms "unsup" and "sup" in the algorithm column refer to unsupervised and supervised respectively.

Algorithm	Advantages	Disadvantages	Applications and references
RNNs	Model temporal relationships of time-series data. Possibility for self-learning	Suffer from vanishing gradient problems; therefore, they cannot model medium and long-term dependencies. They have lengthy training and high computational requirements	Aero engine [177]
LSTMs	Same as a vanilla RNN; however, LSTMs can model longer time dependencies than can vanilla RNNs, and they outperform vanilla RNNs	Although LSTMs handle the vanishing gradient problem better than vanilla RNNs, they still have difficulty modelling long-term dependencies and have lengthy training and high computational requirements	Aero engine [177], rolling bearing [225, 226], lithium batteries [227, 228]
GRU	Same as LSTMs but easier to train	Same as LSTMs but may achieve slightly worse results	Aero engine, lithium batteries [177, 227]

The use of LSTMs and GRUs is more common than that of vanilla RNNs given that

they allow the modelling of longer time dependencies. LSTMs are more commonly used for prognosis in the PdM field, whereas GRUs achieve similar results but are simpler and therefore easier to train. The choice of one model type over the other depends on the addressed use case.

When target failure types are known and there is either a priori knowledge or observations of target class is available, uncertainty quantification can help to identify which predictions of the generated model are trustable and which not. This is specially relevant in the case of prognosis, given that as the prediction time horizon increases, prediction uncertainty is higher. A common technique to quantify uncertainty of data-driven models is by bayesian inference, which is implemented in articles presented by Wang et al. [229] and Kraus et al. [230]. However, when there is not enough data collected from the target failure types or it is tackled as one-class classification, the aforementioned techniques cannot be used; in this cases, self-supervised metrics like variance gain relevance for uncertainty modelling.

3.1.3.5 Mitigation with deep learning

The research methodology followed to research state-of-the-art, showed few DL-based mitigation publications. Several possible reasons for this fact are described bellow. The majority of DL works are focused on optimising a single performance metric for the ML task to be solved, like maximising accuracy or F1 score on classification, and minimising errors like MAE or RMSE on regressions. These works' solutions are usually compared in simulated reference datasets, looking for the architecture that outperforms the rest on the aforementioned metrics. Nonetheless, deep learning models are the hardest ML type to understand given their higher complexity that makes them more accurate at modelling high dimensionality complex data, and therefore they fail to meet the industrial explanation facility requirement.

The publications that generate automatic data-driven maintenance policies using deep learning models for PdM are based on reinforcement learning, an emerging trend in this field. The article presented by Paraschos et al. [231] uses reinforcement learning for the generation of control policies that optimise maintenance on degrading failure manufacturing system. Moreover, Rocchetta et al. [232] present a reinforcement learning framework to optimise power grids maintenance using Q-learning on a fully-connected neural network. Likewise, the work by Hoong Ong et al. [233] proposes an automatic

learning framework that creates optimal maintenance decision policies based on machine health state, which is derived from sensor data and proposes actionable recommendations.

Predictive maintenance systems should provide mitigation advice or at least explanations about the reasons for making predictions, which could be supported on the emerging field XAI. Furthermore, the final and most ambitious step in this PdM stage should be the automatising of recommendations for domain technicians to integrate PdM in the maintenance plan, by optimising industrial maintenance process via maintenance operation management.

3.1.3.6 Combination of deep learning models and remarkable works

The DL techniques already presented throughout current section are the basic elements and architectures used for PdM. It is worth highlighting there are infinite possible architectures by combining these techniques among them, or used together with other data-driven or expert-knowledge based techniques. The combination and adaptation of models for the problem being addressed results into more accurate models that fulfil its requirements. Table 3.5 summarises how these models are commonly combined in architectures, presenting their strengths and limitations.

Moreover, Table 3.6 contains relevant works of the aforementioned types, which merge traditional FE or deep learning FE with traditional data-driven or deep learning models. This collection of works shows that combination of techniques can address all PdM stages using supervised or unsupervised approaches.

The principal deep learning works for PdM have been reviewed, even though the number of possible architectures is infinite by combining and adapting the presented techniques. Despite this fact, several common architectures of reviewed publications for anomaly detection, diagnosis and prognosis are presented in Figure 3.5, Figure 3.6 and Figure 3.7 respectively.

Table 3.5: Possible combination of deep learning techniques for PdM architectures.

Algorithm	How they work	Strengths	Limitations
Traditional and DL features combined with DL models			
Traditional and DL-based FE with AE	Combine traditional and DL FE methods with already presented autoencoder architectures in the same model	Outperform traditional ML and simple DL architectures. No need of handcrafted features. Automatic FE. Can model time-series dependencies using CNN, LSTM and GRU by context extraction	Understanding deep features is not straightforward. Slower and more complex than simple ANN models
Traditional and DL-based FE with DBN	DL and traditional FE methods with DBN stacked to other models	Same as above	Same as above
Hybrid: combination of features and models			
DL FE techniques combination	Combine CNN, LSTM, other DL FE techniques and traditional features to extract more complex features	Automatic dimension reduction. Outperform other FE techniques. Model temporal relations and neighbors. With bidirectional RNNs, future context is available	More complex and need more resources than traditional ML and simple DL models. Bidirectional RNN cannot be done online

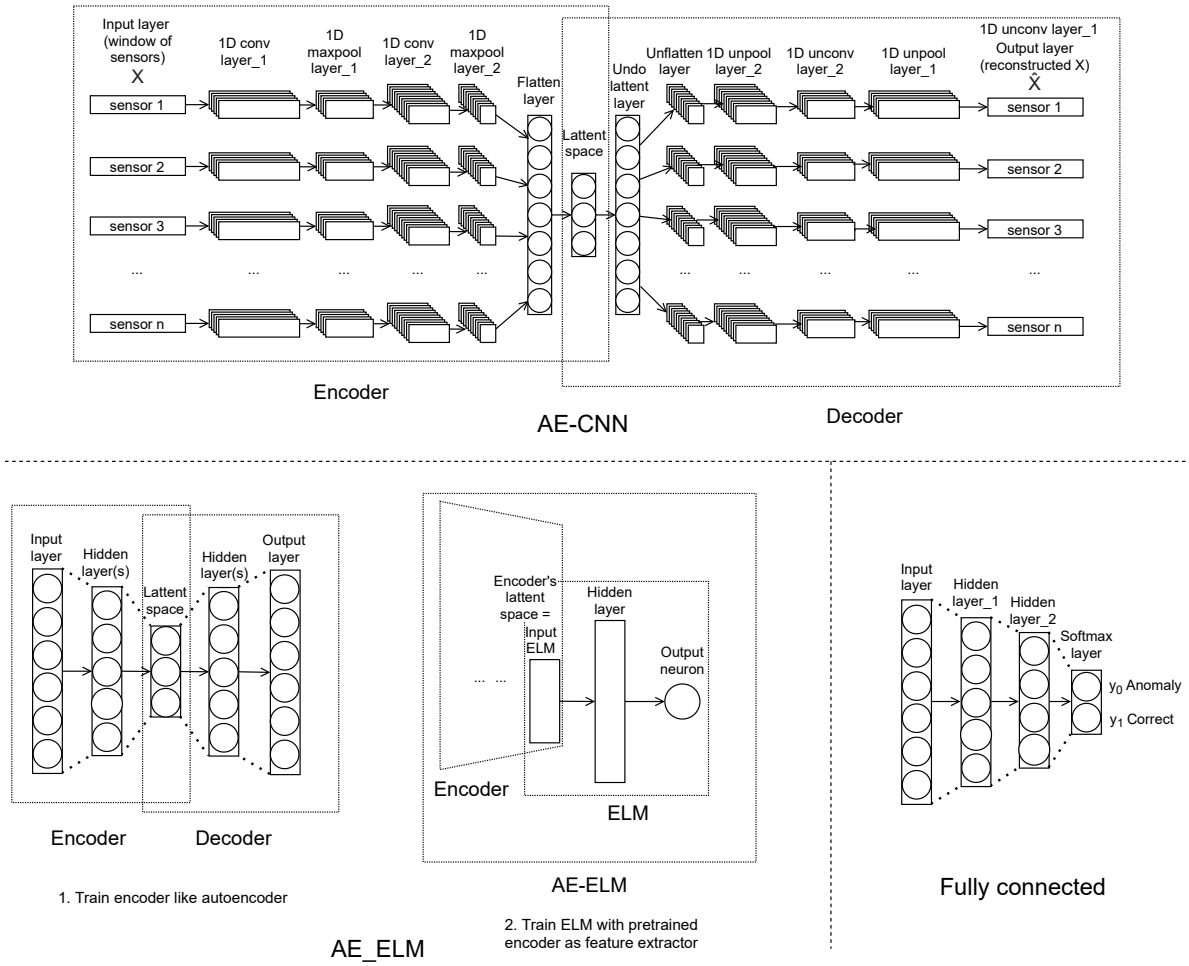


Figure 3.5: Diagram of three common deep learning architectures for anomaly detection in predictive maintenance: convolutional autoencoder on top, autoencoder-based extreme learning machine on bottom left and autoencoder-based ELM in bottom right.

Table 3.6: Combination of deep learning techniques for PdM: relevant works summary.

Architecture	How it works	Strengths	Limitations	Applications and refs
Autoencoders				
AE with ELM.	Unsupervised AD tracking error of ELM for OCC, trained with normal data.	Two steps training. Easy to train.	Unable to model non-linear or complex relations in ELM.	Power plant [234], machine lifetime estimation [235].
Stacked SAE	Unsupervised FE adding noise	No need of preprocessing. Robust to noise. Severity identification.	Difficult optimisation of deep architecture	Rolling bearing [209]
Stacked CNN-based AE	Unsupervised FE modelling temporal relations in sliding window	Model temporality using neighbours.	Only short temporal relations	Gearbox vibration [236]
AE with LSTM	Unsupervised FE modelling temporal relations	Model temporality	Higher computational requirements	Aviation [237], turbofan and milling machine [238], solar energy, electrocardiogram [239] and manufacturing [240]
VAE with RNN, GRU or LSTM	Unsupervised generative FE modelling temporal relations and reducing to latent gaussian distribution	Model temporality. Regularised latent space	High computational requirements	Motor vibration[237], turbofan [241], sensors [242]
Restricted boltzmann machines and deep belief networks				
DBN	Unsupervised FE by hierarchical representations	Fault classification from frequency distribution	Need preprocessing. Tendency to overfitting. Not modelling temporal relations	Induction motors fault simulator [186]
Regularised RBM + SOM + RUL	Probabilty modelling, health assesment and RUL prognosis using distance	RBM regularisation improve FE for RUL	Single RBM, can be improved by multiple of these layers.	Rotating systems [181]
Image generation + DBN + MLP/FDA/-SOM	Supervised or unsupervised FE modelling from vibration image data	Model temporality in an image. Combine with image processing methods	Difficulty on extracting clusters' meaning, relying on domain knowledge.	Journal bearing [243]
Hybrid: combination of features and models				
Bidirectional LSTM	Unsupervised FE modelling temporal relations	Health estimation and then RUL mapping. More robust. Future context is available.	Need all signal to be processed: no streaming. More complex than simple LSTM.	Turbofan [244]
AE + Convolutional DBN + Exponential Moving Average (EMA)	Unsupervised probability modelling by automatic FE, modelling temporal relations. Training in steps	Model temporality. More stable than traditional ML and simple DL. Each model complement others weaknesses	Each part trained independently, not for problem. EMA only model shorter term temporal relations.	Electric locomotive bearing fault [245]
CNN and bidirectional LSTM based AE + fully connected + linear regression	Unsupervised FE modelling temporal relations	Raw sensor data modelling. Model long-term temporal dependencies	Sliding window needs complete window. Higher complexity combining DL techniques	Milling machine [246]
Traditional FE + bidirectional GRU combined with ML models	Unsupervised FE modelling temporal relations	Same as above	Same as above	Aviation bearing fault detection, gear fault diagnosis and tool wear prediction [247]

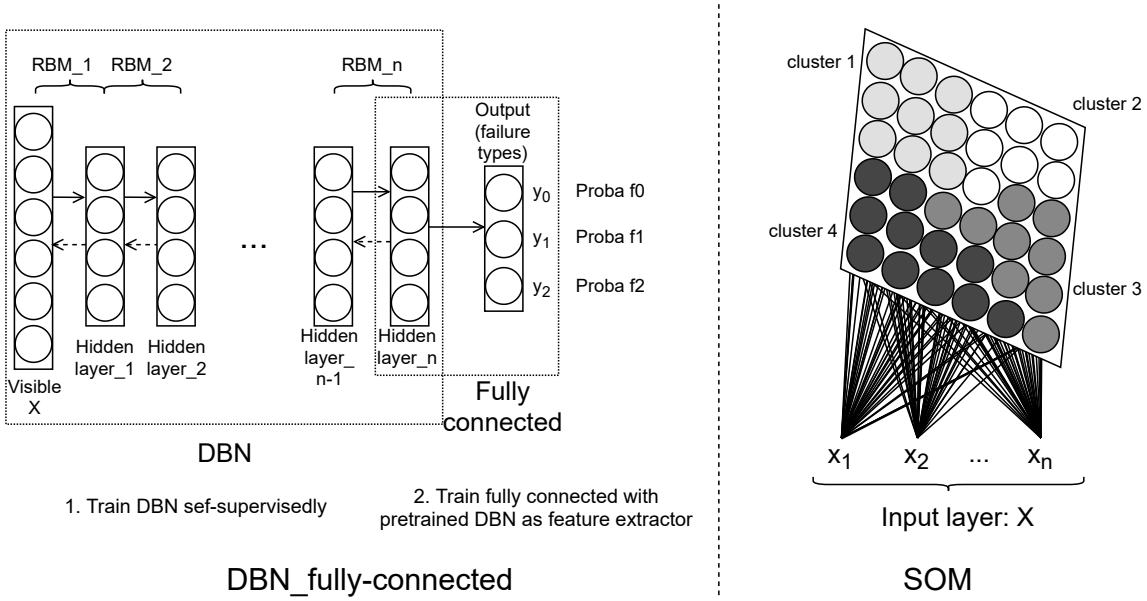


Figure 3.6: Diagram of two common deep learning architectures for diagnosis in predictive maintenance: deep belief network with feed-forward predictor on left and self organizing map on right.

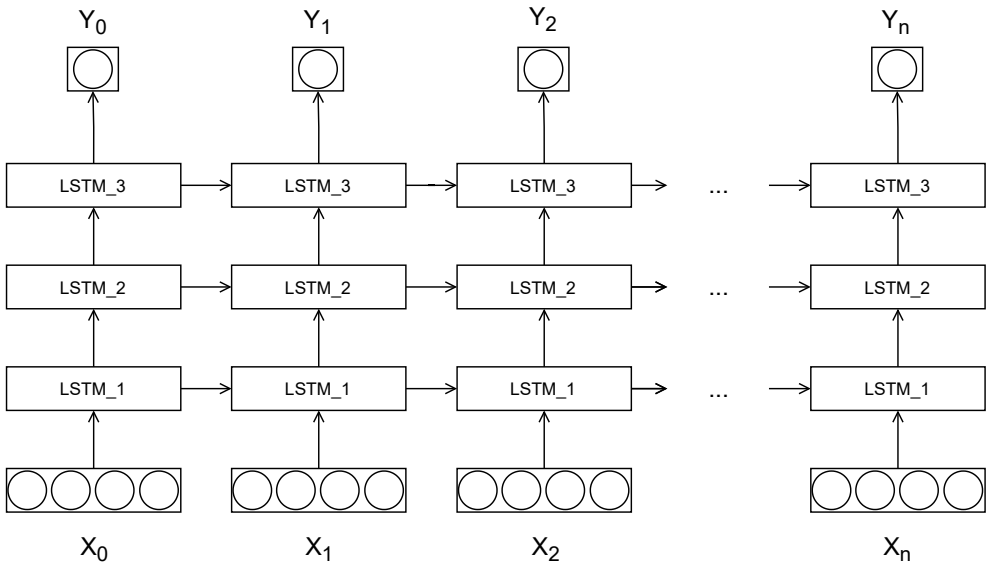


Figure 3.7: Diagram of a common deep learning architecture for predictive maintenance prognosis, based on LSTM layers.

The rest of this subsection summarises the contributions and strengths of relevant analysed works. One interesting article was published by Shao et al. [248], where a method-

ology of AE optimisation for rotating machinery fault diagnosis is presented. Firstly, they create a new loss function based on maximum correntropy to enhance feature learning. Secondly, they optimise model's key parameters to adapt it to signal features. This model is applied to fault diagnosis of gearbox and roller bearing. Another relevant publication is by Lu et al. [249] which uses growing SOM, a extension of SOM algorithm that does not need specification of map dimension. It has been applied to simulated test cases with application in PdM.

Guo et al. [250] propose a model based on LSTM and EWMA control chart for change point detection that is suitable for online training. An additional interesting work is presented by Lejon et al. [251], who use ML techniques to detect anomalies in hot stamping machine by non-ML experts. They aim to detect anomalous strokes, where the machine is not working properly. They present the problem that most of the collected data corresponds to press strokes of products without defects and that all the data is unlabelled. This data comes from sensors that measure pressures, positions and temperature. The algorithms they benchmarked are AE, OC-SVM and IF, where AE outperforms the rest achieving the least number of false positive instances. As the authors conclude, *the obtained results show the potential of ML in this field in transient and non-stationary signals when fault characteristics are unknown*, adding that AEs fulfill the requirements of low implementation cost and close to real-time operation that will lead to more informed and effective decisions.

As previously mentioned in this section, the possibility of model combination is infinite. For instance the publication by Li et al. [12] with scheme in Figure 3.8, combines a GAN structure with LSTM neurons, two widely used DL techniques that achieve results. Additionally, DL techniques can be combined with other computing techniques as Unal et al. [252], combining a feed forward network with Genetic Algorithms.

The last highlighted article that combines DL models is by Zhang et al. [13], one of the most complete unsupervised PdM works, contained in Figure 3.9. They build a model that uses correlation of sensor signals in the form of signature matrices as input that is fed into an AE that uses CNN and LSTM with attention for AD, partial RCA and RUL. The strengths of this work are the following: they show that correlation is a good descriptor for time-series signals, attention mechanism using LSTMs gives temporal context and the use of anomaly score as HI is useful for RCA, mapping the detected failures to the input sensors that originate them. Conversely, the RCA they do is not

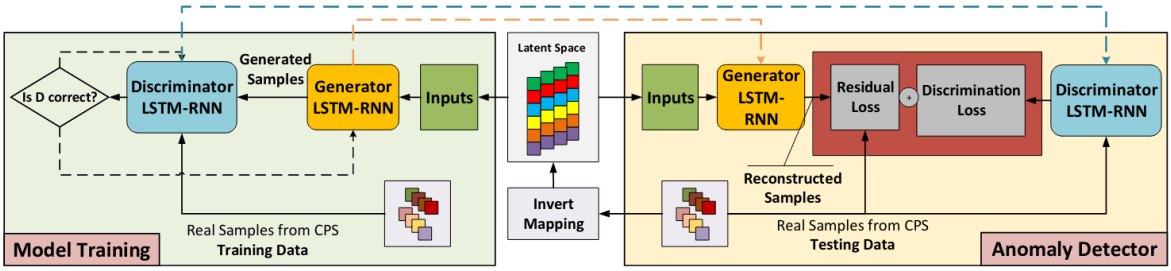


Figure 3.8: Architecture of anomaly detection PdM using GAN and LSTM proposed by Li et al. [12].

complete since they only correlate failures to input sensors but are not able to link them to physical meaning. Moreover, the lack of pooling layers together with the combination of DL techniques results in a complex model that is computationally expensive, needs more time and data for training and its decisions are hard to explain.

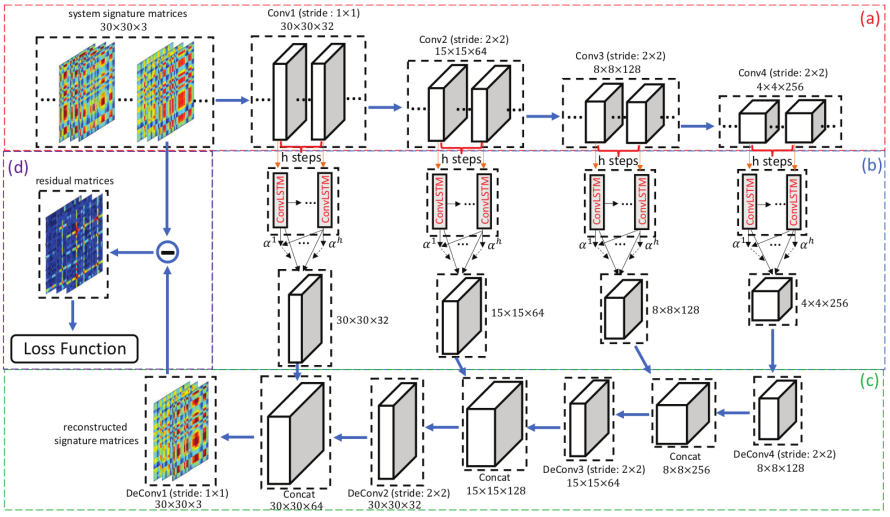


Figure 3.9: Architecture of semi-supervised autoencoder by Zhang et al. [13].

The following publications use other ML tasks combined with DL models for PdM, and other DL techniques. Wen et al. [253] use transfer learning with a SAE for motor vibration AD, outperforming DBNs. The article by Wen et al. [254] proposes a transfer learning based framework inspired in U-Net that is pretrained with univariate time-series synthetic data. The aim of this network is to be adaptable to other univariate or multivariate anomaly detection problems by fine-tuning.

Martinez-Arellano et al. [255] present a bayesian and CNN based DL classifier for AD.

They first use a small labelled dataset to train the model. Then, the model is used to classify the remaining data and then, it uses uncertainty modelling to analyse the observations that cannot be correctly classified due to high entropy. Finally, it selects the top 100 with highest entropy to query an domain knowledge technician, asking him/her to label them in order to retrain the model with this new data. This procedure is followed until the model obtains a good accuracy. This work is an example of how to use two interesting techniques in the field of PdM to address the problem of lacking labelled data by querying domain technicians, showing them the instances from which the model can learn the most. Concretely, the aforementioned techniques belong to semi-supervised classification type using active learning. Similarly, the review by Khan et al. [16] mentions that expert knowledge can help troubleshooting the model and, if domain technicians are available, the model could learn from them using a ML training technique called active learning where the model queries them in the learning stage. Moreover, the work by Kateris et al. [256] uses SOM as OCC model for AD together with active learning, to progressively learn different stages of faults.

The architectures of stacked autoencoders and stacked restricted boltzmann machines stated above are commonly used to optimise the creation of more complex deep learning architectures by stacking one simple architecture type multiple times. However, there is little research applied to ensemble learning that combines different deep learning techniques for predictive maintenance, or even with other data-driven systems. The article by Li et al. [257] trains base algorithms separately and then uses a parallel ensemble method that weights the prediction of each base algorithm based on their performance in order to produce the output of the ensemble algorithm for aircraft data; the weight vectors are optimized using particle swarm optimization and sequential quadratic optimization algorithms. Similarly, the article by Li et al. [14] in Figure 3.10 presents a method that weights predictions of different remaining useful life algorithms; it could be used to combine different deep learning models with themselves or other data-driven models. The work presented by Bose et al. [235] uses an ensemble-based voting system to create a one-class classifier relying on ELMs that optimises consumption and speeds up calculations; this enables its installation in edge computing given the achieved neuron quantity reduction.

Additionally, there are methods to fuse deep learning architectures as Shao et al. propose in [258], where autoencoders are stacked based on majority voting, selective ensemble

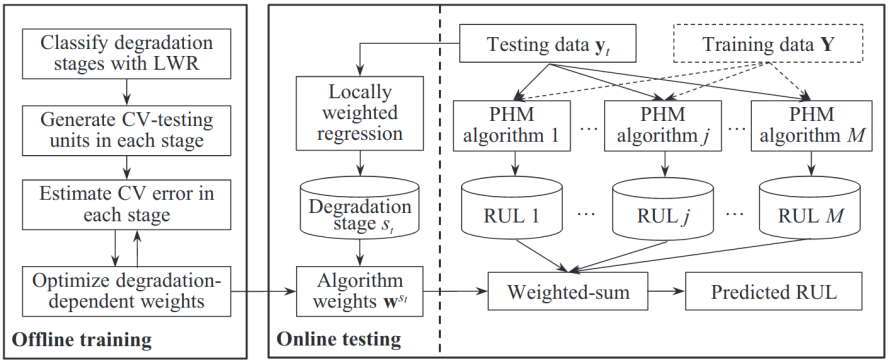


Figure 3.10: Architecture to ensemble prognosis algorithms proposed by Li et al. [14].

and weight assignment techniques for roller bearing diagnosis. Likewise, a stacked ensemble of recurrent neural networks for remaining useful life estimation is presented by Mashhadi et al. [259]. All in all, ensemble techniques have shown promising results in the field of predictive maintenance. However, the combination of algorithms in a meta-model increases the complexity and therefore difficult explainability, so the choice of implementing ensemble methods or not is tied to the objectives of each use-case.

Another interesting technique with PdM applications is deep reinforcement learning. The publication by Zhang et al. [260] uses it for HI learning, outperforming feed-forward networks but underperforming CNN and LSTM for AD and RUL. This technique consists of transferring the knowledge adquired from one dataset to another one. The procedure consists of reusing a part or the whole pretrained model adapting it to new's requirements, which sometimes requires retraining the model but this needs less data and time. In addition, Koprinkova-Hristova et al. [261] use reinforcement learning on echo state networks to predict possible alarm situations in an industrial power plant, enabling model learning by experience, online readaptation from new information and human expert advice accounting.

3.1.3.7 Related review works summary

This subsection summarises the most relevant information of the review works related to DL-based PdM, highlighting their main contributions, detected challenges and gaps in the works and their conclusions. In addition, Table 3.7 compares the contributions state-of-the-art reviews and surveys about deep learning-based PdM application, analysing their applicability on PdM stages and adaptability to relevant industrial requirements.

Moreover, their description and limitations are presented and compared with the contributions of this section.

Table 3.7: Summary of related review works regarding DL application for PdM and comparisons with this section. The columns evaluate whether the works conduct a review of the corresponding characteristics.

		PdM stages				Industrial requirements							
	Compare PdM results	Anomaly detection	Diagnosis	Prognosis	Mitigation	Semi-supervised	Data variability	Adaptability	Transfer learning	Ensemble learning	Reinforcement learning	Uncertainty modelling	Description and limitations
Zhao et al. [262]	✓	✓	✓	✓	✗	✓	✓	✓	✗	✗	✗	✗	Covers the main models: AE, RBM, DBN, CNN, RNN, but does not cover generative models. The results are compared quantitatively in a local dataset. Several techniques required to address industrial requirements are not covered.
Zhang et al. [89]	✓	✓	✓	✓	✗	✓	✗	✓	✗	✗	✗	✗	Only feed-forward and AE models are included. Their accuracy in different public datasets is presented. Several techniques required to address industrial requirements are not covered.
Khan et al. [16]	✗	✓	✓	✓	✗	✓	✓	✓	✗	✗	✗	✓	Covers RBM, DBN, CNN, RNN, but does not cover generative models. It covers a few techniques required to address industrial requirements, but several are missing. It does not compare PdM results.
Fink et al. [263]	✗	✗	✗	✗	✗	✓	✗	✗	✓	✗	✓	✓	Reviews the main DL architectures including generative ones. It reviews the principal works, focusing on challenges. It does not compare architectures nor how they are applied to solve PdM stages. It includes a few techniques required to address industrial requirements, but several are missing.
This section	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	Reviews the principal DL works by category, including one-class neural networks, SOM and generative models. Compares and discusses the results in a public dataset quantitatively. It also compares models qualitatively, which facilitates architecture fusion. Moreover, ensemble learning is reviewed to enable robust PdM models. The PdM mitigation step is presented, supported on domain technicians. It includes several techniques required to address industrial requirements that complement existing works.

The work by [264] conducts a qualitative narrative review on the SotA fast DL models applied for PdM in industrial IoT environments. They argue that real-time processing is essential for IoT applications, meaning that a high-latency system can lead to unintentional reactive maintenance due to insufficient maintenance planning time. Moreover, they highlight how DL models can be optimised. They state that weight sharing on RNNs enables parallel learning, which can help in training these types of networks that achieve SotA results in most PdM applications. Accordingly, they also justify the use of max-pooling layers when dealing with CNNs to eliminate redundant processing and thus optimise them.

Two DL reviews applied to other fields contain information about models that could be used for PdM: DL models for time series classification by [265] and DL used to model sensor data by [266]. However, these works do not focus on PdM, and therefore, their design, development and validation do not address predictive maintenance use case requirements.

The review by [262] explains there are algorithms that use traditional and hand-crafted features whereas others use DL features for the problem, and presents the most common FE methods for DL based PdM systems. They state that both aforementioned features work properly in DL models, supported on their state-of-the-art revision. These works usually use techniques to boost model performance as data-augmentation, model design and optimisation for the problem, adopting architectures that already work in the state-of-the-art. They also adapt the learning function and apply regularisations and tweak the number of neurons, connections, apply transfer learning or stack models in order to enhance model generalisation and prevent overfitting. The advantage of traditional and hand-crafted features is they are not problem specific, being applicable to other problems. Moreover, they are easy to understand by expert-knowledge technicians given that they are based on mathematical equations. However, as they are not problem specific, in some cases DL-based FE techniques perform better since these are learned specifically for the problem and directly from the data. However, they are not as intuitive as aforementioned features, meaning that technicians can have problems trying to understand how they work.

The article by Zhao et al. [262] also summarises the information already stated throughout this section: DL models can achieve state-of-the-art results, pre-training in AEs can boost their performance, denoising models are beneficial for PdM because of the nature

of sensor data and that CNN and LSTM variants can achieve state-of-the-art results in the field of PdM using model-optimisation, depending on the dataset's scale. In addition, domain knowledge can help in FE and model optimisation. Conversely, it is difficult to understand DL models even if there are some visualisation techniques because they are black-box models. Transfer learning could be used when having little training data, and PdM belongs to a imbalaced class problem because faulty data is scarce or missing.

The survey by Zhang et al.[89] compares the accuracy obtained by ANN, Deep ANN and AE in different datasets, which allows comparisons, however these comparisons are done with models applied to different datasets and therefore they are not fair. Nonetheless, they show high accuracy results, most of them between 95% and 100%, emphasising that DL models can obtain promising results. They state that deeper models and higher dimensional feature vectors result in higher accuracy models but sufficient data is needed. With the increase of computational power and data growth in the field of PdM, research on this area tends to focus on data-driven techniques and specifically DL models. However, DL models lack of the explainability and interpretability of taken decisions.

The review by Khan et al. [16] states that the developed DL architectures are application or equipment specific and therefore there is no clear way to select, design or implement those architectures; the researches do not tend to justify the decision of selecting one architecture over another that also works for the problem, for instance selecting CNN versus LSTM for RUL. Its authors also argue that algorithms as the ones presented throughout this section all have shown to be working correctly and are not different from one another. In addition, the work by Fink et al. [263] reviews relevant PdM works and current tendencies, but they do not detail how to build DL-based models for each PdM stage.

Even if this section has been focused on DL models for PdM, these works are often integrated with traditional models and/or traditionally FE features, such as time and frequency domains, feature extraction based on expert knowledge or mathematical equations.

As the authors Khan et al. state [16], there is a lack of understanding of a problem when building DL models. They also argue that VAE is ideal for modelling complex systems, achieving high prediction accuracy without health status information. The

algorithms that analyse the data maintaining its time-series relationship by analysing the variables together, at the same time, are the most successful: no matter if using sliding window, CNN or LSTM techniques. Most of algorithms focus on AD, whereas they can also be adapted to perform RUL by a regression or RNN, where the majority use LSTMs. Regressions commonly use features learned for the used AD models, or even use traditional and hand-crafted features. Generative models like GAN do not work as good as expected. However, CNN works well while needing less data and computing effort. This means that even DL models can achieve similar accuracy using traditional features or deep features extracted from the data unsupervisedly.

Finally, the reasons for existing few real application publications are summarised. Industrial companies avoid publishing their data or implementation details to protect their intellectual property and know-how from competence. Moreover, many data-driven research publications lack of domain technician feedback so they tackle the problem only relying on data-driven techniques, without embracing domain knowledge.

3.1.4 PdM datasets and state-of-the-art results

This section introduces several publicly available datasets for PdM application, and then compares and discusses the results of different data-driven models on a selected PdM dataset.

3.1.4.1 Benchmark PdM datasets

The review made by Khan et al. [16] states that one of the problems of PdM proposals is the lack of benchmarks that difficult their comparison. There are several public PdM datasets for prognosis released by the Nasa in the repository [267] belonging to the scope of predictive maintenance, which are described in the following paragraphs.

Milling dataset [267] gathers acoustic emission, vibration and current sensor data under different operating conditions with the purpose of analysing the milling wear. Regarding PdM stages, it allows the application of AD, RCA and RUL.

Bearing dataset [267] gathers vibration data from 4 accelerometers that monitor bearings under constant pressure until failure, obtaining a run-to-failure dataset where all failures occur after exceeding their design life of 100 million revolutions. Its possible PdM applications are AD and RUL estimation.

Turbofan engine degradation simulation dataset [267] contains run-to-failure data from engine sensors. Each instance starts at a random point of engine life where it works correctly, and monitors its evolution until an anomaly happens and afterwards reaches the failure state. The engines are working under different operational conditions and develop different failure modes. Its possible PdM applications are AD, RCA and RUL.

Femto bearing dataset [267] is a bearing monitoring dataset inside the Pronostia competition that contains run-to-failure and sudden failure data. The used sensors are thermocouples gathering temperature data and accelerometers that monitor vibrations in the horizontal and vertical axis. Its possible PdM applications are AD, RCA and RUL.

Industrial companies are reluctant to publish their own datasets because they tend to trade secret their data and knowledge in order to protect themselves from their competence. The dataset that approximates most to companies data is the one published by Semeion research center named *Steel plates faults dataset* [268], where steel plate faults are classified into 7 categories.

3.1.4.2 Data-driven PdM results comparison

During the elaboration of this part, the majority of reviewed works aimed at anomaly detection and diagnosis use private datasets, therefore there is no opportunity to compare or replicate their results. However, the prognosis stage is widely researched with the public dataset NASA turbofan, which has been used as reference to compare model performance.

This subsection compares different relevant data-driven works for PdM application on turbofan dataset introduced in previous subsection, which is generated using the *Commercial modular aero-propulsion system simulation*. The reasons for choosing this dataset are that it is one of the reference datasets of PdM, it enables the application of all PdM steps and it is one of the most used dataset for model ranking; despite the majority of works focus on prognosis.

This challenge is divided into four datasets, each of them with different characteristics. The first, FD001 dataset, contains 100 train and 100 test trajectories with one operational condition and unique fault mode; the FD002 dataset contains 260 train and 259 test trajectories regarding to six operational conditions and unique fault mode;

FD003 contains 100 train and 100 test trajectories with one operational condition and two different fault modes; FD004 contains 248 train and 249 test trajectories with six operational conditions and two different fault modes. All datasets contain 3 operational setting variables and 26 sensors.

The dataset lacks of the RUL label, which is the target column. Hence, it is commonly assumed to be constant in the initial period of time where the system works in correct conditions and degrades linearly after exceeding the changepoint or initial anomalous point. The constant value in initial period is a parameter denominated as R_{max}, which is set to values near 130 for many state-of-the-art works (see Table 3.8), enabling a fair comparison of their results.

The most common metrics to evaluate model performance in this dataset are the following ones [175]: RMSE is the square root of the sum of all the squared errors between real and predicted values divided by the number of predictions, which penalises outliers more than the mean absolute error, and defined in Equation 3.3; and a score function used for this problem in PHM 2008 data challenge [269] and defined in Equation 3.4, which is asymmetric and penalises more latter error predictions than earlier ones, growing exponentially when distancing from target value, and early predictions have lower exponent value than latter ones to penalise late predictions. In previous equations, N is the number of engines in test set, S is the computed *score*, and $h = (EstimatedRUL - TrueRUL)$.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} h_i^2} \tag{3.3}$$

$$S = \begin{cases} \sum_{i=1}^{N} \left(e^{-\frac{h_i}{13}} - 1\right) & for \quad h_i < 0 \\ \sum_{i=1}^{N} \left(e^{\frac{h_i}{10}} - 1\right) & for \quad h_i \geq 0 \end{cases} \tag{3.4}$$

Table 3.8 gathers the state-of-the-art results of data-driven models from 2014 on the four data subsets that use the presented two equations for model evaluation. As explained by Ramasso et al. [270], few works previous to 2014 used the subsets' testing for model evaluation, and many used different performance metrics, what complicates comparison. Therefore, these works have been omitted, focusing only on novel works that overcame

results of what was already published on at least one of the four data subsets.

Table 3.8: State-of-the-art results on four turbofan data subsets since 2014. The lower the metric, the better the model is considered to perform on average. The best results are highlighted in bold.

Reference	R_{max}	Architecture	FD001 RMSE	FD002 RMSE	FD003 RMSE	FD004 RMSE	FD001 Score	FD002 Score	FD003 Score	FD004 Score
Ramasso et al. [270]	135	RUL-CLIPPER	13.3	22.9	16.0	24.3	**216**	2796	317	3132
Babu et al. [175]	130	FFNN	37.6	80.0	37.4	77.4	1.7e+4	7.8e+6	1.7e+4	5.6e+6
		SVR	21.0	42.0	21.0	45.3	1381	5.8e+5	1598	3.7e+5
		RVR	23.8	31.3	22.4	34.3	1504	1.7e+4	1431	2.6e+4
		DCNN	18.4	30.3	19.8	29.2	1287	1.3e+4	1596	7886
Zhang et al. [271]	130	MODBNE	15.0	25.1	12.5	28.7	334	5585	422	6558
Zheng et al. [272]	130	LSTM + FFNN	16.1	24.5	16.2	28.2	338	4450	852	5550
Li et al. [198]	125	CNN + FFNN	**12.6**	22.4	12.6	23.3	273	10412	284	1.2e+4
Ellefsen et al. [273]	115-135	RBM + LSTM	**12.6**	22.7	**12.1**	22.7	231	3366	251	**2840**
Kakati et al. [274]	125	LSTM + attention	14.0	**17.7**	12.7	**20.2**	320	**2102**	**223**	3100

The results comparison of Table 3.8 does not show only model's performance, but also the combination of preprocessing and feature engineering techniques. Therefore, results show the performance of the complete data process applied to the dataset until prediction. Nonetheless, the table shows that deep learning based architectures are the ones that achieve state-of-the-art results in recent years. Concretely, these architectures are composed of combination of different DL techniques.

The subset FD001 obtains lower errors given it only contains one operational condition and one failure type, and subset FD003 obtains similar results while containing two failure types. In contrast, performance on subsets FD002 and FD004 is significantly worse given operational conditions change at each cycle during the same experiment. Therefore, it is normal to have a significantly lower errors in subsets FD001 and FD003 in comparison with subsets FD002 and FD004 for all algorithms.

3.2 Predictive maintenance methodologies

There exist a great number of architectures for the application of PdM systems. Nonetheless, most of them generally converge into the following *reference architecture*, addressed in incremental stages: first anomaly detection, after that diagnosis of the anomaly, then prognosis of its evolution and finally its mitigation [93], as already presented in Figure 2.12.

The standardisation, specifications and guidelines on manufacturing and PdM are gathered in norms and standards. The norm EN 13306 [21] defines the terms used in maintenance and maintenance management, which enable the understanding of key PdM concepts. It also classifies maintenance strategies in a tree scheme regarding their base technique, summarises the distribution of operating time in down state and the substates of up state, and defines how to address intelligent maintenance by prioritising the most critical failures.

The OSA-CBM specification [102] is a standard information flow architecture for a CBM system contained in Figure 3.11, on which the aforementioned reference architecture is based. It proposes XML schemes to facilitate implementation and 7 layers of information flow. The international standard organization ISO 17359 [275] which, based on other standards, provides a guideline for CM and diagnosis of machines based on sensor data. It presents a procedure for implementing PdM on a schematic flowchart, divided in key steps that complement the reference architecture. Moreover, O'Donovan et al. [276] presents a set of data and system requirements for implementing equipment maintenance applications of smart manufacturing in industrial environments. It also presents an information system model that provides a scalable and fault tolerant big data pipeline for integrating, processing, and analysing industrial equipment data.

Many publications present their own variations and frameworks that complement the aforementioned reference architecture and standards. Some of the most relevant works are presented bellow. Rana et al. [277] present a guideline to help companies in the analysis of the most suitable maintenance strategy for each use-case. As they state, two relevant aspects are the analysis of data collection method and FMEA and FMECA Jordan [91]. If the predictive maintenance turns out to be most optimal, there exist other works that recommend the stages or suitable steps to facilitate its implementation, as the ones presented below.

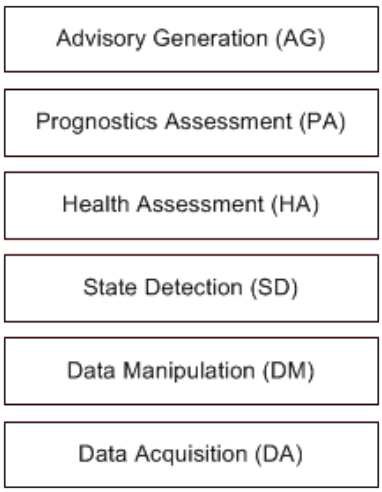


Figure 3.11: OSA-CBM functional blocks by [15].

The article published by Deloitte [278] presents a PdM journey in levels. Level 0 provides a maintenance based on reactive resolution of failures, level 1 is based on visualisation, level 2 is depends on rules created by expert-knowledge, level 3 is supported on data-driven AD, level 4 is based on prognosis models and the last, level 5, is based on the identification and mitigation of the anomaly using RCA.

The review by Khan et al. [16] proposes a data-driven step by step flowchart methodology to successfully apply PdM, see Figure 3.12. It is supported on OSA-CBM and reference architectures, and their proposal is aligned with most of the articles that have been reviewed in this section. Moreover, it states that the difference between traditional data-driven, statistical and traditional ML, and deep-learning methods is that the latter ones feed the model with the preprocessed data because features are extracted inside the model directly whereas the former ones also need the extraction and selection of features. Conversely, this section is focused on the capability of deep learning models application in system health management.

Moreover, Nuñez and Borsato [279] presents an ontological model to guide the implementation of expert systems for prognosis and health management. They demonstrate its applicability by implementing an expert system that models the RUL of a mechanical machine before entering into a functional failure.

Furthermore, the publication by Bousdekis et al. [280] proposes how diagnosis, prognosis and decision support is influenced by company management and the relation of these

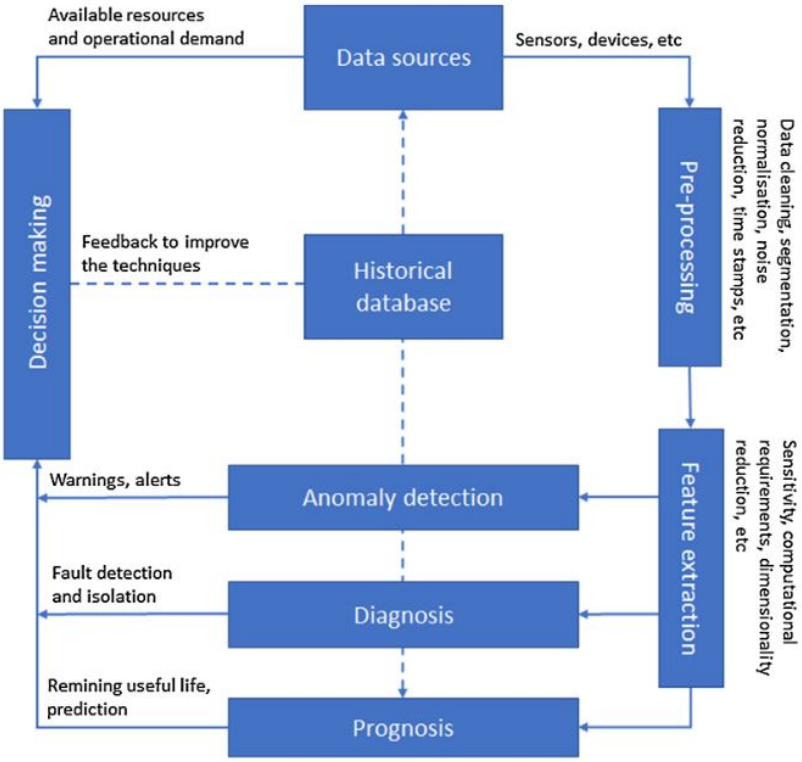


Figure 3.12: PdM flowchart proposed by Khan et al. [16]

.

tasks with maintenance management. It also explains how each PdM stage can influence or result into maintenance actions. This is the way PdM systems impact company maintenance by giving recommendations and prospect.

In addition, there are other methodologies designed to handle general machine learning life-cycle that can also be used for PdM model development. One relevant methodology is CRISP-DM by Chapman et al. [281] contained in Figure 3.13, even though it is not specifically designed for maintenance and therefore it does not consider how to handle industrial requirements.

A research on electronic databases including Scopus, Engineering Village, Springer Link, Science Direct and the search engine google scholar was performed in the time period between 2011 and 2021, to search for articles with the terms "predictive maintenance" AND "methodology" AND ("data-driven" OR "life-cycle" OR "development"). The research reported no methodology for data-driven predictive maintenance systems application that combines data-driven models with domain knowledge adapted to industrial

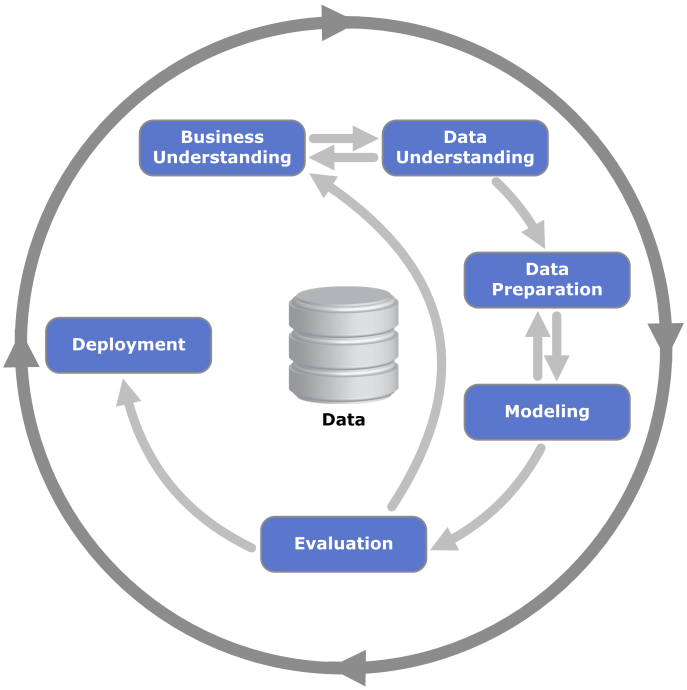


Figure 3.13: Diagram of CRISP-DM methodology by Wikipedia [17]

.

use-case requirements, specifying the steps required to manage their life-cycle.

3.3 Critical assessment of state-of-the-art

After reviewing state-of-the-art predictive maintenance works and methodologies, this section summarises current research trends and concludes highlighting research gaps.

The majority of the reviewed PdM works were created and tested in research environments but not transferred or tested in industrial companies. Even if there are some models trained with real industrial process data, the majority of the reference datasets that have been preprocessed and specifically prepared for the task are generated in simulation or testing environments. However these works do not address industrial companies' requirements presented by Venkatasubramanian et al. [97].

Industrial companies nowadays have collected much data for PdM by monitoring assets under normal working condition, but it contains little to none failure data. In this scenario, research on unsupervised, semi-supervised and one-class classification algorithms has gained relevance. Concretely, deep learning architectures like autoencoders or deep

belief networks with LSTMs or CNNs are widely researched given their high accuracy at modelling time-series data semi-supervisedly. Nonetheless, the design and optimisation of DL architectures is mainly guided by previous experience and trial and error. In addition, published unsupervised, semi-supervised and one-class classification data-driven models are unable to link novel detected failures to their physical meaning. The main reason is that these models ignore expert knowledge.

Deep learning models have gained popularity in PdM due to their high accuracy, achieving state-of-the-art results when trained with enough data. However, many works do not address industrial requirements for PdM models such as interpretability, real time execution, novelty detection or uncertainty modelling, given that mainly laboratory datasets have been used.

Nowadays there are emerging trends that may tackle mentioned gaps, such as combining explainable artificial intelligence and domain knowledge to interpret more accurate grey and black box models' behaviour; developing edge computing systems that integrate simplified architectures, reducing complexity to enable online data processing; researching unsupervised and semi-supervised architectures that enable novelty detection by modelling only correct machine data to discover failures; and enriching model output with the probability for each prediction in order to model uncertainty. However, there are few publications applying these techniques in PdM by modelling production data to address industrial requirements.

Another little researched area with promising potential is diagnosis of semi-supervised PdM systems, given the necessity to perform RCA and classify novel failures by linking to physical meaning. In addition, transfer learning may facilitate model re-usability along machines and assets, by reducing the required amount of data and training time. Research on mentioned topics is fundamental to transfer any machine learning model to real industrial use cases, and run in production.

To sum up, most industrial companies need PdM models to be accurate, easy to understand, process data on streaming, reusable and capable to detect and diagnose novel failures, adapting to process data characteristics like correct data modelling. Industrial companies are starting to implement data-driven PdM models in their use-cases, which have different characteristics and requirements. Therefore, a general methodology that defines the main steps for PdM data-driven integration could guide them in the process

and facilitate their implementation.

Existing methodological publications for data-driven PdM life-cycle management are: about CBM; focused on details of data-driven model types; roadmaps that show trends and highlight future directions; focused on technical aspects of one or two PdM steps; do not consider the importance of domain expertise; not specific for PdM and therefore not adapted to its characteristics; or are not developed to address industrial companies' requirements. For these reasons, many PdM publications end up following their own steps to address their use-cases.

The research on electronic databases reported no methodology for data-driven predictive maintenance systems application that details: their design, development and implementation, defining the required steps and resources, while specifying how to combine data-driven models with domain knowledge adapted to industrial use-case requirements.

Chapter 4

Design of a methodology to guide data-driven PdM life-cycle integrating expert knowledge

This chapter proposes a methodology that guides the life-cycle of data-driven techniques for predictive maintenance by integrating expert knowledge. This methodology addresses the first specific objective of the thesis, and its validation is performed in Chapter 5.

The schematic of the methodology is presented in Figure 4.1, containing the general stages and main steps to guide manufacturing companies in the design, development and deployment of data-driven PdM systems. It is open and modular, being flexible and adaptable to address different industrial use-case requirements iteratively while keeping simple to facilitate its implementation and understanding. Therefore, it enables the addition of new steps adapted to each use-case requirements while permitting the omission of the steps without asterisk, which are advisable yet not indispensable.

Moreover, this methodology presents the tasks required to complete the steps, the required worker profiles to succeed in its application, and specifies which deliverables are generated by the end of each stage in Figure 4.1. Concretely, three main working profiles collaborate in the implementation of this methodology: business profile contributes with business vision and leads problem definition; domain technician contributes with domain expertise supervising the project and collaborating in tasks; and data-scientist guides the project and handles the tasks related to data-driven model development, deployment, and monitoring.

The complete step-by-step version of the methodology is presented in Figure 4.2. It

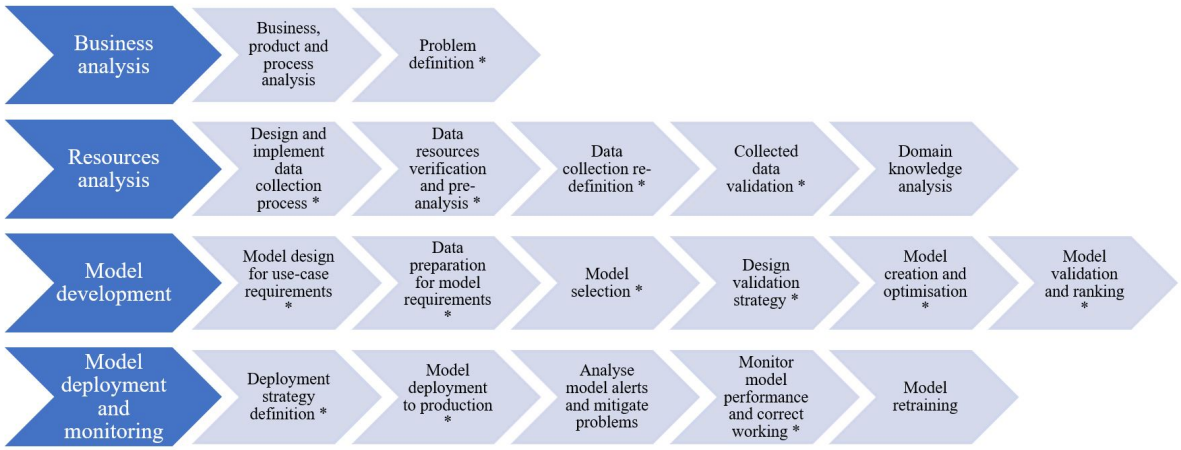


Figure 4.1: Scheme of proposed data-driven PdM methodology. Consists of four stages and their steps.

summarises methodology's main steps, containing a complete visual scheme with all the tasks, principal subtasks and relations that form steps of the methodology. Being an open and modular methodology, its implementation is flexible and does not require the adoption of all its steps, only the ones marked by an asterisk in the diagram. Moreover, it contains the principal profiles involved in each stage and the deliverables like documents and models produced at them.

The methodology is inspired by CRISP-DM data analysis methodology as described by Chapman et al. [281], PdM standards and methodologies like the ones presented in the article by Reliasoft Corporation [282], and the publications presented in Section 3.2. It contains four main sequential groups, denominated stages: business analysis, resources analysis, model development and model deployment and monitoring. These stages are explained and described in sections.

Methodology's iterative design facilitates and fastens models going into production, and then promotes incremental iterations to enhance their performance.

4.1 Business analysis

Business analysis is the first stage of the methodology, which is composed by two steps.

- *Business, product and process analysis.*

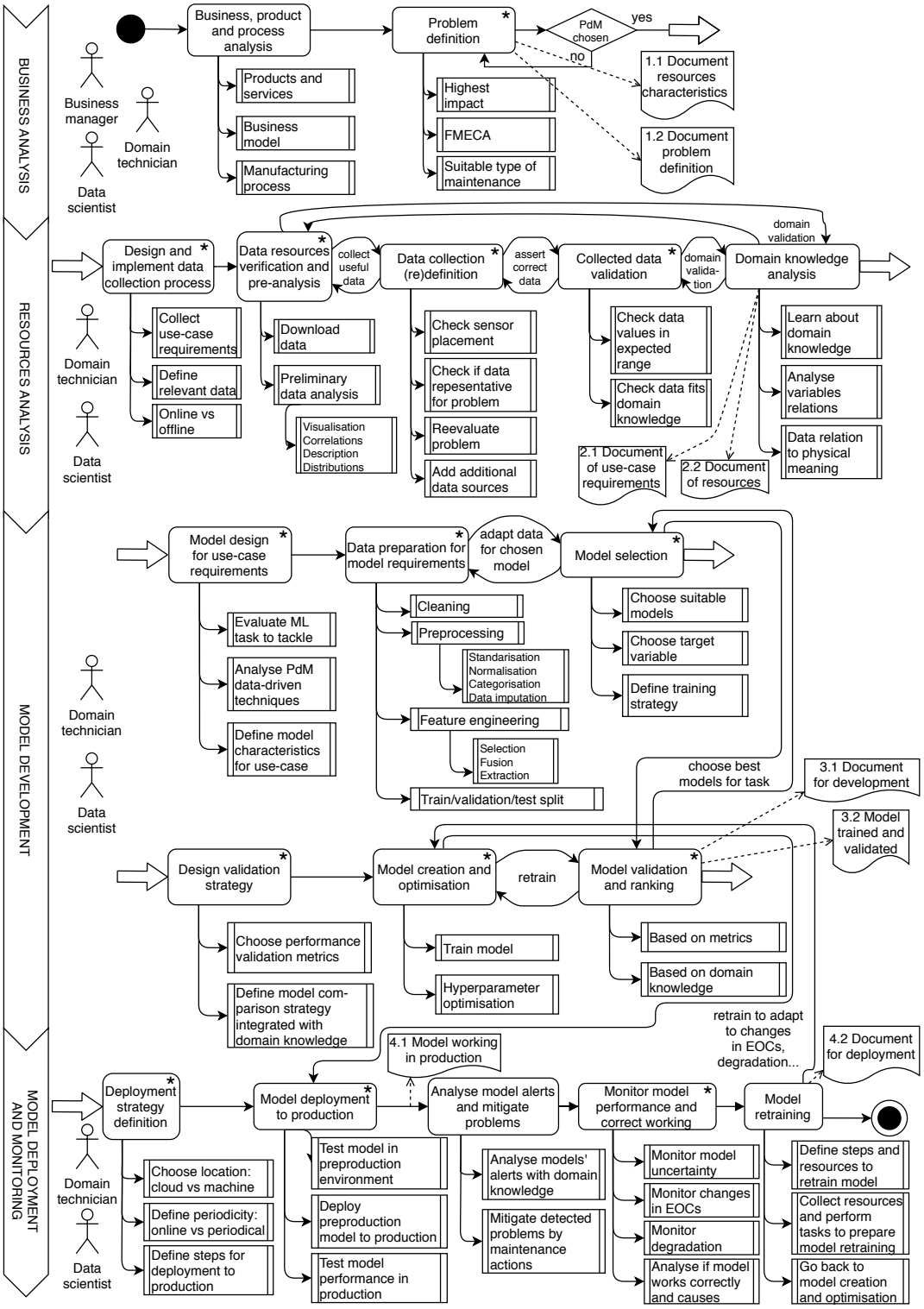


Figure 4.2: Detailed version of proposed data-driven PdM methodology, specifying its stages, steps and tasks in a flow diagram. It also contains the required profiles and indicates deliverables created at each stage.

The first step is to perform *business, product and process analysis* adopting a business vision by understanding the products and services the company offers, its business model and how they are related to the manufacturing process. It enables to prioritise among problems that address business requirements, such as production parameters optimisation, quality control and choosing the suitable maintenance strategy.

- *Problem definition.*

  If previous step's analysis results in prioritising maintenance optimisation, the assets with highest impact have to be evaluated, starting with the most common and critical failure types identified with tools like FMEA [90] and FMECA [91]. After that, the most suitable maintenance strategy for each asset has to be evaluated: whether predictive maintenance, periodical maintenance or corrective maintenance; thus completing problem definition. A good resource for this analysis is the guideline by Rana, Kumar and Srivdya [277]. Finally, if predictive maintenance is the maintenance strategy to be implemented, then the methodology advances to the second stage; which focuses on analysing use-case resources.

Business analysis is the only stage where the three required profiles for the methodology work together for its completion: the business manager provides business perspective and helps to define business requirements; the domain technician provides technical and operating expertise; and the data scientist helps to guide this stage while learning background from other profiles.

As a result of completing this stage, two documents can be created: Document 1.1 resources characteristics contains a summary of how business works, business model, products and services, and how manufacturing process works; and Document 1.2 problem definition, which explains manufacturing critical assets ordered by impact, which are maintenance requirements for these assets and components, and analysis of maintenance strategy suitability for them.

4.2 Resources analysis

After understanding the business and defining use-cases, the second stage is *resources analysis* for the problem.

- *Design and implement data collection process.*

  The first step is to *design and implement data collection process.* The data should be collected initially under similar EOC that want to be modelled from the same machines and assets through time. Comparisons among different machines/assets of the same type, even if being built under the same specifications, is difficult given differences in EOC, tolerances, adjustments, etc. EOC information is essential to give data-driven algorithms working context of the monitored assets, like environmental perturbations or working conditions that can affect its performance and result in anomalies or component degradation. Whereas monitored EOC can boost model accuracy, there exist other relevant variables that are not controlled or even monitored like environmental noise and disturbance, lack of sensors or their misplacement. Missing information like this that influences the monitored process adds uncertainty, and a result, the created PdM system will be less accurate.

  In addition, sampling rate of variables is of utter importance. According to the Nyquist-Shannon sampling theorem, a signal of unknown frequency locations has to be sampled at least at 2 times its frequency in order to enable signal reconstruction, thus maintaining enough information to avoid non reversible information loss by the aliasing effect, presented by Mishali and Eldar [283]. Anyway, collecting more data than needed is preferable to collecting less than the stated, given in oversampled data downscaling is possible, but under-sampled data cannot reconstruct original data correctly. However, big data collection and storage results into higher costs, so the collection strategy should be correctly designed to fit use-cases requirements to reduce costs and computational time. The use of signal processing techniques is encouraged to design a suitable data collection strategy that addresses the use-case's PdM characteristics. Signal processing techniques can help to determine a suitable sampling frequency. Moreover, signal processing techniques include filters such as IIR Filters, Chebysev, Butterworth or Bessel as stated by Almaged and Hale [284], which can be used to reduce the bandwidth of a signal that has a higher sampling frequency than required. When the sampling rate of the variables is different, in order to enable data analysis in any timestep for all available variables, timestep by imputation such as repeating last value or interpolation can be useful.

- *Data resources verification and pre-analysis.*

The second step is *data resources verification and pre-analysis*. It must be retrieved from the storing device, usually the online platform or hard disk on Programmable Logic Controler (PLC). After that, a preliminary data analysis has to be performed using tools such as time-series or bi-variate plot visualisation, correlation analysis, feature description and distribution plots. It is extremely important to take into consideration that correlation does not mean causality and therefore, avoid believing that two or more variables are related whilst this relation is caused by external factors. For instance, these could be collected under same circumstances or were only modified at the same time by coincidence. This problem can be avoided with the integration of theoretical and domain expertise to validate aforementioned relations.

- *Data collection re-definition.*

  The third step is *data collection re-definition*, where the integration on similar machines in aspects such as sensor type, models or placement is standardised. Moreover, the problem has to be re-evaluated considering the gathered data to check its suitability. This stage may reveal that collected data is not adequate for the defined use-case requirements and company characteristics, so iteration between current and previous steps is necessary until these are addressed; performing tasks like adding data sources or even revaluating the problem. Despite the initial machine monitoring process may not collect the most representative data for the designed task, this step helps to identify its gaps for further improvement. In addition, this methodology enables model contribution maximisation whilst minimising development effort; thus facilitating improvement by iterating steps.

- *Collected data validation.*

  The fourth step is *collected data validation* from data sources in general, and sensors in particular. The procedure consists of asserting it is in the expected range given sensor placement, sampling frequency, sensor type and related aspects. If any deviation is detected, iterations between current and previous steps are necessary to fix it. In addition, this step must be validated from domain technician perspective, thus iterating between current stage and following one to enable validation based on domain knowledge.

- *Domain knowledge analysis.*

  The fifth and last step of resources analysis stage is *domain knowledge analysis* to gain insight about the use-case, its variable types, their physical meaning, how they are collected and the relation among them, both theoretically and in real process. This knowledge is essential for many steps such as data analysis, data preparation, model selection or ranking, and facilitates the creation of simpler, more accurate and easier to understand models. Nonetheless, even if domain knowledge is a key resource to understand data, there are usually differences between theoretical knowledge and collected data given the physical process is affected by many factors of collection procedure and component variability. For this reason, it is important to analyse if data behaves as theoretically expected and if does not, being able to reason why. This can help discover problems in the data acquisition process.

Data scientist and domain technician profiles have to work together in this stage in order to define use-case requirements and design the data collection process. Moreover, the data scientist will learn domain knowledge to understand better relations among variables and therefore facilitate data analysis and interpretation, combining domain knowledge with data-driven techniques. Domain technicians will also help to validate collected data and refine it until the obtained data is representative for the use-case.

By the end of this stage, two deliverable documents can be generated that may help in the analysis of resources suitability for the use-case and detect gaps, thus guiding the implementation of the methodology. Use-case requirements are collected in Document 2.1, whereas Document 2.2 contains information about resources to address them like: description of available data and how data collection process works; information of data pre-analysis with documentation of data and its characteristics, data visualisation plots, sensor placement, or normal working range of variables; and data relation to physical meaning and domain knowledge.

These documents can facilitate the understanding of use-case requirements linked to its objectives, physical process, and data. Moreover, presentations composed by visual plots and concise text descriptions are interesting to convey the results of data analysis and resulting models to domain technicians and thus facilitate the communication to acquire knowledge. Therefore, domain knowledge can be used to complement data and resources analysis.

4.3 Model development

This is the stage where the data-driven model of PdM system is designed, data is prepared according to its requirements, it is created, and validated, obtaining as a result a version ready for deployment. It assumes that, after performing a preliminary data analysis and validation with domain expertise, the selected data subset contains predictor variables that are related to target variable. Moreover, relationship among variables is unknown given the complexity of physical systems.

Therefore, the model is created under the basis that predictor variables have the power to predict target variables using an unknown function that the created model aims to represent in this stage. Several examples of target variables for PdM are: anomaly detection, diagnosis by RCA, health index calculation and prognosis, and RUL. Another assumption is that the observed data is big enough and of sufficient quality to represent those relations that can be generalised beyond the training process.

This stage has a more in-depth technical background than the rest to facilitate its implementation, given a high number of questions and difficulties arise when dealing with tasks related to data adaptation for the model, model selection and validation. To facilitate the understanding of this section, it is divided in the following subsections: *Model development flow* explains the flow of this stage's steps, while *Data preparation extension* and *Model selection extension* complement the information of corresponding steps with technical information. The reader not interested in technical details can read *Model development flow* and then move on to *Model deployment and monitoring* stage in Section 4.4, to skip these two technical subsections.

4.3.1 Model development flow

- *Model design for use-case requirements.*

  The first step of this stage is *model design for use-case requirements.* The addressed machine learning task must be chosen to accordingly, defining model requirements that could better fit that task with current data characteristics, such as classification, regression, clustering, one-class classification, etc. Then, state-of-the-art research on data-driven predictive maintenance models should be carried out to find commonly used data-driven architectures and models that could fit the

problem and selected task, supported on articles like [121, 285].

- *Data preparation for model requirements.*

  The second step is *data preparation for model requirements*, based on these processes: cleaning, preprocessing, feature engineering and split into train and test datasets. Commonly used data preprocessing techniques are: incorrect values cleaning, encoding and discretisation, segmentation, feature scaling (including normalisation and standardisation), noise reduction (reduce random variations of sensor output that are not related to sensor input) and imbalance data handling. Feature engineering can be done either extracting hand-crafted/traditional features that are relevant for the problem, or by using algorithms like PCA or deep learning to extract features automatically from preprocessed features. The first type of features are easier to understand but require domain knowledge and are not specifically designed for the problem. In contrast, the second type of features are more difficult to understand but are trained directly from the data for the problem, so these do not require manual design of features.

  The extracted features should always be adapted to problem requirements and characteristics such as time-series, for instance extracting them in time-windows or cycles, to create features in new space where data context is easier to identify. When there is less information or data available, domain expertise and theory can help to gain additional insight or learn knowledge beyond the data. Data preparation is an essential step to achieve meaningful model results, and therefore, more in-depth information is presented in the subsection *Data preparation extension.*

- *Model selection.*

  The third step performs *model selection*, analysing state-of-the-art models and evaluating which could better fit the characteristics defined in step one of current stage. Moreover, the set of target variables must be chosen for the model and think of how these are related to the PdM stage it will perform, adapted to data characteristics like information level and additional resources like the available domain knowledge. More than one type of model can be combined to create a more robust model, thus complementing the gaps that only one model can have. Furthermore, the training strategy for the model must be defined to assert the model is robust to noise or changes in EOC by selecting the appropriate data

train/test partition strategy or using cross-validation. In order to facilitate the selection of a model that addresses the desired PdM stages, the subsection *Model selection extension* explains how to create them, and which type of models are most suitable for use-case data characteristics and requirements.

- *Design validation strategy.*

  The fourth step is the *design validation strategy* that will be used to compare and rank models in training and testing phases. It consists of choosing the most suitable metrics according to use-case characteristics, considering which are the target variables and how the model is designed to fit them. In addition to the quantitative approach the validation metrics offer, additional qualitative comparison strategies can be defined with domain technicians to integrate domain validation; these strategies assert that models also address use-case peculiarities from technical perspective.

- *Model creation and optimisation.*

  Hereafter, the *model creation and optimisation* takes place in step five, based on the defined use-case requirements, prepared data, selected model and designed validation strategy in current stage. The model is trained to map predictor variables to target variables with the objective to minimise the error of its estimations, thus learning to relate them based on data. However, it must be constrained to generalise from data beyond the training set; this way it will work with novel data belonging to the same distribution.

- *Model validation and ranking.*

  The sixth and last step of this stage is called *model validation and ranking*, where validation metrics together with domain knowledge are used to evaluate, compare and finally rank the generated models. This step enables to prioritise and validate models in a systematised way, asserting that the chosen ones are the most suitable for the data and are aligned with use-case requirements; commonly, iterating between the two last steps of this stage is necessary to achieve this suitability. The Monte Carlo simulations technique can be used validate data-driven models for PdM, as implemented by Ley and Orchard [286]. Once the model meets the desired characteristics, it is ready for deployment.

This *model development* stage is guided by data scientists, who use different data-driven techniques to clean and prepare the data to create the chosen data-driven model. Nonetheless, constant interactions with domain technicians are necessary to assert the developed data-driven model addresses use-case requirements and ensure its estimations are related to physical meaning. This facilitates diagnosis, increases trust of stakeholders in the model, and ensures the model is created in a robust way; avoiding data biased relations by validation with domain knowledge.

In this stage two deliverables are generated: Document 3.1 contains the decisions and steps performed for data preparation and model development, gathering the following aspects: definition of suitable model characteristics to address use-case requirements; data preparation steps for the model; research on state-of-the-art models and reasons for prioritising some models over others; definition of model validation strategy; and definition of how domain knowledge is integrated into models, specifying how it guides their development and validation. In addition, the deliverable 3.2 is the data-driven Model trained and validated in training data, guided by domain technicians to integrate domain knowledge.

4.3.2 Data preparation extension

The data-driven model should be chosen to address use-case and data requirements. Accordingly, its performance, interpretability, processing time and many other characteristics vary among use-cases, and are tied to their limitations and decisions undertaken during the preparation. For instance, linear models are usually faster and easier to interpret, but they have limitations when modelling non-linear data relations.

The most challenging task of PdM system development is to obtain a dataset that is representative for the problem, preprocessed, and containing only features that are relevant yet interpretable if possible. It is better to focus efforts on collecting better data when the collected one is not good enough or little for the designed task, rather than optimising a specific model to achieve a slightly higher accuracy. The reason behind this statement is that even the most complex models capable of modelling any kind of relationship are useless in a dataset that lacks of information on target variables, or when these are not useful for the previously defined business problem.

Commonly, process data contains time-series signals, which can show characteristics

like seasonality, stationarity and trend. Thus, observations of one variable are related to observations of the same in different timesteps and cycles, which can be useful to detect trends. This data is typically analysed together by taking chunks of specific size of continuous observations, technique denominated as sliding window. After analysing a specific time or cycles frame, the window advances to next chunk. Data can also be divided and loaded into smaller datasets when it does not fit into computer or server RAM, enabling to load and free memory at will. Some libraries that are specifically designed for that task facilitate this implementation. In addition, many factors influence performance of developed algorithms; the main ones are discussed below.

When modelled data belongs to different working conditions, these can be grouped by similarity to be analysed together, or even create one model for each working mode. The latter can improve accuracy while simplifying the model, but does not generate common relations among data of different EOC. This issue may be solved by using only one model instead of many and feeding it with different working condition data.

With the objective to create a model that adapts to use-case requirements, data peculiarities, and in further stages be able to interpret its predictions or trust it, it is necessary for data scientists to gain domain knowledge. However, many times they will need the assistance of expert technicians for model interpretation or optimisation on any step performed by following the predictive maintenance stages of the methodology.

4.3.3 Model selection extension

Selecting the most adequate machine learning task to solve for each use-case is not trivial. This subsection aims to facilitate the analysis and choice of ML architectures given data characteristics, and recommended ML type to solve the corresponding task presented in Figure 4.3. The ML possibilities are described in incremental levels according to the information companies have with regard to data.

The roadmap presented in this Figure can be used to select the correct model type to address target steps of PdM roadmap (Figure 2.12). Moreover, this section describes the characteristics of ML tasks for each PdM stage: anomaly detection, diagnosis, prognosis, and mitigation. Data acquisition and preprocessing are two additional stages that prepare data for PdM which are often overlooked. Despite this fact and being resource demanding, these are necessary to obtain high-quality data and as a result accurate

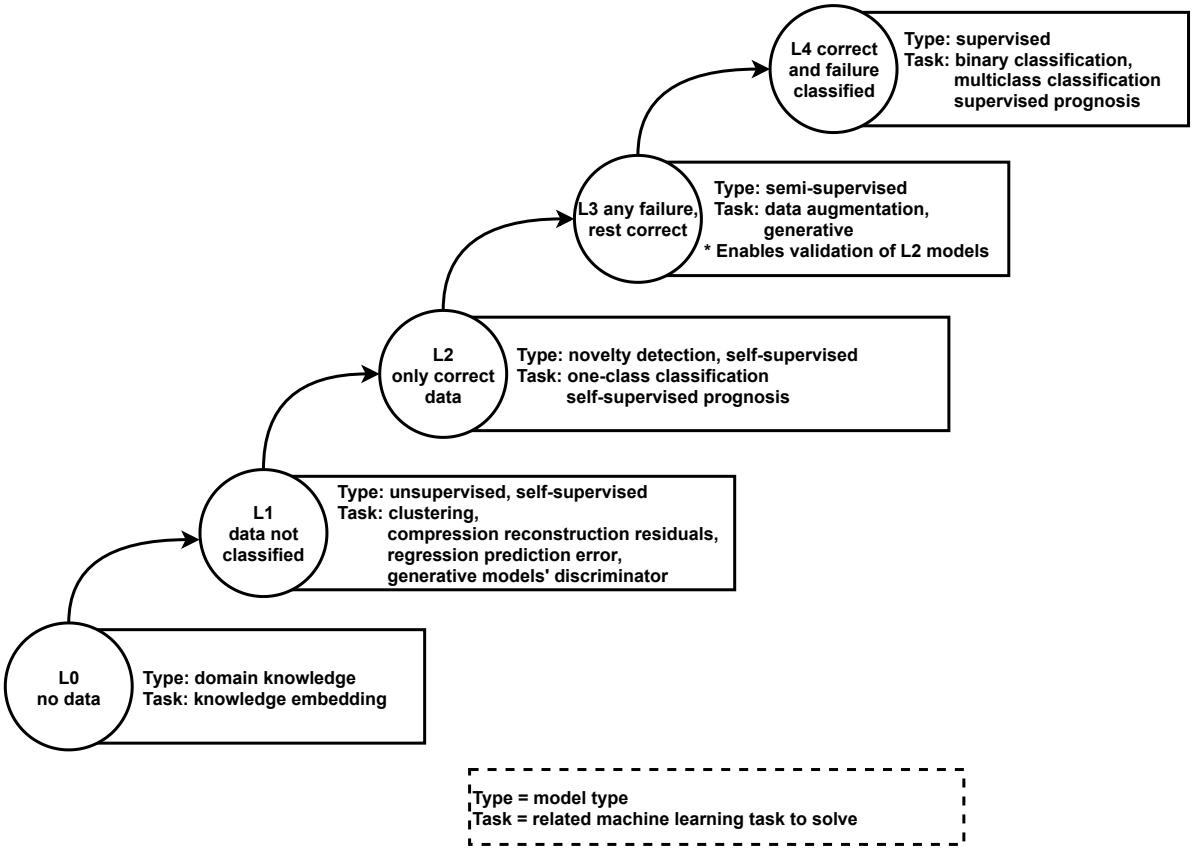


Figure 4.3: Roadmap to assess in data-driven task and machine learning model selection according to available data levels. Higher levels indicate higher information on collected data, which enable more accurate results and possibilities to address PdM roadmap of Figure 2.12.

models.

Information about the capabilities of each data level in Figure 4.3, and which PdM steps can address is explained in the following list:

- L0) the company lacks working historical data. Therefore, the only possible approach is using domain knowledge-based systems that embed prior theoretical knowledge and expertise into the system.

- L1) unclassified data can only be treated in unsupervised way by clustering data-driven systems, and self-learning by tracking reconstruction residuals using compression techniques like autoencoder, calculating prediction error of regressors, or by using discriminators of generative models.

- L2) domain technicians confirm that collected data belongs to normal working

condition, so novelty detection techniques, commonly one-class classification algorithms, are used to detect instances different or far from the already known. This level also enables self-supervised prognosis, which can be used to identify changing trends on machine condition by integrating domain expertise.

- L3) there are few failure observations classified and the rest are classified as correct data, so semi-supervised algorithms are used, data-augmentation for imbalance class data handling, and synthetic failure data generators. Monitoring few failures of one failure type may not be enough to train a classification model, and therefore more observations may be required. However, any failure data can be used to validate and calibrate models trained on level two, obtaining more accurate and robust models.

- L4) there is enough failure and non-failure data, therefore the problem is supervised. In this scenario, binary classification algorithms are used to classify failure and correct observations, and when there are different types of failures classified in training data, multi-class classification algorithms are used to differentiate them. This supervised historical data has applications on supervised machine condition prognosis to estimate how it will evolve, and therefore detect trends like degradation.

Each data level can use tasks of lower levels in addition to the ones of itself. The higher level in the roadmap, the more complete and accurate predictions the model can achieve. Machine learning tasks of data levels are used to solve predictive maintenance steps:

- *Anomaly detection* is commonly performed by classification models that classify asset condition into faulty or correct, and some can even classify different types of failures. However, these models can only be used when there are enough observations of the target failure types. In many cases, there is little or no failure data so the common strategy in these scenarios is modelling asset normal behaviour and detect anomalies when the data is different. There are three type of anomalies regarding the number of observations involved: individual observation denominated as local, global which are formed by several observations, and context which are not an anomalous by themselves but become anomalous when additional information is given. The type of anomaly to be detected should be chosen according to the use-case.

- *Diagnosis*: once an anomaly has been detected, diagnosis should be performed to analyse which components have been affected, in which way and to what extent. Some possible common factors are measurement errors, changes in EOC, component degradation while keeping correct working mode, failures and conditions that can lead to them. A useful technique to detect failure causes is root cause analysis, which is defined by Andersen et al. in the book [287] as "structured investigation that aims to identify the true cause of a problem and the actions necessary to eliminate it". It also and defines three levels of causes: symptoms, the first-level causes that lead to the problem, and the higher level causes that lead to first-level causes, where root cause is the origin.

  The used diagnosis techniques have to adapt to use-case, its data requirements and the implemented anomaly detection model. When correct and failure data or even different type of failures is available, performing a classification of failure types is straightforward. Conversely, when there is no failure data classified, this step has to be performed in unsupervised or self-supervised ways, by combining these types of machine learning models with domain expertise. These techniques can make use of health indexes, which represent the percentage of deviation the assets suffer with respect to past correct working data that could be related with damage.

- *Prognosis* in PdM is based on remaining useful life models which estimate the remaining time to failure of a component or asset according to working conditions of that moment, based on its state and the detected and diagnosed anomalies. These models can also provide a confidence bound of their estimation. Conversely, when there is no historical run-to-failure data, data-driven prognosis models can only perform health degradation monitoring and estimation by tracking health index in a semi-supervised way.

- *Mitigation*: the last step consists of providing operators with data-driven notifications and recommendations to speed up and optimise maintenance. These should be based on alarms and information gathered from previous PdM stages, contained in a simple yet effective way to understand by domain technicians.

Table 4.1 presents how the integration of domain knowledge with data-driven systems can help in each step of predictive maintenance methodology application, supported on

the scheme presented in the review by Khan et al. [16].

All in all, no algorithm is better than the rest, their suitability depend on use-case and data requirements. In addition, there can be more than one model suitable for the same use-case. It is useful to analyse their specifications, and choose based on guidelines, related reviews and state-of-the-art works like [24, 285]. Nonetheless, nowadays it is common to combine different techniques to create a more complete architecture that overcomes the gaps of containing algorithms, thus better addressing use-case challenges.

Stage	Domain knowledge	Data-driven techniques	Combined
Data adquisition and database creation	Define relevant features to monitor according to experience and theoretical background. Direct relation to physical meaning. Assert collection is correct given knowledge.	Feature relevance with respect to target feature according to data.	Most relevant features selection. Knowledge gain on process and understand it better combining data and domain knowledge.
Data preparation	Extract and select the most relevant features to monitor according to domain knowledge and relation to physical meaning. Validate and clean data.	Preprocessing: automatic machine learning based techniques to preprocess data: encoding, data cleaning, scalling, noise reduction, imbalance data handling and not available data handling. Automatic feature extraction, selection and fusion.	Select and extract the most representative features for the use-case and target variable that could be understood by domain technicians or linked to physical process. It may be automated by data-driven techniques, previously guided and afterwards validated by domain technicians.
Anomaly detection	Help create the data-driven system. Then, assert that anomaly detector works well in preproduction, production and detect changes in trends, helping to decide when to retrain. They also enrich models' output with expertise.	Automatically detect anomalies based on process and related data, comparing it with historical database and embedded knowledge.	Automatic data-driven anomaly detection and verification based on experience and theoretical background.
Diagnosis	Validate the diagnosis performed by data-driven system and complement it with expertise, theoretical background and add relevant external information, i.e. collecting additional EOC information not gathered in the data.	Diagnose anomaly type if anomaly detector is a multiclass classifier. Conversely, when the model is binary classifier, unsupervised or semi-supervised, it can outline the reasons or variable values that made it be anomaly, helping technicians perform diagnosis.	ML extracts data and models correlations to help in diagnosis, and domain technicians use context and knowledge to validate and complement algorithm predictions and gain additional knowledge.
Prognosis	Prognosticate degradation based on asset properties like materials, designed lifecycle and working experience.	Prognosticate asset degradation by tracking their health based on data, monitoring how it changes with time.	Combine data and knowledge to perform more accurate prognosis and gain knowledge.
Decision making and mitigation	Plan and coordinate maintenance actions supported on maintenance management and manufacturing operation management processes, using PdM system information to address process requirements. This enables moving towards a more optimised maintenance.	Raise alarms, notify strange working conditions and give recommendations to prevent failures. In addition, advice how to perform more optimal maintenance by comparing current condition with historical data.	Domain technicians investigate data-driven alerts, recommendations and highlighted data by comparison with previous events and knowledge to interpret, understand and validate their predictions. Based on these resources, technicians create and execute maintenance plan.

Table 4.1: Contributions of data-driven and domain knowledge in each PdM stage individually and combined.

4.4 Model deployment and monitoring

The final stage of the methodology consists of preparing the model and taking the necessary steps for its deployment to production.

- *Deployment strategy definition.*

  The first step is *deployment strategy definition* to systematise and speed up the action of putting models into production. Firstly, the most suitable location for model must be selected, choosing whether it will be executed in cloud or a PLC in the production plant or edge. Another relevant aspect is the execution periodicity, which can be offline, on streaming, or periodical after performing a certain number of cycles or working time. This strategy should also contain detailed steps to follow for model deployment to production, which will provide with a framework that facilitates model testing in production environment while avoiding mistakes.

- *Model deployment to production.*

  The second step is *model deployment to production.* For that, first the developed model must be tested in a preproduction environment that shares the characteristics of the production environment. This enables to detect incompatibilities and possible problems of the model in production without interrupting or damaging that system, facilitating the deployment of the pretested model. Once the model is running in production, it has to be tested by monitoring its performance with new production data and maybe generating synthetic failures or degradation data to check that it works correctly and raises alert messages.

- *Analyse model alerts and mitigate problems.*

  The third step of this stage is to *analyse model alerts and mitigate problems.* It aims to assert correct model performance in production, so the model has to work correctly with novel production data. In case any problem or abnormality arises in the process, it should be addressed with domain knowledge support until the model works properly according to the defined requirements.

- *Monitor model performance and correct working.*

  The fourth step is to *monitor model performance and correct working* by tracking

its evolution and adaptability to production data with techniques like prediction uncertainty, detection of changes in data such as EOC, and machine degradation that should be reflected on data. When any of these indicators suggests that the model is not working correctly with collected data, a more robust analysis should be performed before accepting this conclusion. This analysis must combine data-driven techniques and domain expertise to perform a complete evaluation from both perspectives, gaining more insight and facilitating the cause detection of incorrect working.

- *Model retraining.*

  If the previous analysis concludes that the model is not working correctly, then the fifth step of this stage and last step of the methodology must be performed: *model retraining*. This step is supported on the conclusions of the abnormal working analysis performed in the previous step, which will be used to define the tasks and resources required to retrain the model. After the resources are collected and tasks are performed to prepare the retraining, the process returns to the model creation and optimisation step of the model preparation stage, where the model will be adapted to new requirements and its development and deployment will continue from this step.

In this last stage, domain technician and data scientist profiles have to work together to: define the most suitable deployment strategy for the model in the use-case according to its requirements and resources; deploy the model and validate it combining data-driven metrics and domain knowledge; monitor its performance; and go back to retraining when it stops working correctly to adapt to new working conditions.

During this stage, two deliverables are generated: deliverable 4.1 contains the Model working in production that analyses production process data either on streaming or periodically. This model has been tested to assert it works correctly in production, and protocols to handle its alerts and retraining are collected in the deliverable 4.2. The last deliverable is 4.2 Document, which defines the deployment steps necessary to take the model generated in previous stage to production, containing: a protocol to test whether the model is working correctly in production; a protocol to analyse and mitigate PdM alerts the model generates, thus facilitating domain technicians the application of PdM in the system; definition of how to monitor model performance in production using data-

driven signals and domain knowledge to analyse and define when retraining is required; and guidelines to define how the retraining process should take place.

As explained throughout this section, the methodology is iterative, which means that backward steps are recommended to create a model that fits better use-case requirements. According to the previous argument, even when the model development has finished and it is into production by the implementation of the last step, it can be further improved and should be monitored to adapt to industrial evolving requirements. However, even if there is no need for the model to be continuously under development, it could have different versions as time goes by to adapt to new circumstances and integrate novel knowledge.

Chapter 5

Validation of the data-driven PdM methodology in industrial environments

The aim of this chapter is to validate the data-driven PdM methodology proposed in Section 4 by its application in three industrial use-cases. This empirical validation involves using the methodology to guide the life-cycle of data-driven PdM systems in use-cases with diverse requirements, characteristics and maintenance objectives. The first use case consists of modelling correct engine working data to detect anomalies in run-to-failure aviation data; the second estimates and explains remaining useful life on bushing testbed experiments; and the third use-case implements an adaptable semi-supervised anomaly detection and diagnosis in press machine process data.

5.1 Semi-supervised anomaly detection and diagnosis in simulated aerospace data

This section presents the implementation of the data-driven PdM methodology proposed in this thesis on an aviation engine use-case. Engines are the principal components of aircrafts, so their right maintenance is critical to ensure flight safety and reduce maintenance costs.

This use-case focuses on the application of PdM's anomaly detection stage on turbofan engine data. This stage has been addressed by state-of-the-art semi-supervised data-driven PdM models that learn from correct data to detect anomalous cycles. The methodological steps implemented in this use-case for the PdM life-cycle management

are contained in Figure 5.1, which ends with a functional model that fulfills use-case's requirements.

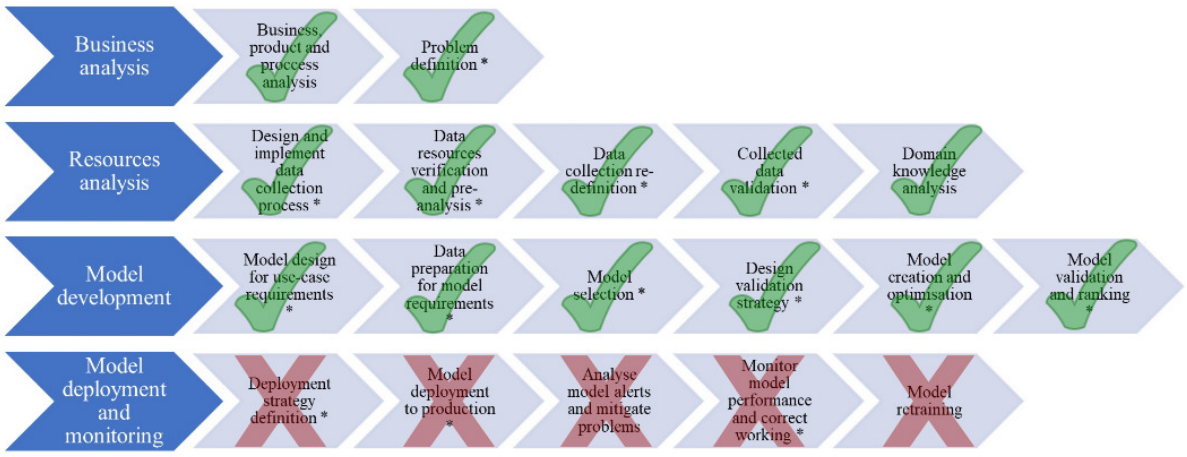


Figure 5.1: Diagram of methodology adoption in the turbofan use-case. The implemented steps are indicated with a tick and not implemented ones with a cross.

5.1.1 Use-case definition

5.1.1.1 Process and dataset description

The data used for this use-case belongs to *turbofan engine degradation simulation dataset*, released by the Nasa in the repository [267]. It has been created by generating run-to-failure engine data on a simulation environment, where each simulation starts at a random point of the engine life where it works correctly, and monitors its evolution until an anomaly happens and afterwards reaches the failure state.

According to Wikipedia [288], the turbofan is a widely used engine in aircraft propulsion. It is composed by a turbo that generates mechanical energy from combustion, and a fan that uses this mechanical energy to accelerate air rearwards. Its main components are presented in Figure 5.2.

Firstly, the *Business analysis* stage has enabled to understand and analyse the turbofan process. In addition, the objective to detect novel anomalies in engine data using semi-supervised data-driven models has been set. Secondly, *Resources analysis* stage has enabled to learn domain knowledge about the problem and assert that the monitored

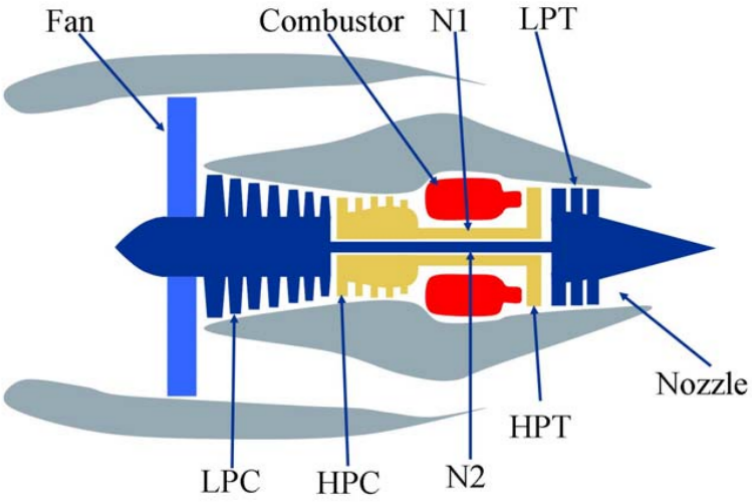


Figure 5.2: Scheme of main turbofan components by Frederick et al. [18]. The abbreviations of the figure refer to the following terms: N1 fan spool speed, LPT low-pressure turbine, LPC low-pressure compressor, HPC high-pressure compressor, N2 core spool speed and HPT high-pressure turbine.

variables are suitable for the defined objective. Finally, in the *Model development* stage different semi-supervised anomaly detection models have been developed, tested and ranked according to their accuracy. The *Model deployment and monitoring* stage has not been implemented in this use-case given that it is not required for being a simulated dataset.

The database contains three operational variables: altitude, mach number and throttle resolver angle. It is also composed by 21 sensors that measure temperature, pressure and speed variables in engine's different components, and variables related to coolant and air flow. More information about this simulation database can be found in the articles by Saxena et al. [269] and Frederick et al. [18].

This database has been widely used for research purposes given that its experiments contain different operational conditions and develop different failure modes, and these facilitate experimentation. The majority of state-of-the-art works on this dataset tackle the problem supervisedly, using the number of cycles until each experiment ends as target RUL variable. Conversely, this scenario rarely happens in industrial and manufacturing environments given that industrial companies ensure their assets are working correctly and try to prevent failures.

The database contains 4 run to failure datasets. However, in this use-case only 2 datasets have been used: the ones that contain the same EOC observations during each experiment because these enable modelling engine's condition evolution with time. Concretely, the datasets F001 and F003 have been used, which share EOC but the first has only one failure type and the latter contains two failure types.

5.1.1.2 Requirements and objectives

The objective of this use-case is to create a semi-supervised anomaly detection model that learns from correct working machine data and detects novel failure types in a simulated dataset. The model has to work with data of the same EOC correctly.

The anomaly detection models that have been validated in this simulated use-case are used for further experimenting in the industrial process use-case explained on Section 5.3. Therefore, these algorithms must address several industrial requirements:

- Robustness: models are validated with different data splits using cross-validation.
- Damage estimation: in addition to anomaly or not anomaly label assessment, models provide a damage index that increases with higher magnitude anomalies, or a health index that decreases with higher magnitude anomalies.
- Accuracy/complexity trade-off: this use-case requires accuracy and simplicity, therefore, complex models are only justified when achieving higher accuracy than simpler ones. Moreover, withing a model architecture, simpler models are preferred to complex ones to facilitate explainability.

5.1.1.3 Predictive maintenance techniques

The majority of state-of-the-art works use this dataset for prognosis, calculating the remaining time to failure counting the number of cycles from each cycle to the last one and using this variable as target. Nonetheless, this use-case is addressed in a semi-supervised way in order to simulate industrial scenarios, where failure data is scarce.

The state-of-the-art and classical data-driven anomaly detection algorithms selected for this use-case are PCA [289], ELM [50], OC-SVM [290], a CNN-based Autoencoder (CNN-AE) [291], a CNN-based Variational AutoEncoder (CNN-VAE) [292] with the

same structure of previous autoencoder but having a stochastic latent space, and a CNN-based AutoEncoder combined with ELM in the latent space (CNN-AE-ELM) [293].

The algorithms PCA, ELM and OC-SVM have been executed and evaluated in two ways. The first way has evaluated experiments' observations individually, categorising them between normal and anomalous classes. The other way consisted of using a rolling window of length 30 and a slide of one to extract four simple traditional features for all sensors: maximum, minimum, mean and variance. The CNN-AE and CNN-VAE algorithms extract features that are adapted to the problem from preprocessed sensor data, and use the reconstruction error for anomaly detection. In addition, CNN-AE-ELM first trains a CNN-AE and then uses the encoder as feature extractor for ELM. The parameters of anomaly detection algorithms are explained in Table 5.1, and deep learning models' architectures are contained in Figure 5.3.

Table 5.1: Parameters of semi-supervised data-driven anomaly detection algorithms used in turbofan dataset.

Anomaly detection model	**Training parameters**
PCA sensor	num. components = 90%; loss func. = RMSE
PCA trad. features	num. components = 90%; loss func. = RMSE
ELM sensor	h_dim = 10; target = 1; loss func. = absolute difference
ELM trad. features	h_dim = 10; target = 1; loss func. = absolute difference
OC-SVM sensor	kernel = rbf; kernel coef. = 1/n_features; nu = 1%
OC-SVM trad. features	kernel = rbf; kernel coef. = 1/n_features; nu = 1%
CNN-AE sensor	2 feature engineering layers CNN&maxpool&relu flatten and fully connected relu optimizer = adam(lr=0.001, betas=(0.9, 0.999), eps=1e-08) early stopping, patience = 2 train loss = MSE; loss func = RMSE
CNN-VAE sensor	Same as CNN-AE, maps stochastic latent space to normal distribution (mean and std vectors)
CNN-AE-ELM sensor	First train CNN-AE, then use encoder as feature extractor for training an ELM; h_dim = 50.

These algorithms provide a damage index that increases with high magnitude anomalies, except OC-SVM, which provides a health index that decreases with high magnitude anomalies. For anomaly detection, a threshold on validation damage/health indexes is set to categorise as anomalous any test instance that surpasses it. This work has tested two techniques to set the anomaly detection threshold. The first has been the selection

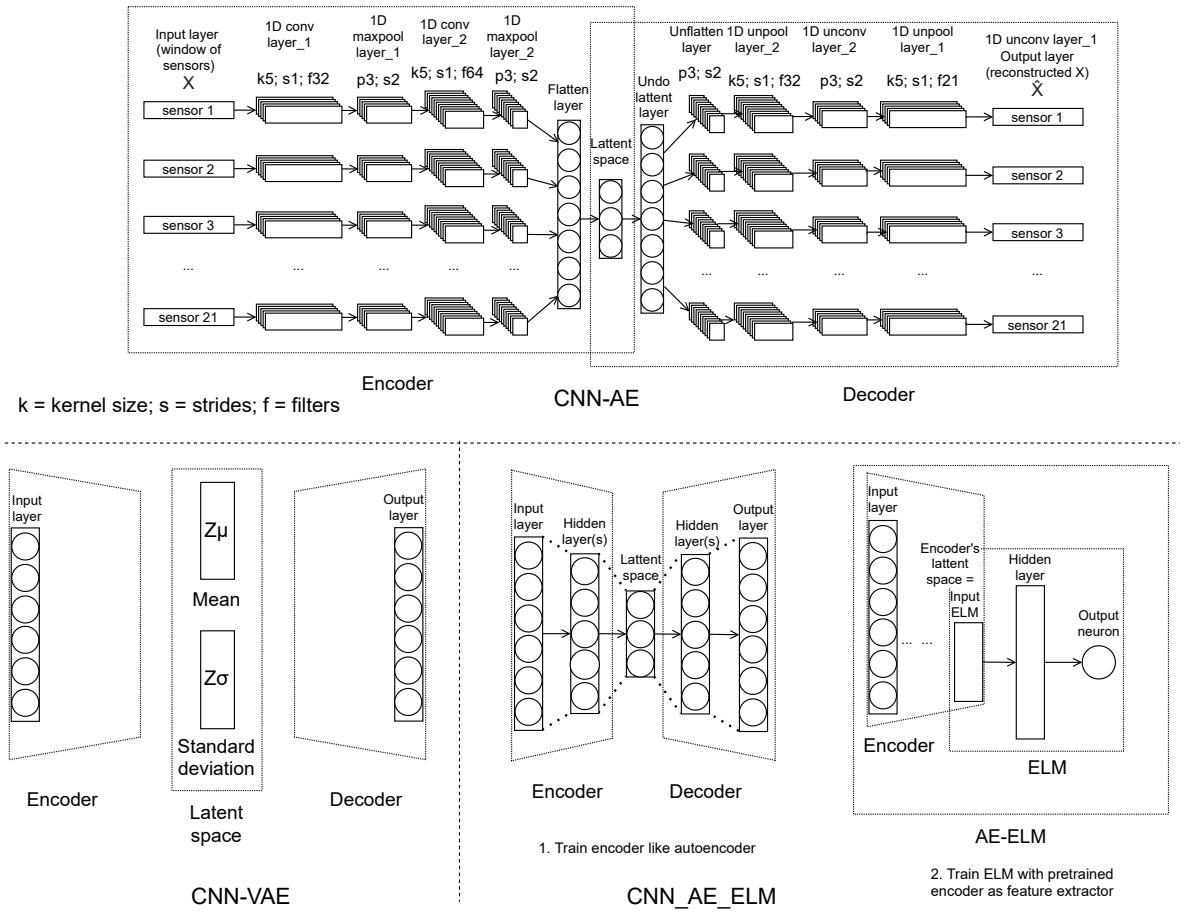


Figure 5.3: Scheme of three semi-supervised deep learning-based models for anomaly detection in turbofan dataset: CNN-AE, CNN-VAE and CNN-AE-ELM.

of the percentile 99 on correct instances under the assumption that 99% of training data is correct, similar to the work by Rodriguez et al. [294]. The second way has used the modified Z-score as proposed by Iglewicz et al. [295] and presented in Equation 5.1, where instances with values above 3.5 are categorised as anomalous [296]. In the equation, MAD is the median absolute deviation, x_i refers to each observation of the variable, $\tilde{x}$ is the median of the variable, and M_i represents the modified Z-score value of x_i.

$$\begin{aligned} MAD &= median(|x_i - \tilde{x}|); \\ M_i &= \frac{0.6745(x_i - \tilde{x})}{\text{MAD}} \end{aligned} \tag{5.1}$$

5.1.2 Experimentation procedure

The initial stage of the use-case implementing the methodology is *Business analysis*, which has been focused on understanding the turbofan process and defining use-case objectives and requirements. Turbofan is a run-to-failure simulated engine dataset; therefore, there is no business or manufacturing process to address. The objective of this use-case is to detect novel behaviours by modelling normal working conditions with semi-supervised anomaly detection data-driven models. The requirements of this use-case have been aligned with the characteristics of industrial companies, aiming to achieve a balance between model performance and complexity to facilitate the explainability of models when it is required.

Afterwards, the *Resources analysis* stage has been implemented, which consists of downloading the data from NASA's repository [267] to analyse and visualise it. Then, an analysis on the variables has been performed to check that data values are in the expected range, and analyse the relations among variables to learn domain knowledge about the problem. The data is representative for the problem given that it has been specifically designed to enable PdM in this use-case, monitoring variables that can indicate the degradation of engines.

Then, the *Model development* stage has been performed. Initially, a selection of state-of-the-art data-driven semi-supervised anomaly detection models for PdM has been performed, defining model characteristics and adapting them to address use-case's requirements and its data characteristics. In addition, the data has been prepared before it is fed to the anomaly detection models based on their requirements, by removing constants, standardising for PCA and z-scaling for the rest algorithms.

The validation strategy has been defined to simulate industrial environments where data-driven PdM models are trained with correct machine data collected in a period of time, and they are tested with data collected afterwards. The algorithms have been trained to minimise the modelling error on training data, and then a threshold on damage and health indexes was set on validation data to differentiate between normal and anomalous observations. 80% of the experiments have been used to train and validate models using a 10-fold cross validation, and they have been evaluated with the remaining 20% of the experiments that have been held out for testing.

Moreover, the data has been labeled to enable model training, validation and testing. The first 100 observations of each experiment belong to the correct working class, the last 25 observations are categorised as faulty, and observations between these classes are labeled as degrading. These assumptions are supported on experiments' run-to-failure design, their length, and domain knowledge about engines which explains they require several working cycles to degrade and derive into faulty state. In fact, only the last 20 cycles of each experiment have been used to check model performance in faulty data; this ensures that even the anomaly detection models that require a window of sensor data contain at least 5 cycles of anomalous observations when evaluating failure cycles. These data split and labelling procedures are summarised in Figure 5.4.

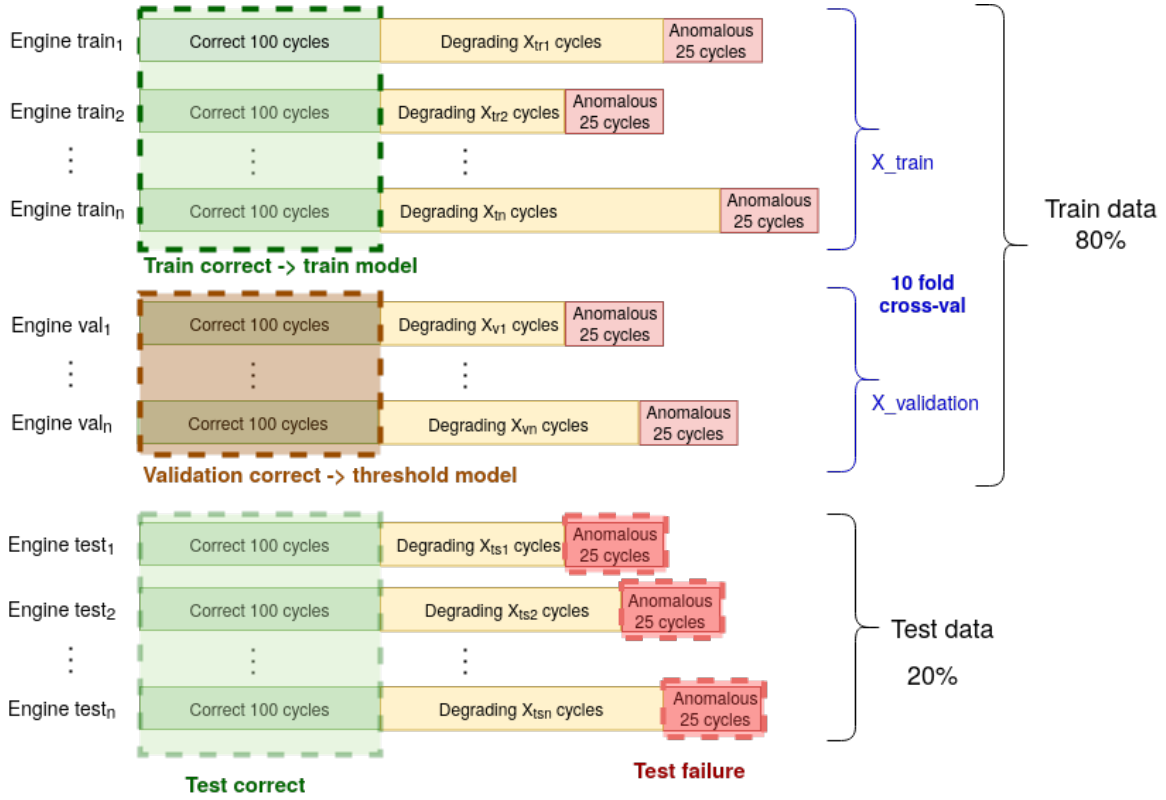


Figure 5.4: Procedure to split the dataset into train, validation and test sets containing correct and failure labels.

The semi-supervised data-driven anomaly detection PdM models selected for this use-case and their parameters have been defined in Section 5.1.1.3. For model ranking, the F2 score metric was selected to evaluate model performance on test data, which is contained on Equation 5.2. The F2 score overweights recall over precision, penalising

more errors in faulty cycles than in normal ones.

$$F_2 = 5 \cdot \frac{precision \cdot recall}{(4 \cdot precision) + recall} \tag{5.2}$$

This use-case ends with the development and ranking of anomaly detection models that address use-case's requirements. The *Model deployment and monitoring* stage of the methodology has not been implemented given that the work is supported on an evaluation dataset of the state-of-the-art and therefore, it does not require deployment.

5.1.3 Results

This section presents the results obtained after implementing the experimenting procedure of the previous subsection. Provided the anomaly detection models are trained in a semi-supervised way, their parameters cannot be optimised for target classes. For that reason, they have been trained using their default or commonly used parameters in the state-of-the-art, as explained in Section 5.1.1.3. The results of anomaly detection models on two turbofan datasets are presented in Table 5.2.

Table 5.2: Results of semi-supervised data-driven anomaly detection algorithms in two turbofan datasets, evaluated with F2-score. Two anomaly detection thresholds are evaluated: percentile 99 and modified z-score, indicated as *p99* and *M z-score* respectively.

Anomaly detection model	**Dataset F001**		**Dataset F003**	
	p99 F_2	**M z-score** F_2	**p99** F_2	**M z-score** F_2
PCA sensor	0.02	0.00	0.57	0.45
PCA trad. features	0.34	0.28	0.73	0.73
ELM sensor	0.45	0.54	0.20	0.34
ELM trad. features	0.37	0.54	0.18	0.32
OC-SVM sensor	**0.98**	**0.98**	**0.98**	**0.97**
OC-SVM trad. features	0.86	0.82	**0.94**	**0.94**
CNN-AE sensor	**0.98**	**0.98**	**0.92**	**0.96**
CNN-VAE sensor	**0.97**	**0.98**	**0.95**	**0.96**
CNN-AE-ELM sensor	**0.92**	**0.92**	**0.92**	**0.93**

The models that obtained the highest F2 scores are based on deep learning and OC-SVM. CNN-AE and CNN-VAE models obtain similar results on average, so the first is preferable over the second given that the stochastic part of CNN-VAE adds complexity to the network, being CNN-AE simpler. In addition, the model CNN-AE-ELM adds

complexity to the CNN-AE by including an ELM in the latent space while obtaining a little lower F2 score than the original model. Figure 5.5 presents the damage indexes of the CNN-AE anomaly detection model in one experiment.

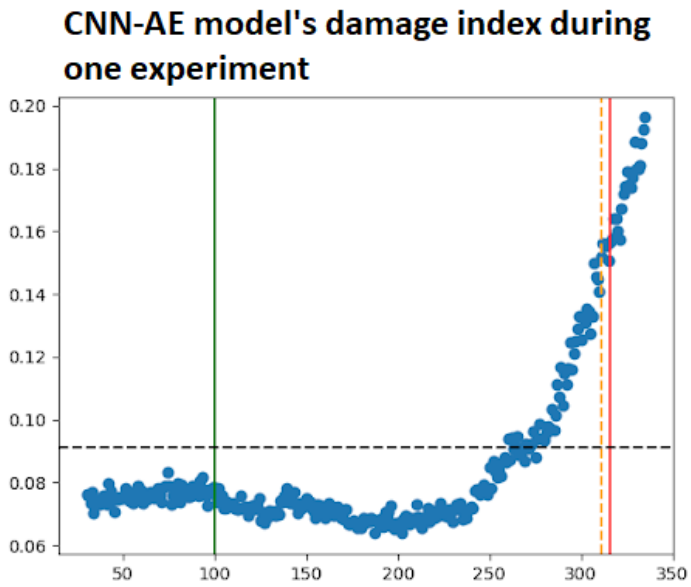


Figure 5.5: CNN-AE model's damage index for anomaly detection in one turbofan experiment. Cycle predictions are represented in blue dots, cycles up to green line are labeled as correct, cycles beyond orange dashed line are labeled as anomalous even though labels beyond red line are used for failure evaluation, and black horizontal line indicates the anomaly detection threshold set with percentile 99.

Regarding OC-SVM, both traditional features and sensor models obtain high F2 scores, but OC-SVM sensor model achieves higher scores specially in the first dataset. Figure 5.6 shows health indexes of one experiment using the OC-SVM sensor model for anomaly detection. This health index is similar to damage index, but it takes lower values when assets are more damaged, surpassing the anomaly detection threshold with smaller values. In contrast, the low F2 score values of PCA and ELM on sensor and traditional features indicate they are unable to detect novel anomalies. Figure 5.7 shows the damage indexes of one experiment using the PCA traditional sensor model for anomaly detection, where most damage indexes are below the anomaly detection threshold.

The two techniques implemented for anomaly detection threshold selection have reported similar F2-score metrics. Percentile 99 has the advantage of being easier to understand than modified z-score, but selecting the most adequate percentile in a semi-supervised way may be difficult. In contrast, modified z-score has the advantage of not requiring

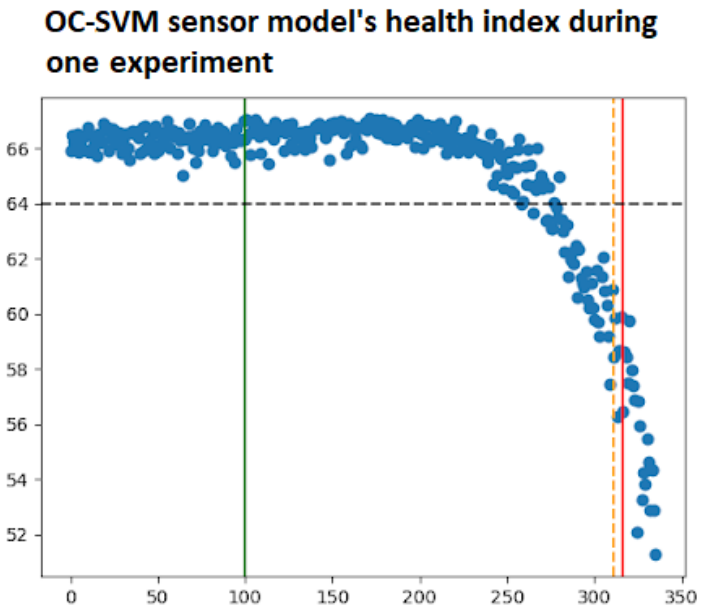


Figure 5.6: OC-SVM sensor model's health index for anomaly detection in one turbofan experiment. Cycle predictions are represented in blue dots, cycles up to green line are labeled as correct, cycles beyond orange dashed line are labeled as anomalous even though labels beyond red line are used for failure evaluation, and black horizontal line indicates the anomaly detection threshold set with percentile 99.

parameter selection, which can facilitate its general application. However, interpreting this metric is not straightforward, requiring more effort.

5.1.4 Discussion

This use-case focuses on the implementation of semi-supervised PdM data-driven models for anomaly detection in turbofan dataset. Its results cannot be compared with the literature given the majority of publications work on the dataset by performing prognosis, using the remaining useful life as label. Nonetheless, this work has selected state-of-the-art and classical data-driven anomaly detection models for PdM with the objective to facilitate comparison by training and testing them under the same criteria.

They have been trained to model correct engine data, and their performance has been measured by using F2-score on correct and failure data on test experiments. The best models are OC-SVM on sensor data, and deep learning based models: CNN-AE, CNN-VAE and CNN-AE-ELM.

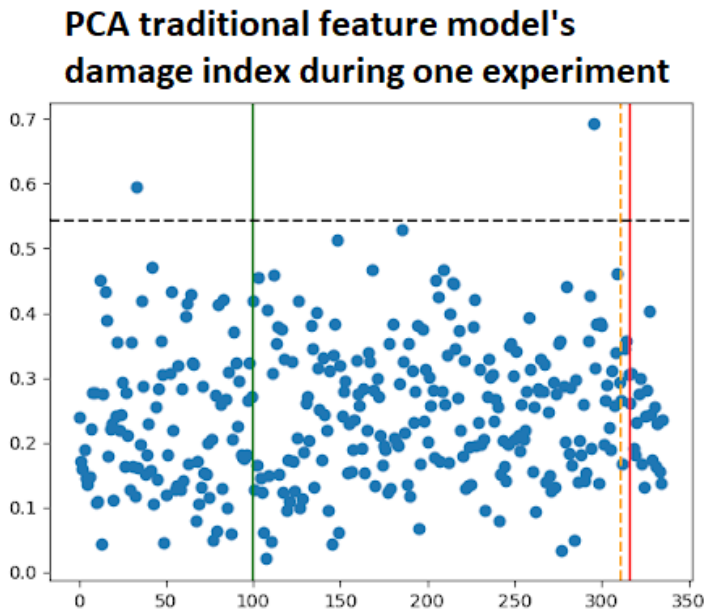


Figure 5.7: PCA traditional feature model's damage index for anomaly detection in one turbofan experiment. Cycle predictions are represented in blue dots, cycles up to green line are labeled as correct, cycles beyond orange dashed line are labeled as anomalous even though labels beyond red line are used for failure evaluation, and black horizontal line indicates the anomaly detection threshold set with percentile 99.

These selected models provided a damage index or health index that increases or decreases according to engines' damage condition. Moreover, the two implemented ways for anomaly detection threshold selection have worked correctly, where the percentile technique has the advantage of being easy to understand, and the modified z-score has more facility for generalisation.

The methodology proposed in this thesis has been used as guideline to implement each step of this use-case, successfully covering the *Business analysis*, *Resources analysis* and *Model development* stages. However, the *Model deployment and monitoring* stage has not been implemented given that the simulated dataset was downloaded from a public database and it did not require model deployment to production.

Even though domain technicians did not participate in the process, acquiring domain knowledge by reading about the process and learning about sensors and data has helped in many steps of the PdM life-cycle, including data preparation and model design. All in all, this use-case has validated the first three stages of the proposed methodology in a reference PdM simulation dataset.

5.2 Remaining useful life estimation on bushing testbed

This section presents the application of the methodology on a bushing testbed where fatigue tests are performed to characterise bushings. Bushings enable the transmission between linear and rotary motion, working like roller bearings in environments where big loads are applied. They are a critical component of machine tools, so their maintenance is essential to ensure right machine working condition.

The application of PdM on this use-case is the prediction of experiment remaining time for each data observation, considering that components are useful until reaching fatigue failure. This RUL model has been used to understand which are the features that have more influence in the experiments' remaining time and find relations among experiments, helping to design better bushes by taking into account the importance metrics of different design parameters. Figure 5.8 shows the methodological steps validated by the implementation in this use-case, ending with the model deployment to production.

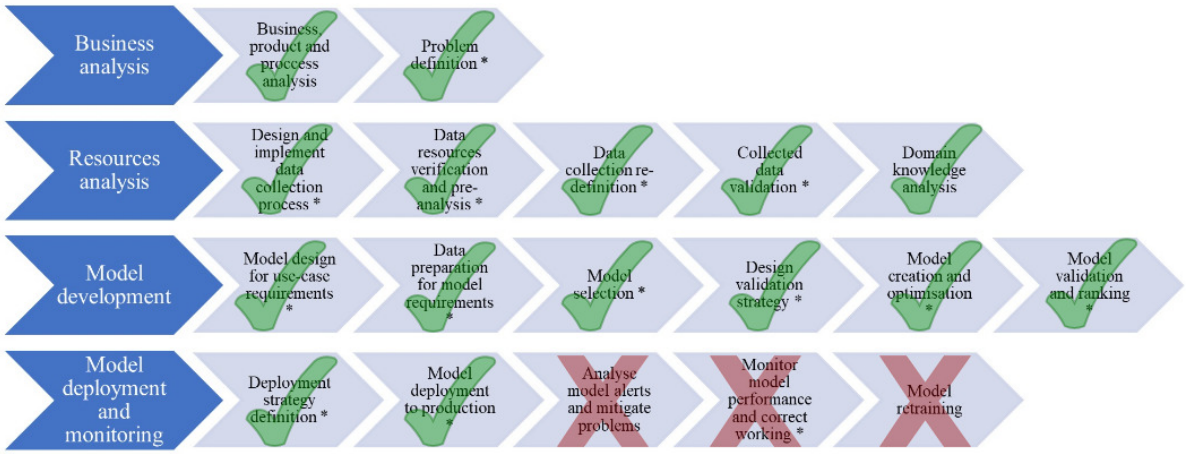


Figure 5.8: Diagram of methodology adoption in bushing testbed use-case. The implemented steps are indicated with a tick and not implemented ones with a cross.

Firstly, the *Business analysis* step in this use-case enables to understand the testbed and learn background about machine process. The objective of the use-case has been defined as creating a data-driven model that estimates the RUL of bushing experiments and explains its predictions to help discover the features that have more influence on

bushing life. Secondly, the *Resources analysis* stage collects the data from the testbed, performs an analysis on the dataset and enables to learn knowledge about the physical process. Moreover, it selects the most representative features and asserts that the collected variables are suitable for the use-case. Thirdly, *Model development* stage focuses on developing a data-driven model that estimates the RUL of bushing experiments that is robust to changes in EOC. The data is prepared for the model and its predictions are explained using XAI techniques, which are interpreted by domain technicians. Finally, in the *Model deployment and monitoring* stage, the developed model is deployed into an industrial computer to gather the data from the bushing testbed and integrate its predictions with the Human-Machine Interface (HMI). This way, operators can see the RUL predictions of bushings in real time.

5.2.1 Use-case definition

5.2.1.1 Process and dataset description

The data collected for the research of this work is based on a set of experiments performed in a fatigue testbed. Their aim is to find the characteristics under which bushings are able to tolerate the biggest accumulated load, which equals to seeking characteristics that make experiments last longer. The discovered optimal operational characteristics will be extrapolated to real machines in order to improve their components' working life. This is the reason why testbed's EOC are similar or proportional to real ones.

The testbed consists of a hydrodynamic journal bearing and a shaft that is rotating inside it, presented in Figure 5.9. This system is used to reduce friction between moving and static parts. The gap between them is lubricated with oil, enabling the hydrodynamic state. The PhD dissertation by Hassasin [297] presents the underlying theory of hydrodynamic bushings, and uses a testbed to apply high frequency vibration analysis techniques for PdM.

Expert technicians can perform general approximate reasoning and predict when the experiment will end just few seconds before failure occurs, by monitoring sensor variables and analysing them based on domain knowledge. Taking into account the high dimensionality of the data, they are not able to extract the influence of each variable in their remaining time and neither they can predict the remaining life until last observations. XAI techniques have been used to address this issue.

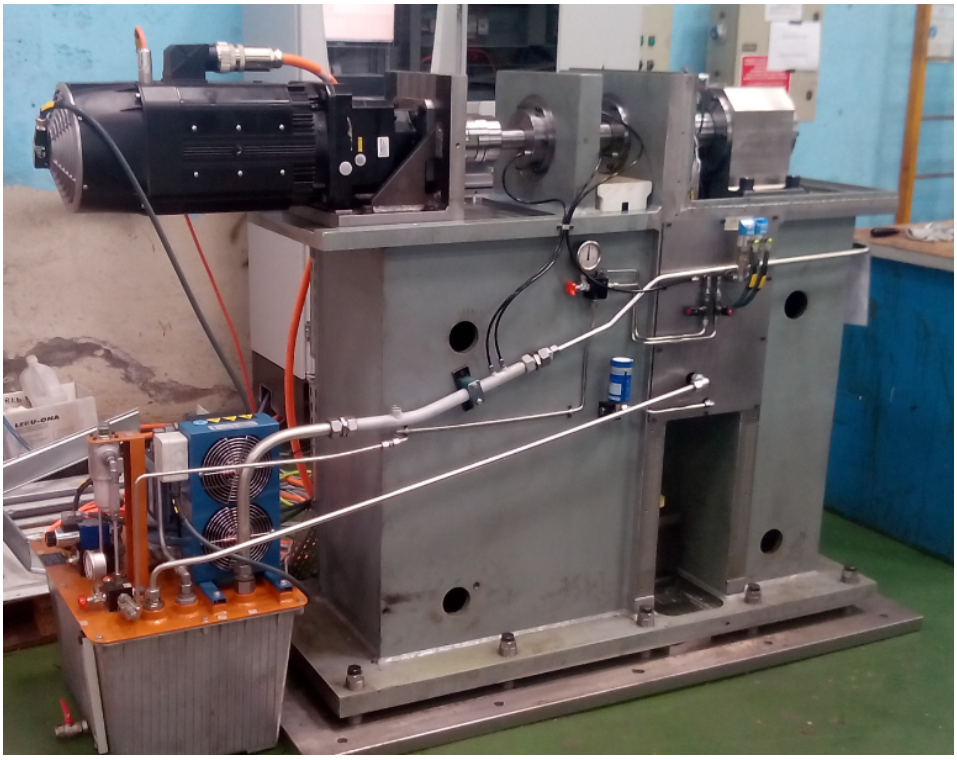


Figure 5.9: Bushing testbed.

The dataset consists of 576 experiments and 97 EOC variables. Some variables are time-series data collected with a sampling rate of 1 sample per second. The remaining are process variables composed of experiment characteristics and identifiers, so they are constant for each experiment. The time-series variables have been collected from sensors related to speed, load, temperature and lubrication. Furthermore, material quality has been measured using testers.

Given the industrial nature of data, the *same* characteristics tested in different experiments have different results. This happens due to differences in components' manufacturing tolerances, environmental conditions, assembly adjustments etc.

The experimenting procedure consists of designing which characteristics to test in a set of experiments. Firstly, an analysis is performed to determine which are the characteristics that have high influence on bushings' life based on domain knowledge. Then, mechanical calculations are performed to estimate the bushings' life for different configurations of the selected characteristics. Finally, several of these configurations are selected and tested in the bushing testbed to analyse their performance in a physical environment.

In the experiments, the average duration is calculated to abstract from components' manufacturing variability.

The dataset lacks of incorrect or unfinished experiments because expert technicians have removed them.

5.2.1.2 Requirements and objectives

The main objective of this use-case is to estimate the remaining experiment life based on process data and material characteristics. This can help to reduce the number of experiments necessary to categorise each tested characteristic, helping domain technicians to optimise experiment design strategy. It can also reduce the time and resources needed to test each experimented characteristics by only focusing on the most relevant variables.

To achieve that, the influence of variables in the duration of experiments has to be analysed, inferring knowledge. This process combines expert-knowledge and XAI techniques to explain model's decisions. Furthermore, explainability techniques are used to analyse experiment characteristics grouped by feature relevance.

To sum up, the resulting PdM model is required to have an accuracy acceptable from domain knowledge perspective. Moreover, the model must have the interpretability characteristic so that domain technicians can understand model predictions and therefore be able to trust it.

5.2.1.3 Predictive maintenance techniques

For PdM model selection, common ML regression models of state-of-the-art have been selected with the exception of deep learning models, to fit the dataset and use-case requirements. The models analysed in the dataset have been gaussian naive bayes, linear support vector regressor, k-neighbors regressor, linear discriminant analysis, random forest regressor and XGBoost regressor, using their default parameters for python's library scikit-learn. All models have been used for supervised regression problems and they use different techniques to predict the target class.

Despite the last two models are tree-based algorithms, their differences have effect on model explainability. Random forest regressor is a bagging ensemble method, which

underlies on the combination of less precise uncorrelated models to improve the precision and generalisation in an assembled model. It aggregates many random regressor trees trained with different features and data subsets, forming a forest. The regressions are done by averaging tree regressions. XGBoost Regressor works similar to Random Forest, ensembling trees to create a forest. Conversely, this is a boosting ensemble method, which aggregates new trees to fix the errors generated by existing trees based on gradient boosting method.

5.2.2 Experimentation procedure

The work conducted in this use-case has been guided by the methodology presented in this thesis. The steps have been adopted to address use-case characteristics such us understanding the process, dataset particularities, and explainability of predictions.

The first stage is *Business analysis*, to understand the relevance of bushing testbed and the relation of the experiments performed on it with the stamping machines manufactured by the company. Bushings are a critical component of press machines given that their failures result into stops of the entire production line, and their high probability of failure when the machine is working in improper conditions. Therefore, finding the characteristics that improve the robustness of bushings and reduce the failure probability is a priority. This objective is pursued with the fatigue experiments performed in the bushing testbed, where the duration of an experiment is related with the robustness of that bushing characteristics.

The business analysis stage enables to understand how the physical process works theoretically and practically. The objective of this use-case has been defined as modelling the RUL of bushing experiments to predict the life of bushings according to their characteristics and enable interpretation of this model. Consequently, this model permits to learn insight about the data, and be analysed to infer the variables that influence the experiments and how they are correlated.

The second stage is *Resources analysis*. The data has been collected from the testbed and a preliminary data analysis has been performed to understand the data, diving into the process combined with domain knowledge. After that, the data has been analysed with statistical and visualisation tools and the best features for the objective of the use-case have been selected with help of a domain technician. The data has been checked

to be in its expected range, the relations of variables have been analysed and domain knowledge has been learned to relate the variables with their physical meaning and theoretical background. Therefore, this stage ensures that the dataset contains relevant features that are related to the problem and that no additional data sources are required.

The third stage is *Model development*, which consists of generating the ML-based remaining life estimator by implementing ML steps. Preprocessing has been performed by replacing each Not Available (N/A) value with the mean of that variable in the train dataset, the categorical variables have been encoded and the decision to not normalise or standardise the variables because the chosen model did not need it has been taken, to keep the original magnitude and distribution of data. The dataset has been partitioned for model training and testing in the following way: a 10-fold cross-validation stage. It has also been used to analyse not only model prediction stability through data partitions, but also to analyse the stability of data random splits; concretely 80% of experiments for training and remaining 20% for testing.

After preprocessing, feature engineering has been performed. Feature selection has been done to reduce the number of predictor features because having a high dimension of correlated variables added complexity and information redundancy. Firstly, constant variables have been removed. After that, correlation analysis has been created using heatmaps and, together with expert knowledge technicians, derived variables have been removed since they are combination of other variables, selecting only the sensor and setting variables. After that, several feature selection techniques have been tested for ranking and selection of the best features according to error scores.

Concretely, the following techniques have been used and compared: Recursive Feature Elimination (RFE), to remove one feature at a time using default model feature importance; selecting the most relevant feature incrementally using ELI5's [9] black-box feature importance for the model trained with all features, starting with the highest relevant feature, then adding the second one, and so on and so forth until all features have been selected; minimum Redundancy Maximum Relevance (mRMR) based on mutual information [298], which maximises the relevance of selected features with the target feature and at the same time minimises the redundancy among them; selecting the k-best features according to mutual information and ftest indicators; and using lasso and ridge linear regressors' feature weights as importance for feature selection.

Feature extraction and dimensionality reduction methods can also be used to reduce even more the dimensionality and enhance model performance, but these new features are more difficult to understand than original process data. Therefore, these techniques have been discarded, also supported on the fact that the dimensionality of selected variables have also already been reduced to a small subset and the model performed correctly with them.

The ML model selected for the work must be a regressor that, given the data observations of an experiment on a certain moment, predicts the number of seconds remaining until it ends. During an experiment, the model receives data for each elapsed second, which includes characteristics of the bushing and EOC data described in Section 5.2.1.1. To train RUL models, the full data of training experiments has been used, and their corresponding RUL data has been appended as target class. It is calculated by counting the number of seconds until the experiment ends for each data observation.

After selection, model evaluation metrics have been chosen to measure the overall performance of the model. Then, the model has been executed, analysed its accuracy has been ranked using the chosen score, its explainability has been analysed, and iterated in these steps until obtaining an acceptable model.

The explainability of this model has been computed and interpreted using local and global explainability methods. Concretely, ELI5's global explanations have been used to analyse the global influence of each variable in model predictions together with expert knowledge. This technique has been chosen since it removes a variable and analyses model's performance decrease. Local explanations have also been calculated using the library LIME, where the contribution of each predictor to model output is calculated. This is done by measuring how each variable contributes either positively or negatively to the prediction, fitting a linear regression to its neighbour observations. The reason for using linear models is that these are inherently interpretable, so they are used to analyse the behaviour of small perturbations in the observations.

Another experiment has used the final model developed in the previous step and adapts it to fit the data grouped by experiments of similar characteristics. First, a selection of which experiment characteristics are interesting for data grouping based on similarity has been performed. Afterwards, one model has been created for each group. The objective is to create a model to analyse how feature relevance varies among models of

different experiment groups, to find groups that share patterns. For that, the agglomerative clustering model has been chosen to rank features using the feature importance calculated using the library ELI5, the same global explainability method used to rank features in the previous stage. Model explanations have been analysed together with expert technicians, taking into account their theoretic mechanical knowledge and expertise in order to test whether they are comparable or not.

The fourth and last stage is *Model deployment and monitoring*, which consists on deploying the random forest model of 10 features to the experimental environment. First, an industrial computer has been selected to deploy the model, given that it has connection with the bushing testbed and therefore, model predictions can be integrated with the HMI. For the deployment, an environment that contains the requirements to execute the model has been prepared, the model and required preprocessing techniques have been copied there and then the model has been tested to assert it works correctly in the industrial computer. Then, the connection with the testbed has been established to collect the data of experiments, estimate the RUL and return the prediction to HMI so that the operator that performs experiments in the bushing testbed has access to real-time predictions of the bushings' RUL, according to their characteristics and EOC.

5.2.3 Results

The visual and analytical analyses of the dataset have helped to understand and identify data characteristics and how fatigue experiments are performed. Moreover, a summary data dictionary has been created in a table of 97 rows and 6 columns. Its columns contain the following information of the recorded variables: name, type, unit, type of feature, meaning and comments, and each row contained the information of each recorded variable.

There is the need to measure the performance of the developed models for any ML task in order to measure model precision for the desired task, allow comparisons among different models, and assist in model optimisation, architecture selection and parameter setting. Two common techniques for regression evaluation have been analysed: MAE and RMSE. MAE is the normalised sum of all the absolute errors between real and predicted values (2.2); and RMSE is the square root of the normalised sum of all the squared errors between real and predicted values (2.3).

The MAE metric is more intuitive to understand given that it indicates the absolute error performed by the model on observations on average. In contrast, RMSE is more sensitive to outliers than MAE [67], being interesting for some uses. Therefore, RMSE has been chosen as a metric to train models so that they are more robust for outliers. For model ranking, both metrics have been analysed, and for model overall performance evaluation MAE has been chosen given it is easier to interpret and understand, even for domain technicians.

Table 5.3 shows the aforementioned machine learning regression algorithms' performance on 10-fold cross-validation using the MAE and RMSE metrics.

Table 5.3: Model comparison before feature selection process.

Algorithm	**RMSE**		**MAE**	
	mean	**std**	**mean**	**std**
Gaussian Naive Bayes	450.39	16.82	331.59	16.56
Linear Support Vector Regressor	421.14	208.67	300.69	137.16
K-Neighbors Regressor	258.57	20.65	163.44	13.20
Linear Discriminant Analysis	106.54	18.12	69.94	5.67
XGBoost Regressor	**53.96**	12.33	**36.10**	5.11
Random Forest Regressor	**61.64**	12.97	**36.29**	5.43

The models that have obtained the best results are the tree ensemble ones: Random Forest and XGBoost. On average, XGBoost obtains better RMSE and slightly better MAE than Random Forest. However, given their standard deviation, t test has been used to analyse whether there is enough statistical evidence to assert that models' performance is different or not along 10 folds using MAE and RMSE metrics. The results show there is not statistical evidence on performance metrics and therefore, the criteria for choosing one model over the other switched to explaination facility, another requirement for the created model. Thus, Random Forest Regressor has been selected over XGBoost Regressor given it is simpler and provides naive feature explanation by feature importance, which makes it is easier to interpret.

Then, the chosen random forest's robustness have been analysed by executing it with 20 random initialisation seeds over the mentioned 10 folds cross validation. The results show it is stable along different folds with a average MAE of 35.89 and standard deviation of 5.70. These results are accurate according to domain technicians, given experiment duration is several hundreds of seconds and technicians can only estimate remaining

life in experiments' final observations. These results have been used to optimise further experimentation, by splitting a random subset of 80% for training and 20% for testing to iterate over feature selection enabling faster model ranking.

After performing feature selection using expert knowledge and evaluation of model feature importance, the result is a random forest regressor model executed on a subset of 10 original features that obtains a MAE of 41.29 on new split test data. Concretely, five setting features, four sensor features and an operational feature have been selected. Figure 5.10 shows the performance of the model during an average experiment. Table 5.4 shows the same experiments as Table 5.3 after performing feature selection, using the same 10-fold cross validation and metrics for evaluation. Given the results are similar for last two models, the selected 10 feature subset keeps representative information of the original data, simplifying the problem and thus facilitating explainability.

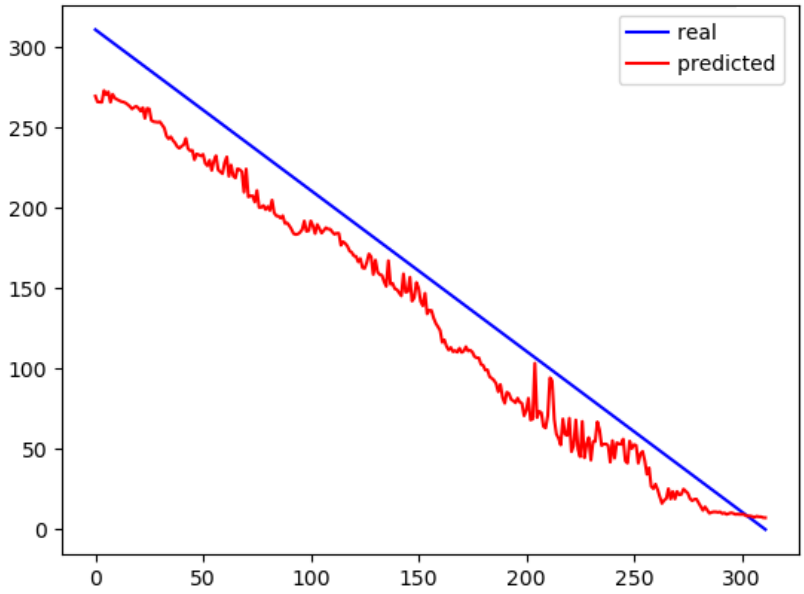


Figure 5.10: Stage three's final model remaining life prediction on an average fatigue test. x axis indicates the number of observation whereas y axis indicates the remaining time in seconds. The blue line indicates the real remaining life and the red signal indicates model's remaining life prediction.

Global explanations have been used to guide feature selection and also to understand overall model performance by analysing feature importance. This enables feature rank-

Table 5.4: Model comparison after feature selection, selecting the 10 most relevant features.

Algorithm	**RMSE**		**MAE**	
	mean	**std**	**mean**	**std**
Gaussian Naive Bayes	299.25	20.50	215.83	13.28
Linear Support Vector Regressor	954.88	1332.73	770.48	1066.45
K-Neighbors Regressor	264.12	21.98	193.80	16.58
Linear Discriminant Analysis	187.56	15.86	140.57	12.71
XGBoost Regressor	**52.54**	9.68	**36.29**	4.32
Random Forest Regressor	**58.36**	11.86	**38.25**	4.80

ing and analysis of contribution to model prediction. Moreover, it has facilitated contrasting model overall performance with domain technicians conclusions according to their expertise and analysis of experimenting results, which turns out to be similar. Likewise, local explanations are used to analyse model behaviour by explaining predictions of several experiments' observations. Domain technicians have found applications of local explanations on analysing how target variable is influenced by changes in selected features. This could be used for experimental setting optimisation, achieving better results with less parameter testing. Conversely, local explainability results could be improved given linear models may not be able to model non-linear data correctly.

Local explanations have been extracted from the previous model using the library LIME. Figure 5.11 shows how each feature affects model estimations given an experiment observation. On the top part, model's prediction for the recorded data instance is shown; in this case it is estimated to be 836 seconds. The Feature list shows names of predictor features in current observation ordered by importance. The bottom left graphic shows how each feature of previous table affects the estimated value. The features are ordered by local importance top to bottom, where value indicates how much influence each has on current prediction, and the color indicates in which direction its change affects model's prediction.

Figure 5.12 shows the intrinsic feature importance provided by the random forest model. In addition, Figure 5.13 shows feature importance calculated using ELI5. To obtain global explanations, ELI5 uses the feature permutation technique, which consists of randomly shifting only the values of one feature and not modifying the rest. Then, model score changes between original and this data are analysed for each predictor shifting, being an objective way to analyse which features are more important for the

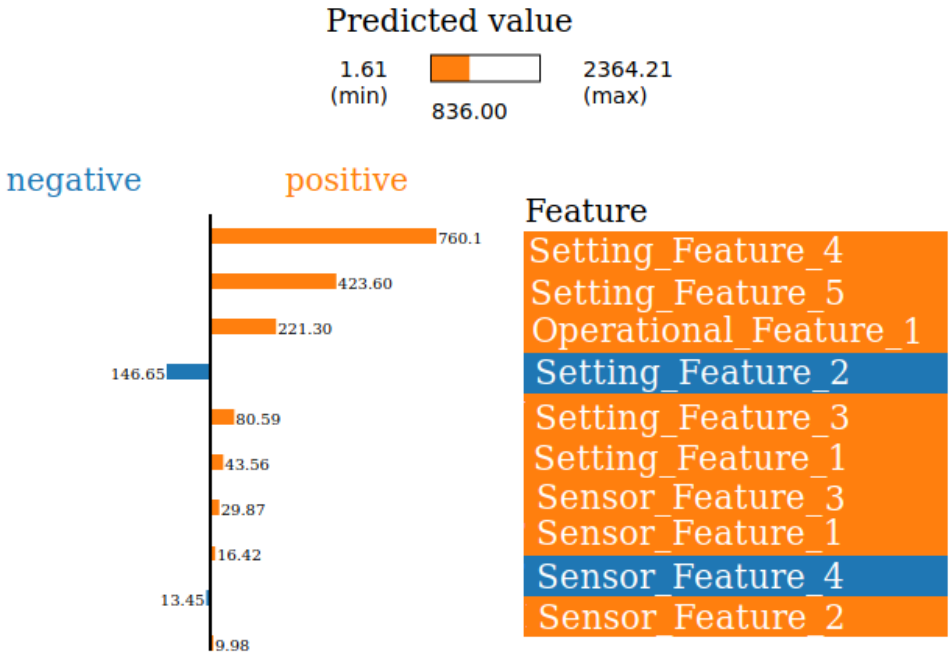


Figure 5.11: Local explanation of a remaining life prediction in an experiment observation. The prediction equals the real value: 836 seconds.

model. Both techniques obtain similar global explanations, but the importance order of the features varies in several cases. According to the figures, the first 5 variables concentrate most of the model information, while the remaining 5 have less importance.

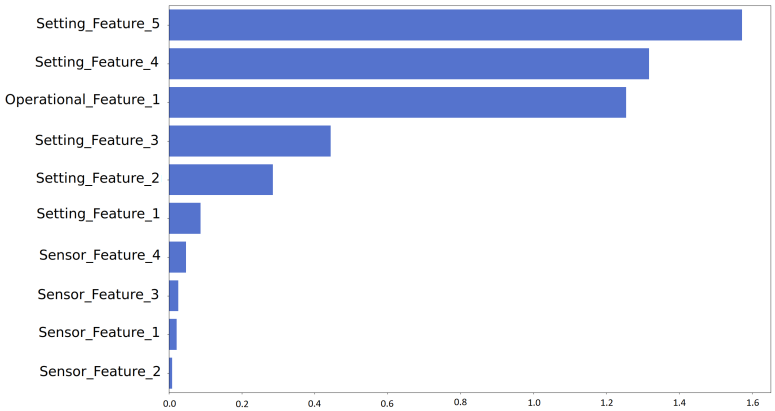


Figure 5.12: Intrinsic global explanation of remaining life predictor model. It shows feature importance ordered from highest to lowest. The relevance magnitude is indicated in x axis.

To the already mentioned model of 10 features, automatic data-driven feature selection techniques have been applied with the objective to select only the most relevant vari-

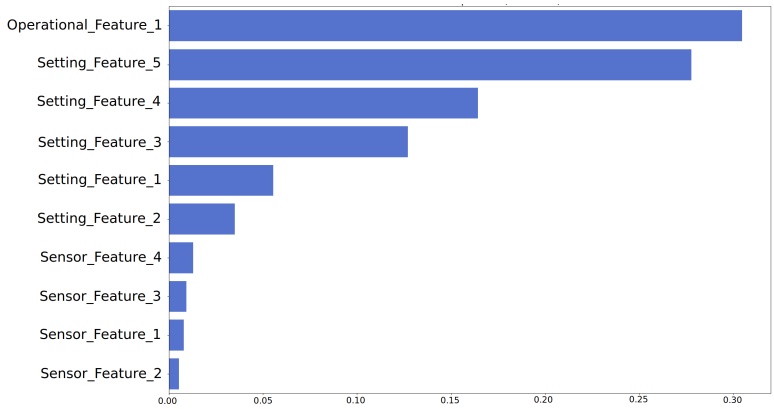


Figure 5.13: Global explanation of remaining life predictor model using LIME. It shows feature importance ordered from highest to lowest. The relevance magnitude is indicated in x axis.

ables that still achieve an acceptable performance on the model. The feature selection techniques and their results are summarised in Table 5.5. The results are ordered by lowest to highest MAE of the chosen model with selected 5 features, and mean metrics of each reduction steps from 10 to 3 features.

Table 5.5: Feature selection techniques tested to reduce dimensionality of model from 10 to 3 features using sklearn, mRMR and XAI-based methods.

Algorithm	**5 features MAE**	**10 to 3 features mean MAE**
RFE, model intrinsic feature importance, step=1	**45.74**	**57.37**
Select most relevant incrementally, ELI5 feature importance, step=1	**45.74**	**57.58**
mRMR, using MIQ, reduce step=1	109.67	78.67
kbest, mutual info	112.02	103.16
kbest, ftest	191.46	127.36
select lassoCV	176.63	146.07
select ridgereg	218.13	176.38

Finally, these are the experiments performed in the fourth stage of the work, using the data grouped under same characteristics of setting variables. First, importance scores have been calculated from the model and they have been used as new variables to identify defined groups. Using this data, an agglomerative clustering have been performed. Its results have been analysed by cutting the dendrogram of Figure 5.14 into different cluster sizes, from 2 to 10.

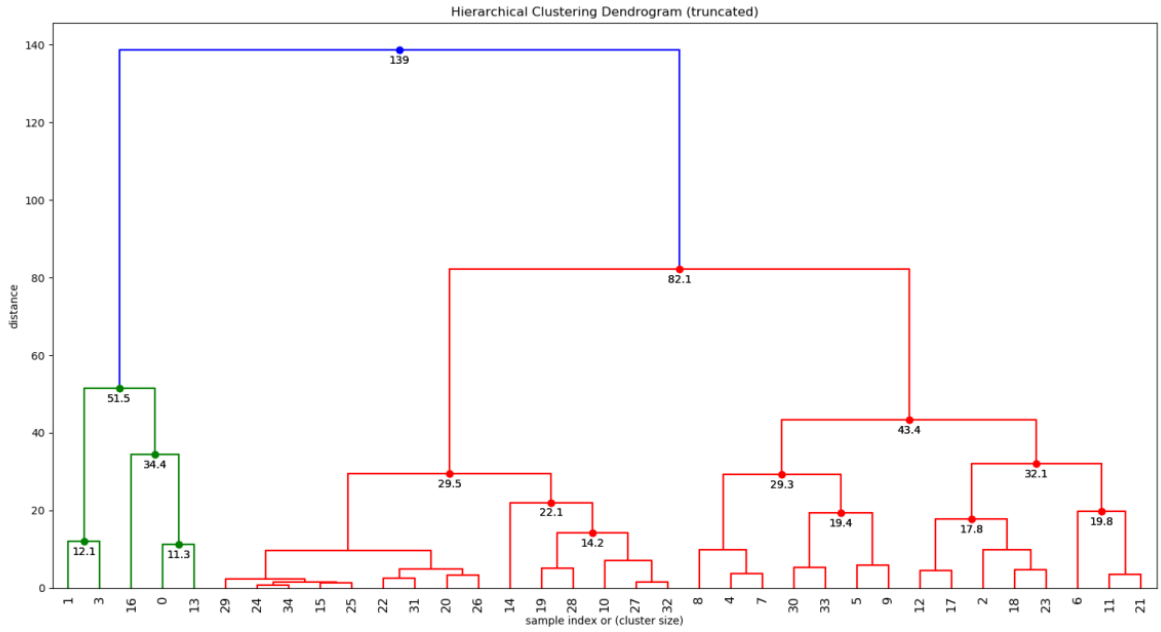


Figure 5.14: Dendrogram of clustering feature importance score of models trained with data grouped by process variables.

The dendrogram shows that two or three clusters could be clearly distinguished by setting the maximum cluster distance to 100 and 60 respectively. To understand the clusters in a visual way, a plot of experiment results has been created using two representative variables. Figure 5.15 presents the results of choosing a cluster size of 3, where clusters are disperse. Near group instances are expected to be in the same cluster, whereas far instances are expected to be in different clusters. However, the fact that this does not happen means that training one model for each group, and clustering based on model feature importance does not provide additional information to domain technicians.

5.2.4 Discussion

This use-case implements a ML model that predicts accurately the remaining life of experiments and gives global and local explanations in order to understand its predictions. In this work, industrial process knowledge has been crucial for both accuracy and interpretability goals.

The variable subset selection process has helped to simplify the developed model while maintaining accuracy. Performance analysis has also been visualised using plots of each experiment in test data. The analysis of top feature performance has concluded that

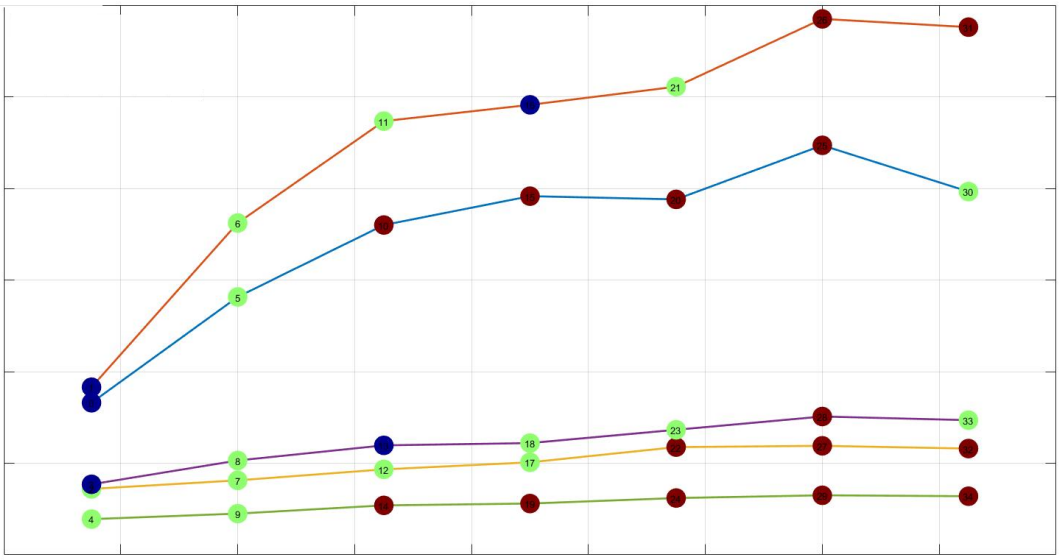


Figure 5.15: Results of clustering experiments. x axis shows a feature related to fatigue and y axis shows a experiment setting variable. Each ball represents a group of experiment data and its color indicates the cluster assigned by the algorithm. The coloured lines that join these balls identify experiment characteristics.

most relevant features are setting variables, given that four out of five most relevant ones belong to this type and the rest are less important.

The application of XAI techniques on random forest has been successful. Global explanations are accurate because the model performs a random shift of feature values. However, local explanations have improvement potential given that there are categorical variables whose change can have high impact on model's output. Even so, global and local explainability techniques have provided feature importance weights that are interpretable in a similar way.

Regarding the model of third stage, it works correctly for the designed task. Its aim is to estimate fatigue test life and infer knowledge about which are the variables that have higher influence on it. The model has been deployed into a computer connected to the PLC of bushing testbed, where technicians are able to see the predicted remaining life of new fatigue tests. After that, the evolution of model performance should be tracked and a retraining strategy will be defined in future stages of the work.

Conversely, the models generated for each group of experiments in the fourth stage have not improved performance over the previous general model. A possible reason could be

that there is much less data available for training each group model, which difficulties their generalisation. Moreover, these models cannot be grouped by feature importance because, as the clustering model has shown, the generated groups do not show any clear relation.

The combination of data scientist mindset with expert knowledge has enabled the integration of complex data-driven models into an industrial use-case. This achievement has been possible by the application of the methodology presented in this thesis. It has enabled the combination of data-driven PdM models with XAI techniques, taking advantage of expert knowledge to guide the PdM life-cycle: model creation, interpretation and optimisation, in order to link its predictions with physical meaning. Global and local understanding of model predictions have been performed, even in the case of black-box ML models.

All the stages of the methodology have been validated and all its steps up to *model deployment to production*, which in this case is the deployment to bushing testbed environment. The methodology has successfully adapted to interpretability requirements of the prognosis model, facilitating the collaboration among working profiles.

Future research is promising to integrate explainable machine learning models to optimise, automatise and assist knowledge discovery in industrial processes based on data. This would bring the reduction in maintenance costs and improve machines' availability, performance and quality.

With the mentioned objectives in mind, future research in this field could move towards obtaining an interpretability-accuracy trade-off. Accordingly, models should be accurate enough to perform the selected task and at the same time interpretable. Thus, expert technicians could infer knowledge, perform diagnosis and finally trust their predictions.

5.3 Semi-supervised anomaly detection and diagnosis in press machine data

This section displays the application of this thesis' data-driven PdM methodology on press machine data. Press machines are rotating machines that form input material by cuts and deformations, and one of the biggest types of machine tools. They play an important role in manufacturing processes, being a key component on stamping production lines of metal-formed components. Their maintenance is critical to ensure a right machine working condition, correct manufacturing quality, prevent downtime and reduce costs.

This use-case implements a real PdM use-case, addressing industrial company's requirements by modelling process data. This work is the most complete of the thesis, with the additional difficulty of modelling real process data that is not controlled unlike simulation and testbed data. Therefore, it validates the methodology application on a production line of one industrial company. As explained in Figure 5.16, all methodological steps have been implemented except for *model deployment to production*, which will take place in the future.

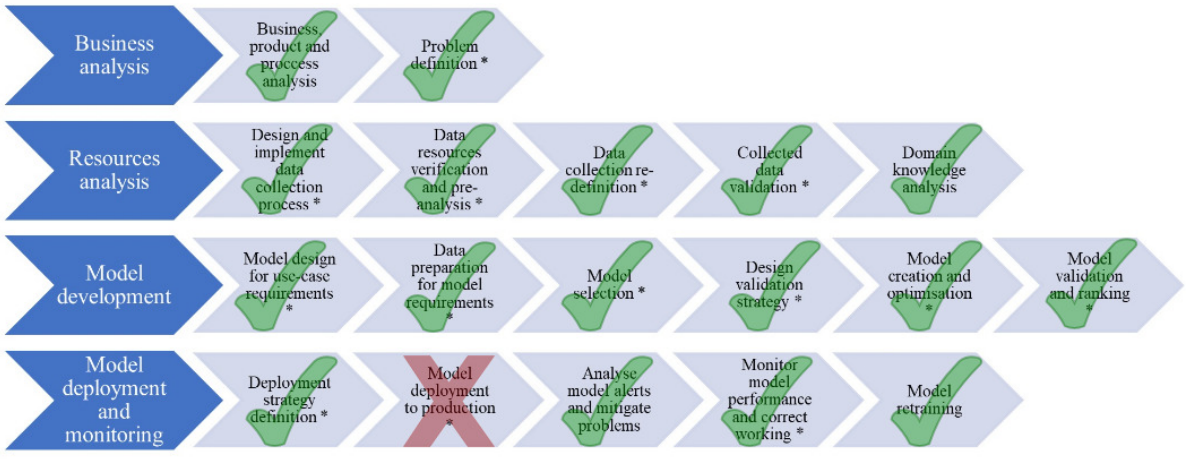


Figure 5.16: Diagram of methodology adoption in stamping machine use-case. The implemented steps are indicated with a tick and not implemented ones with a cross.

Firstly, the *Business analysis* stage has enabled to understand the business, stamping process and how press machines work. In addition, the objective to develop semi-

supervised anomaly detection and diagnosis data-driven models was set. Secondly, the *Resources analysis* stage has enabled to collect, analyse and validate the data from the industrial process. Thirdly, in the *Model development* stage, the AD and diagnosis models have been developed, and AD model's adaptability to changes in EOC has been addressed using transfer learning. Finally, the *Model deployment and monitoring* stage has enabled to define the deployment strategy, define the strategy to handle model alerts for RCA, and define how to monitor the AD system to identify when it requires retraining.

5.3.1 Use-case definition

5.3.1.1 Process data

The experimentation dataset used in this use-case has been collected from a press machine of a stamping production line, facilitated by a private company. Press machines are machine tools that create metal-formed components by applying pressure, forming input material with cuts and deformations. Figure 5.17 shows an image of press machine and its main components.

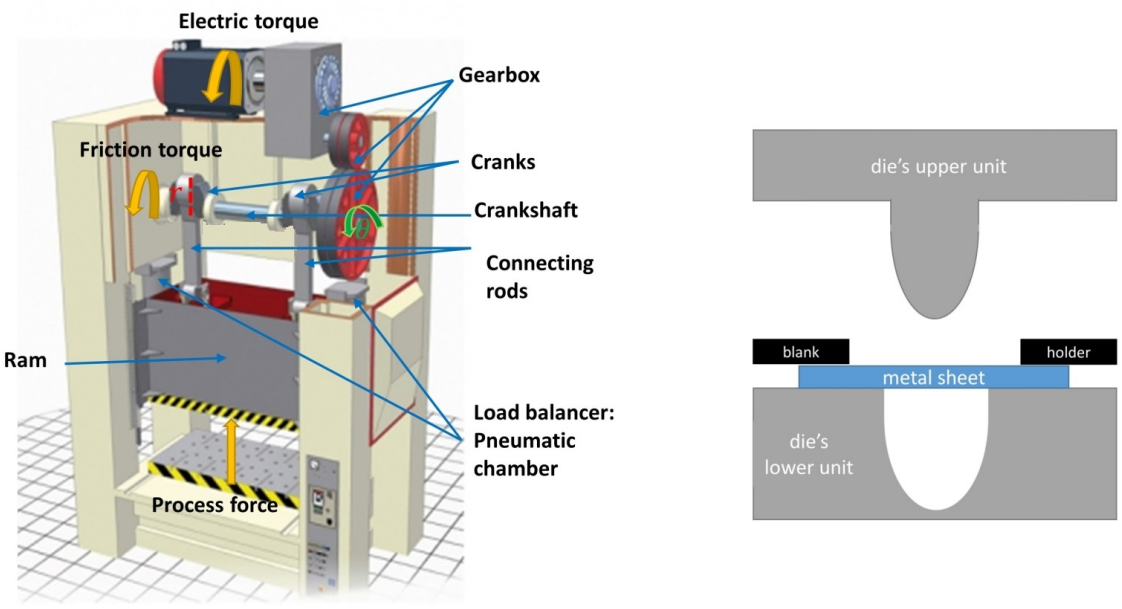


Figure 5.17: Image of a servo press machine with its main components on the left, and slide and ram scheme on the right, by Olaizola [19].

This use-case's press machine has a conventional electrical drive, composed by an electric

engine connected to a flywheel and clutch-brake mechanism, and crank and connecting rod system that turns cinematic chain rotary movement into slide linear movement. The slide moves perpendicular to the floor, going from the top dead centre (0°) to bottom dead centre (180°) and then back to top dead centre (360°), to complete the stamping cycle of 360°. The slide forms input material by stamping in the die, which is a two-part element specifically designed to address the requirements of finished blank presented in right part of Figure 5.17.

The data has been collected by sensors for each stroke, containing both single measure and evolution variables. The dataset is composed by the following variables: 2 speed sensors, 2 power consumption sensors, 6 force sensors and 2 position sensors.

With the objective of reducing industrial data variability to facilitate data analysis, data has been grouped by similar environmental and operational conditions. This grouping is achieved by filtering data by stroke identifier and workorders.

5.3.1.2 Requirements and objectives

An analysis on collected data reported no anomalies neither by machine users nor by domain technicians. Therefore, 90% of collected data is assumed to be correct and the remaining may have outliers. When only correct data is available and there is no faulty data, according to the methodology, semi-supervised approach should be the most suitable. The selected models should either analyse time-series sensor data, or use time-based extracted features from sensor data and then perform AD on these features.

These are additional requirements the anomaly detection PdM model should address in this use-case:

- Accuracy: high ability to differentiate correct and anomalous strokes.
- Novelty identifiability: ability to detect novel anomalies that were not available for model training.
- Isolability: ability to distinguish different failure types.
- Explanation facility: ability to explain which data patterns made the model reach each prediction.

- Damage indication: when an anomaly is detected, a health index should be provided to help in damage severity analysis.
- Robustness to noise: ability to handle industrial sensor data.
- Adaptability and re-usability: ability to adapt to different workorder data of similar EOC and continue working properly.
- Online data processing capability.

The objective of this work is to obtain a semi-supervised anomaly detection model trained with correct condition machine data that identifies novel anomalies, where the most accurate model among different state-of-the-art data-driven models is selected. Afterwards, the diagnosis of these novel anomalies should be facilitated by the selected model, assessing anomaly severity, grouping data by failure similarity, and highlighting anomalies in data signals to help domain technicians identify their root cause. Finally, the model should be reusable and adaptable to data variability, and it should have the capability of real time execution.

5.3.1.3 Predictive maintenance techniques

The algorithms executed for anomaly detection are PCA [289], ELM [50],OC-SVM [290], a Two Dimensional CNN-based AutoEncoder (2D-CNN-AE) [299] and null space [143]. The parameters for training all mentioned models are contained in Table 5.6, whereas the architecture of 2D-CNN-AE and its details are presented in Figure 5.18.

The PCA, ELM and OC-SVM algorithms have been executed in two ways. The first way has been training and performing predictions for cycle data of strokes, setting a threshold on the loss using percentile, and then counting the number of cycle samples surpassing the threshold and setting another threshold on this number to determine if that stroke sample is anomalous or not. The other way has been the extraction of statistical features for each sensor variable and then perform anomaly detection in these features with a percentile threshold. The extracted features are: mean, variance, maximum and minimum values. Null-space and 2D-CNN-AE use cycle data as input, not requiring specific feature extraction.

For diagnosis by isolating failures on latent space features, t-distributed Stochastic Neighbor Embedding (t-SNE) [300] has been used for visualisation of latent space vari-

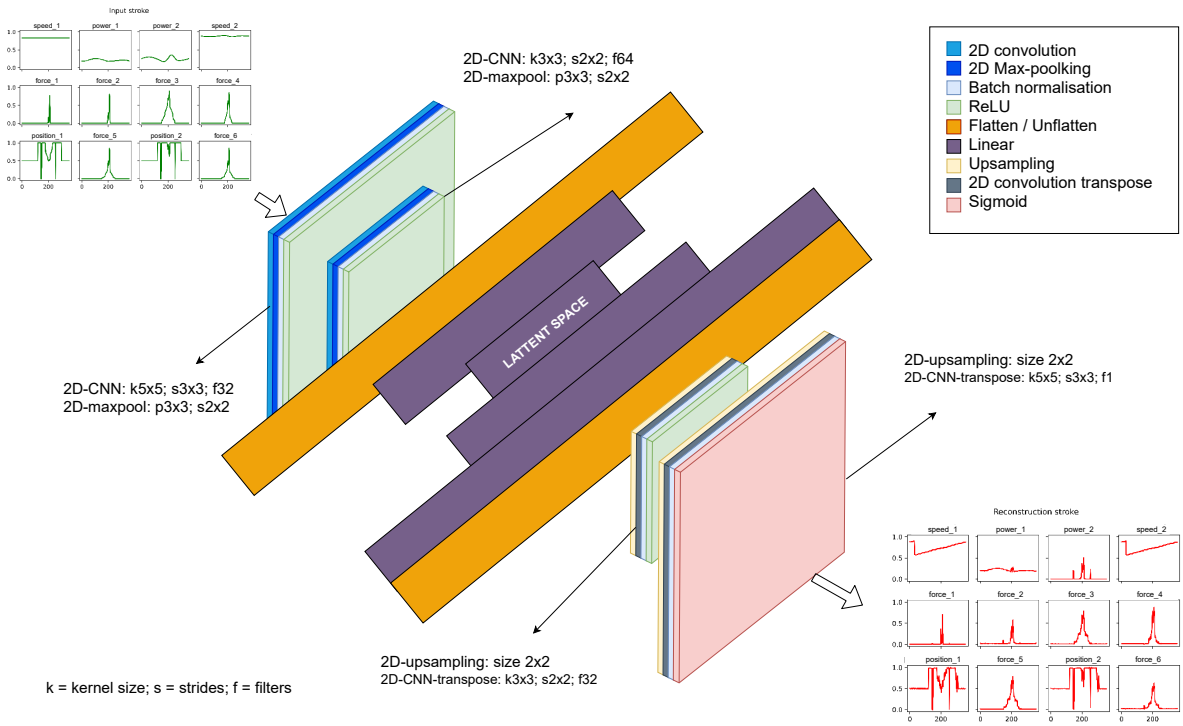


Figure 5.18: Architecture of 2D-CNN-AE and its parameters.

Table 5.6: Parameters for training anomaly detection models. Cycle and trad. feats refer to the way and data used in anomaly detection models.

Anomaly detection model	**Training parameters**
PCA cycle	num. components = 90%; loss func. = RMSE
PCA trad. feats	num. components = 90%; loss func. = RMSE
ELM cycle	h_dim = 10; target = 1; loss func. = absolute diff.
ELM trad. feats	h_dim = 10; target = 1; loss func. = absolute diff.
OC-SVM cycle	kernel = rbf; kernel coef. = 1 / n_features; nu = 1%
OC-SVM trad. feats	kernel = rbf; kernel coef. = 1 / n_features; nu = 1%
Null-space	alpha = 5; beta = 5
2D-CNN-AE	2 FE layers of 2D CNN&maxpool&relu&flatten&FFNN optimizer= adam(lr=0.001,betas=(0.9,0.999),eps=1e-08) early stopping, patience = 2 train loss = MSE; loss func = RMSE(reduce=mean)

ables; Ordering Points To Identify the Clustering Structure (OPTICS) [301] algorithm as a density-based clustering algorithm; GMM [302] as parametric clustering algorithm; and SOM [303] to project data in a new space based on competitive learning.

XAI techniques have enabled to track the loss throughout the 2D-CNN-AE model and link it with variables. Its objective has been to analyse which input variables and concretely which cycle points were causing the anomaly. For this purpose, the SHAP [76] library is used, explaining predictions by stroke. Concretely, GradientExplainer has been the selected method, which according to its documentation, *Gradient explainer combines Integrated Gradients [304], SHAP, and SmoothGrad [305] into a single expected value equation.*

Finally, model adaptability to same die data in new workorder, and to other die data have been tested. First, the original model and threshold have been tested, and afterwards, the experimented transfer learning by freezing convolutional layers and only retraining inner linear layers. This procedure should be enough to adapt the model for data of similar EOC, where convolutional layers are used as feature extractors and retraining linear layers adjusts the model to data variations.

5.3.2 Experimentation procedure

The initial stage of the use-case is *Business analysis*, which comprises understanding the business, how the manufacturing process works, and the components and operation of stamping presses. Afterwards, the problem that this use-case has to address is defined, focusing on modelling the stamping process. Concretely, the objective has been set to detect anomalies on the stamping process using semi-supervised data-driven AD models. The models have to adapt to changing EOC to continue working correctly with novel data. Moreover, the diagnosis of detected anomalies must be addressed in a semi-supervised way, supported on the developed AD models.

The second implemented stage is *Resources analysis*, collecting the data from the web application that is connected to the CPS of the industrial plant. The data has been analysed using statistics and visualisations, supported on domain knowledge to learn about the process, validate that the data is representative for the use-case objective, and ensure that monitored data is in the expected range. Consequently, the variables that are aligned with the use-case are selected and validated.

The third stage of the use-case is *Model development*. The first task of this stage is creating an anomaly detection model only trained with correct machine data. Afterwards, the selected anomaly detection model has been used for diagnosis, and finally,

its adaptation to data of different workorders and different dies has been analysed. All the experiments are designed to address industrial use-case's requirements and validated with domain technicians.

Press machines are widely tested machines designed for durability and minimisation of failure probability purposes. As a result, a press machine can work correctly during months or years before the first failure arises, which difficulties the monitoring of failure data. Therefore, to analyse the results from domain perspective, four synthetic failures have been designed by modifying correct stroke data. Moreover, different versions of each failure have been created by changing their magnitude. Synthetic failure types and their variations are presented in Table 5.7, whereas Figure 5.19 presents visual representations of those failures.

Table 5.7: Synthetic failures and corresponding signal modifications used for model validation.

Synthetic failure	**Signal modification and variations**
Force increment, simulating harder input material	Increase forces on 4 axis (+10%, +25%, +50% and +100%)
Die miss-adjustment, simulating press force anticipation	Anticipation of press force application (50°and 100°)
Machine degradation, higher power consumption for same forces	Increase power consumption (+10%, +25%, +50% and +100%)
Unbalanced loads	Exchange press input and press output forces

For data preparation, data has been first grouped by die ID and workorders, and then, these data subsets have been used for experimenting. Each workorder contains process data collected in a specific continuous time interval no longer than a week, which is delimited by production working periods. This grouping system is suitable for press machines given that each die works under similar operating conditions and performs the same forming process.

Then data has been divided for model creation and validation based on industrial production data order: first 80% strokes are used for training and validation, which are further divided into 90% for training and 10% for validation, whereas last 20% are used for testing. Afterwards, data has been preprocessed to prepare it for the PdM models, and then their results have been compared. The models have been trained under the assumption that after cleaning the dataset with domain technicians, at least 90% of training data is correct and the remaining could have outliers.

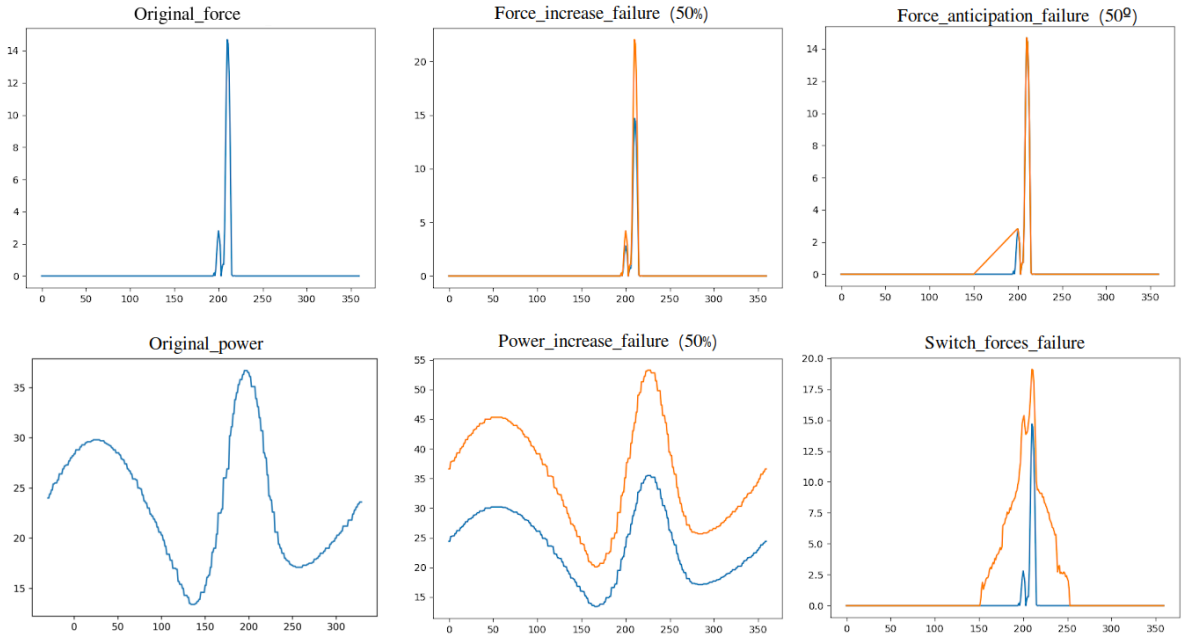


Figure 5.19: Two correct signals on the left, and the four synthetic failures on the right. Blue signal represents normal stroke data and orange signal represents one corresponding synthetic failure signal.

Next, a model evaluation and ranking strategy has been defined, which consists of using F1 metric for overall performance analysis, and precision and recall for further analysis; more information of these metrics is explained in Section 2.1.3. These metrics have been selected because they evaluate errors based on the target failure class and work well with imbalance datasets.

Validation has been complemented with a questionnaire to be filled by domain technicians, which contains plots of 14 selected strokes among normal and synthetic failure data. Its objective is to evaluate technicians' capability to differentiate synthetic anomalies by visual comparison with plots of correct strokes of the same signal. The survey has been answered by 9 domain technicians with high expertise on the field. Figure 5.20 shows an example question of stroke data belonging to the form.

The final stage of this use-case is *Model deployment and monitoring*, whose all steps of the methodology except *Model deployment to production* have been implemented; this step will be implemented in a future work. Initially, the developed models are planned to deploy into the cloud platform, consuming the process data uploaded to it for making predictions. Then, the anomalous alerts are analysed by domain technicians, who can

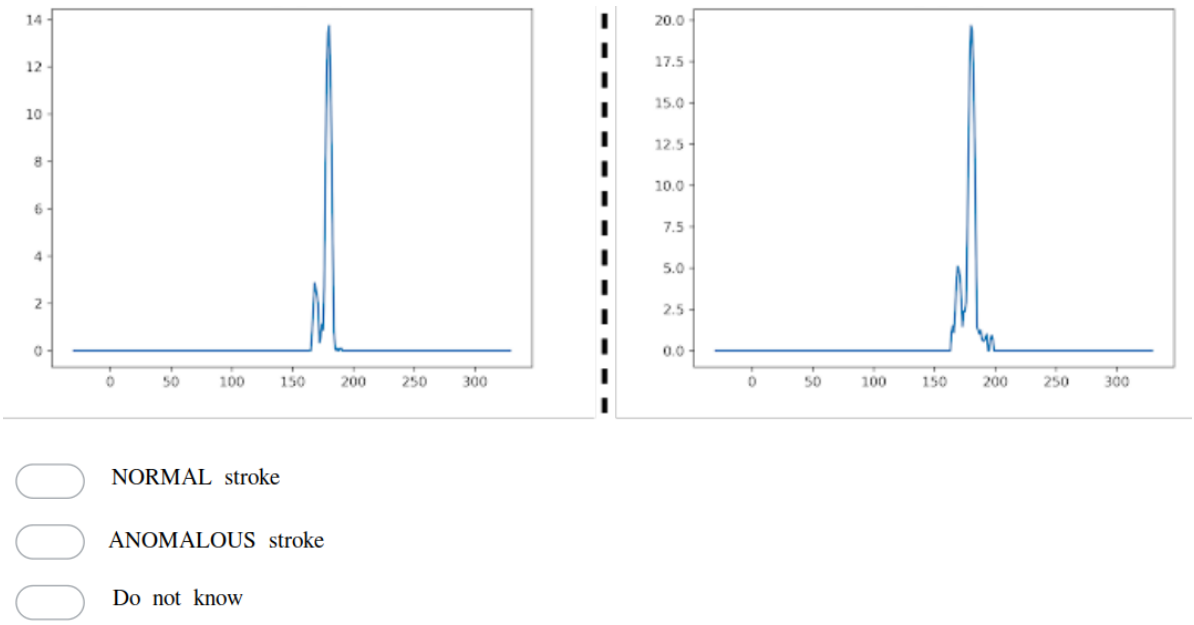


Figure 5.20: A sample of stroke question in the questionnaire for domain technicians.

use the developed diagnosis tools to identify the root cause of the anomaly and plan the corresponding maintenance actions to restore the healthy process state. Moreover, the adaptability of the AD model to changes in EOC has been ensured using transfer learning. When the AD model stops working correctly even using transfer learning, an analysis must be performed with domain technicians to determine its cause. This analysis must determine if there is an anomaly on the process or, in contrary, the model cannot adapt to EOC changes and therefore it requires retraining.

5.3.3 Results

This section describes and interprets the experimental results of this work. It is split into three subsections to facilitate their presentation and interpretation: anomaly detection, diagnosis and adaptability.

5.3.3.1 Anomaly detection

The first PdM stage is creating an anomaly detection system capable to distinguish normal and anomalous working conditions on monitored assets. For this task, a data subset collected in same workorder and die has been selected. This step has selected data strokes that shared EOC, thus reducing data variability to facilitate model creation

and validation.

After that, this dataset has been analysed with domain technicians using plots, correlations, and univariate statistical measures such as mean, in order to discard any anomalous stroke resulted by acquisition problems or missing data. Then, given monitored variables are in different scales, data has been preprocessed before inputting to selected algorithms according to their characteristics. Variables have been standardised to have a mean of 0 and standard deviation of 1 before inputting data to PCA to maximise variance, whereas variables have been normalised in range $[0, 1]$, bringing them to the same range while keeping dispersion before inputting to the rest of algorithms.

Then, the algorithms presented in Section 5.3.1.3 have been executed under the assumption that at least 90% of monitored strokes belonged to correct working condition, which result in selecting the parameters of Table 5.6. The results of these algorithms using F1-score per failure are collected in Table 5.8.

Table 5.8: F1 score per failure of anomaly detection models trained under the assumption that at least 90% of training data is correct. CNN-AE's p90 and p95 refer to the threshold used for anomaly detection, indicating percentile 90 and percentile 95 of correct validation data respectively.

Algorithm	**1) Force increase**				**2) Force anticipation**		**3) Power increase**				**Switch forces**
	10%	**25%**	**50%**	**100%**	**120°-170°**	**70°-170°**	**10%**	**25%**	**50%**	**100%**	
PCA cycle	0.27	0.74	**0.97**	**0.97**	0.09	**0.93**	0.10	0.16	0.65	**0.97**	**0.97**
PCA feats	0.67	0.67	0.67	0.67	0.67	0.67	0.02	0.09	0.62	0.67	0.67
ELM cycle	0.17	0.21	0.24	0.25	0.18	0.21	0.15	0.15	0.15	0.16	0.23
ELM feats	0.07	0.22	0.68	0.76	0.23	0.52	0.06	0.06	0.07	0.07	**0.91**
OC-SVM cycle	0.67	0.68	0.68	0.68	0.66	0.66	0.66	0.66	0.66	0.66	0.68
OC-SVM feats	0.41	0.67	0.67	0.67	0.47	0.67	0.08	0.06	0.04	0.02	0.67
Null space	0.82	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.47	**1.00**	**1.00**	**1.00**	**1.00**
CNN-AE p90	**0.91**	**0.91**	**0.91**	**0.91**	**0.91**	**0.91**	**0.91**	**0.91**	**0.91**	**0.91**	**0.91**
CNN-AE p95	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**

The results of Table 5.8 show that null-space and 2D-CNN-autoencoder are the algorithms that work better than the rest on average, but autoencoder is even capable of detecting the smallest versions of failures. After analysing its precision and recall, several correct strokes have been being classified as failure, so the threshold could be better adjusted than using percentile 90. For this task, a search for the best percentile threshold on correct validation data from 1 to 100 has been performed, selecting the one that

has obtained the best F1 score using synthetic failures of validation data. The best threshold found for 2D-CNN-AE is by using percentile 95, achieving a F1 score of 0.99 for each failure type and thus outperforming the rest algorithms. Similarly, null-space algorithm has achieved an average F1 score of 0.92, which is lower than autoencoder but has the advantage of not requiring failure data for threshold selection. Both algorithms also provide a damage index that takes higher values when they are fed with higher magnitude synthetic failures.

The analysis of domain technician's survey results shows that they are unable to detect the smallest failures with force and consumption variations of 10%, or 50°failure anticipation. Conversely, they have precisely detected more notorious failures like force and consumption with 50% and 100% variations, 100°anticipation and switch forces. The average F1 score of the survey is 0.82. This analysis validates algorithm's results also from domain perspective, obtaining comparable results to their expertise.

At this point, the online data processing capability of 2D-CNN-AE anomaly detection model has been evaluated in a Nvidia 2080Ti graphics processing unit, which is measured by the mean time required calculate 10 stroke data. The autoencoder model was used to make predictions of 50 stroke chunks, each containing 10 stroke data, and the average elapsed time was 7 milliseconds. In addition, null space algorithm requires 20 milliseconds to process each stroke sample. This performance test has validated model's real time data processing capabilities.

5.3.3.2 Diagnosis

After performing anomaly detection, isolating different failure types and diagnosing their root cause is the next stage of predictive maintenance. For this stage, clustering, visualisation, projection and XAI techniques have been used in the 2D-CNN-AE anomaly detection model of previous section given this achieved the best results overall.

The first experiment has required forward passing through the encoder of test correct and test synthetic failure data, which turns each stroke into a 32 feature vector. Its objective is to test the ability to differentiate failure types in this compressed space. Initially, a 2 component t-SNE to feature vectors has been applied with the objective to create a 2D space where clusters are visualised. This results in disperse clusters, so PCA was has been introduced as dimensionality reduction while keeping data variability to

facilitate t-SNE. PCA requires standadisation of features before input, and 6 components were selected to keep 95% of initial variability. After PCA, these features have been fed to t-SNE that is configured with 2 number of components, 10 learning rate, 10000 maximum iterations, and remaining parameters as default for sklearn. A grid search on perplexity has been performed, which is a parameter that represents number of nearest neighbors in manifold learning algorithms, so values from where 5-70 range in strides of 5, and range 70-200 in strides of 10 have been tested. The original paper states that typical perplexity values are between 5 and 50, but the results obtained in this range are disperse and increasing perplexity resulted in more robust clusters.

The results show three groups of data where differentiation between increase forces and increase power consumption failures are difficult given that failures with smallest data variation (10%) are very near in the new space. The experiment has been repeated, and then clustering has been performed using GMM with 4 number of clusters, with the objective to isolate the 4 failure types in t-SNE's embedded space of two dimensions. Low values of perplexity increase data sparsity in the new space, whereas high values increase cluster compactness. However, beyond a certain perplexity value, data distribution does not change much and clusters remained stable, so this point has been selected for final t-SNE results analysis. This point is achieved with perplexity of 145, which is shown in Figure 5.21: it contains real failure labels on the left, whereas results of GMM with 4 number of clusters are presented in the right. Figure 5.22 summarises the pipeline of clustering, projection and visualisation techniques used for diagnosis in this stage.

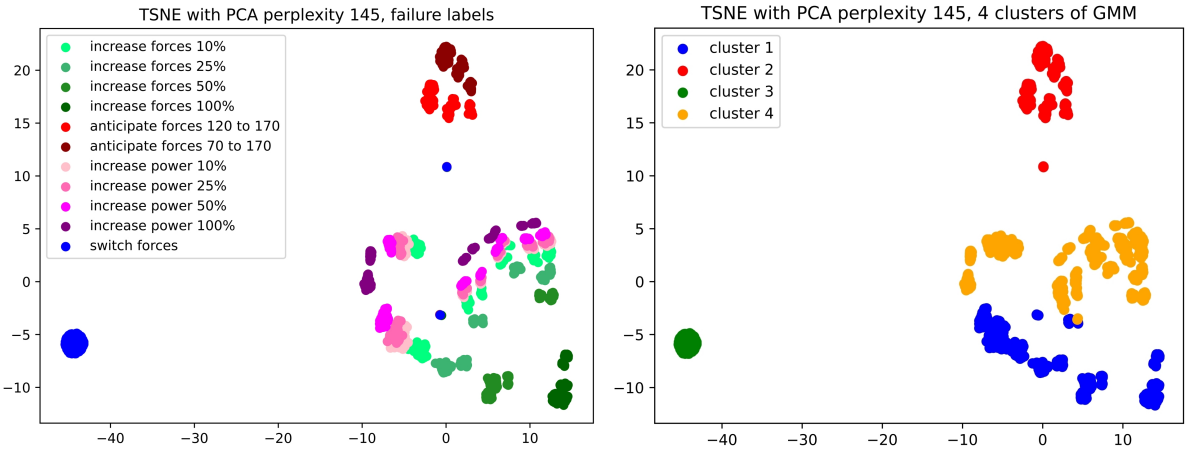
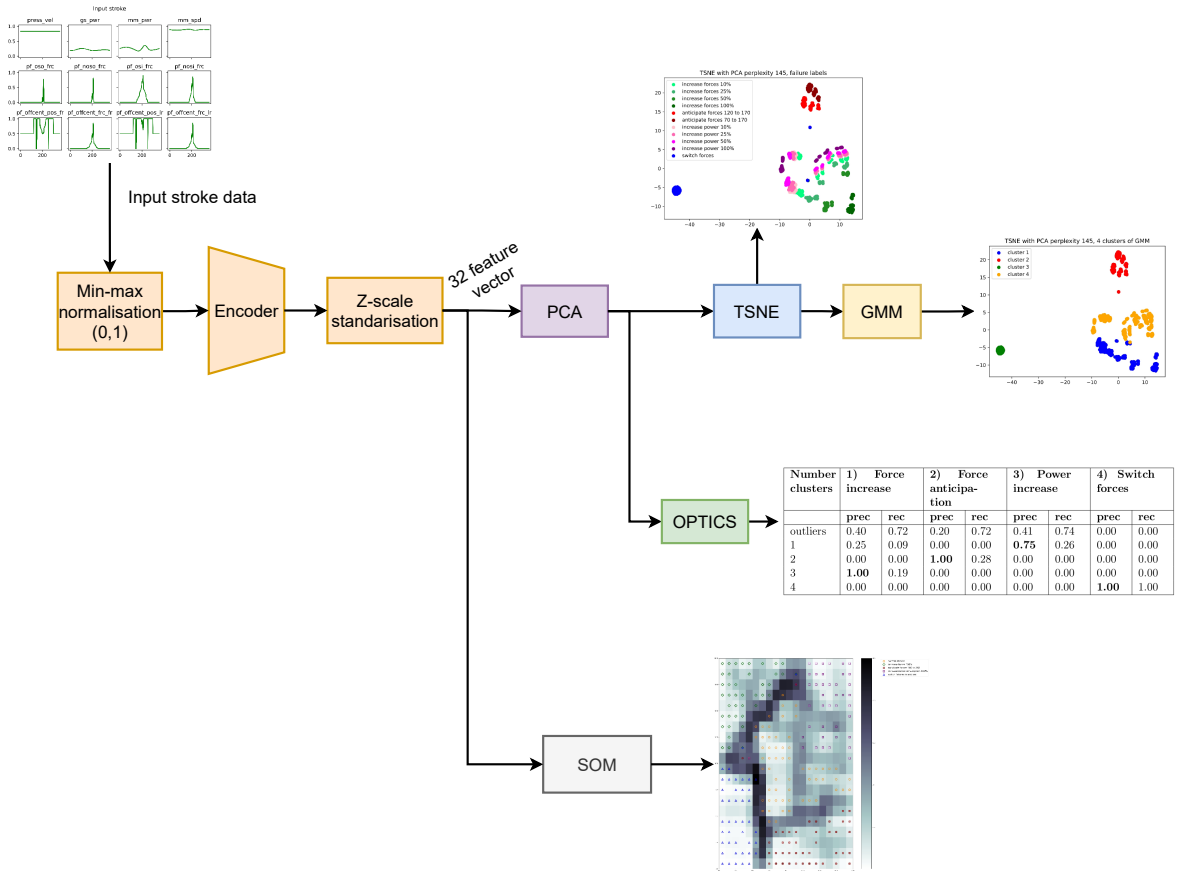


Figure 5.21: Two t-SNE space images, containing failure labels in the left part, and GMM clustering labels in the right part.



Number clusters	1) Force increase		2) Force anticipa-tion		3) Power increase		4) Switch forces	
	prec	rec	prec	rec	prec	rec	prec	rec
outliers	0.40	0.72	0.20	0.72	0.41	0.74	0.00	0.00
1	0.25	0.09	0.00	0.00	**0.75**	0.26	0.00	0.00
2	0.00	0.00	**1.00**	0.28	0.00	0.00	0.00	0.00
3	**1.00**	0.19	0.00	0.00	0.00	0.00	0.00	0.00
4	0.00	0.00	0.00	0.00	0.00	0.00	**1.00**	1.00

Figure 5.22: Diagnosis techniques used to visualise, project and cluster strokes into different failure types.

This t-SNE diagnosis has clearly differentiated the four clusters except for several increase forces (25% and 50%) strokes that have been assigned to power consumption failure, which are still too close to the healthy data for the clustering algorithm to differentiate between them. This technique has the additional advantage of enabling results visualisation. However, given its difficulty for hyperparameter tuning without the information of number of clusters and failure labels, it can be hard to implement with semi-supervised models. Another disadvantage of t-SNE is its possibility to create non-existent patterns on data given its adaptation to it.

To complement t-SNE experiments, another clustering algorithm has been applied to latent space data after PCA, aiming at creating clusters that separate different failure types automatically. The OPTICS density clustering algorithm has been selected for

this task, and the minimum number of samples parameter from 20 to 140 was grid has been searched to find the configuration where the algorithm detected 4 clusters. The min num samples parameter that created 4 clusters is 80, and its results are presented in Table 5.9.

Table 5.9: Clustering results using OPTICS algorithm configured with the hyperparameter of minimum number of samples equal to 80, evaluated with precision (prec) and recall (rec) metrics.

Number clusters	**1) Force increase**		**2) Force anticipation**		**3) Power increase**		**4) Switch forces**	
	prec	**rec**	**prec**	**rec**	**prec**	**rec**	**prec**	**rec**
outliers	0.40	0.72	0.20	0.72	0.41	0.74	0.00	0.00
1	0.25	0.09	0.00	0.00	**0.75**	0.26	0.00	0.00
2	0.00	0.00	**1.00**	0.28	0.00	0.00	0.00	0.00
3	**1.00**	0.19	0.00	0.00	0.00	0.00	0.00	0.00
4	0.00	0.00	0.00	0.00	0.00	0.00	**1.00**	1.00

According to Table 5.9, the failure switch forces is correctly isolated given the cluster 4 has a precision and recall of 1 for this failure type. Cluster 3 contains only instances of force increase failure, but not all of them are gathered given the recall is lower than 1; the remaining instances are assigned several to cluster 1 and the rest to outliers group. Similarly, cluster 2 contains only force anticipation failure data, and the rest instances are assigned to outliers group. In addition, cluster 1 contains mainly power increase failure data, but it also contains some force increase instances, being an overlap of two failures; the remaining power increase instances are assigned to outliers group. Finally, the outliers group gathers an important number of instances belonging to force increase, force anticipation and power increase failure types, but none of switch forces failure type.

All in all, there is a small overlap between force increase and power increase failure types. In addition, many force increase, force anticipation and power increase instances are assigned to outliers group. At this point the results of clustering with lower min samples have been analysed, which generate more clusters. This shows that lower magnitude versions of each failure type have higher probability to be assigned together than higher magnitude ones, which are correctly separated.

To continue diagnosis analysis, correct stroke data and the versions of highest magni-

tude for each failure type have been projected to latent space using 2D-CNN-AE. These feature vectors are afterwards z-scaled, and normalized in [0,1] range to have all variables in a comparable distribution and range. This data have been used to fit and be transformed by a SOM, projecting 32 input neurons to a 20x20 feature space map, and configuring its hyperparameters as: sigma=5, learning_rate=0.5, neighborhood function = bubble and random training of 6000. This map projects similar instances on the original space to neurons that are near in the new space, which are represented in light colors. It also projects different instances to different groups of neurons in the new space, being separated with high distance neurons that are represented with dark colors.

The result of SOM is exhibited in Figure 5.23, which shows a clear separation among all failure types, where instances of the same class belong to near and light color neurons, and at the same time are separated with instances of other classes with high distance dark color neurons. In addition, normal data is separated from failures by high distance dark neurons, but at the same time. Few outlier samples are located in big distance SOM cells; these belong to outlier strokes that were previously identified as outliers in reconstruction error damage indexes.

The last diagnosis tool has been designed to facilitate fault diagnosis for domain technicians is based on XAI, given that deep learning models are not explanatory by themselves. A final layer has been added to the 2D-CNN-AE model of Figure 5.18 to calculate the RMSE between reconstructed and input data, using the Equation 5.3. In the equation, n is the number of cycles in the stroke, i indicates a cycle index, m indicates total number of features of the stroke, and j indicates feature index.

$$\begin{aligned} mean\_squared\_loss\_of\_feat_j &= \frac{\sum_{i=1}^{n}(y_{ij} - \hat{y_{ij}})^2}{n}; \\ loss &= \sqrt{\frac{\sum_{j=1}^{m} mean\_squared\_loss\_of\_feat_j}{m}} \end{aligned} \tag{5.3}$$

SHAP libraries' GradientExplainer class has been fitted with samples of correct validation strokes and their losses, so that the explainer learns which is data normality. Then, this explainer has been used to diagnostic anomalous strokes categorised by the anomaly detection model. Thus, the explainer propagates each stroke's loss gradient along all

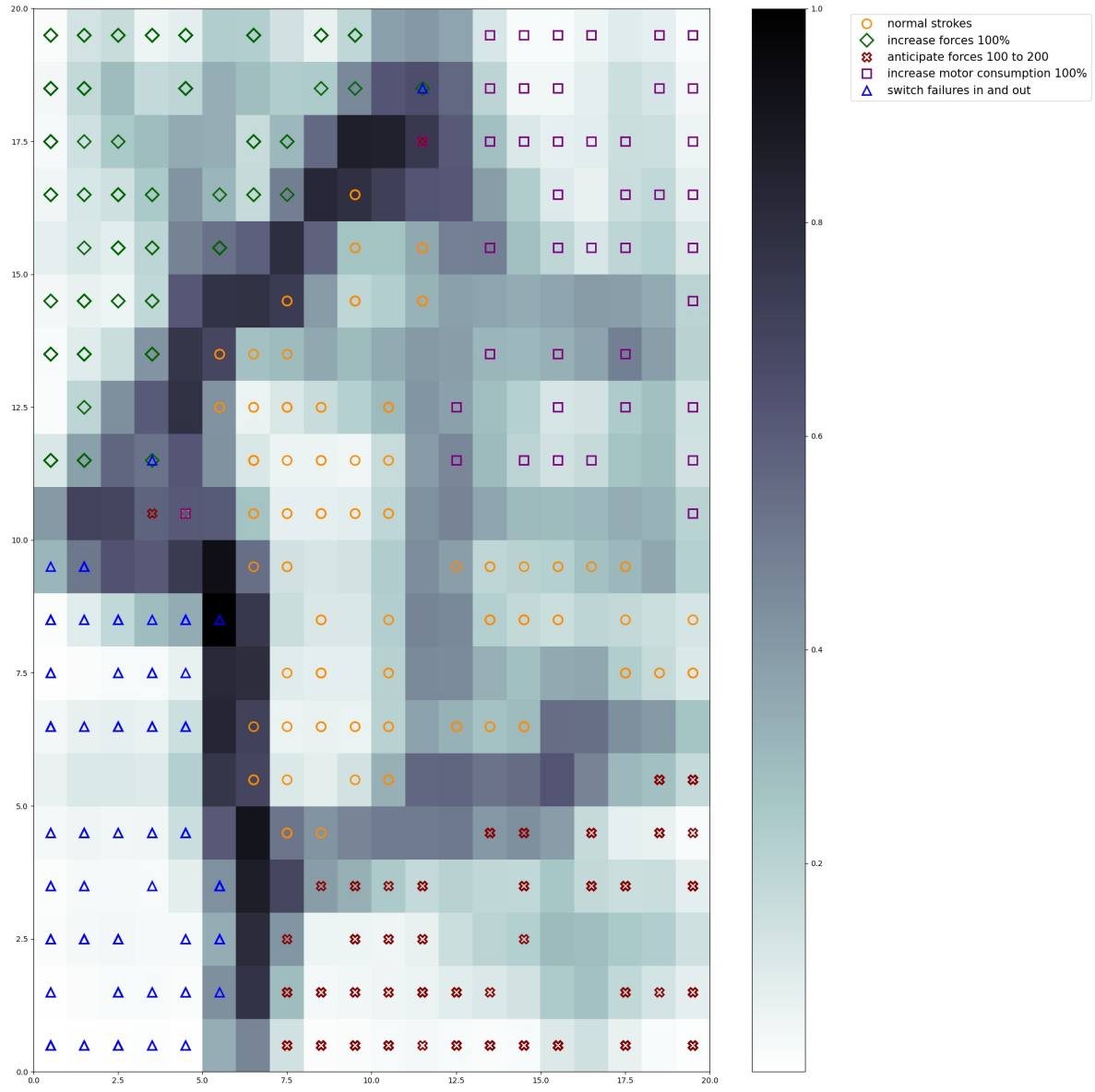


Figure 5.23: SOM of correct data and main failure types, represented by different forms and colors in a 20x20 grid.

layers of the autoencoder until reaching the its input, where SHAP values of each input feature for each cycle data point are estimated. Afterwards, the absolute value of this SHAP value matrix are calculated, the maximum value of this new matrix is searched, and all its elements are divided by the maximum to obtain a matrix of values between 0 and 1. Finally, this last matrix is used to plot an indicator of damage with original input data by drawing red rectangles for each cycle data point, whose transparency is inversely

proportional to matrix values. Thus, a matrix value near to 1 has little transparency and will be clear, whereas values near to 0 will be hardly noticeable.

As Figure 5.24 shows, the developed diagnosis algorithm based on XAI is able to detect which signals are causing the anomaly in a multivariate approach, and without being previously trained for these failure types. The image shows original stroke data in green, stroke data modified with one specified failure type in red, and the background is shadowed in red with the explanation metric presented in previous paragraph. This demonstrates that the algorithm is capable of detecting which features and concretely in which cycle points the failure data is not normal. This tool will be used by domain technicians to isolate anomalies detected semi-supervisedly.

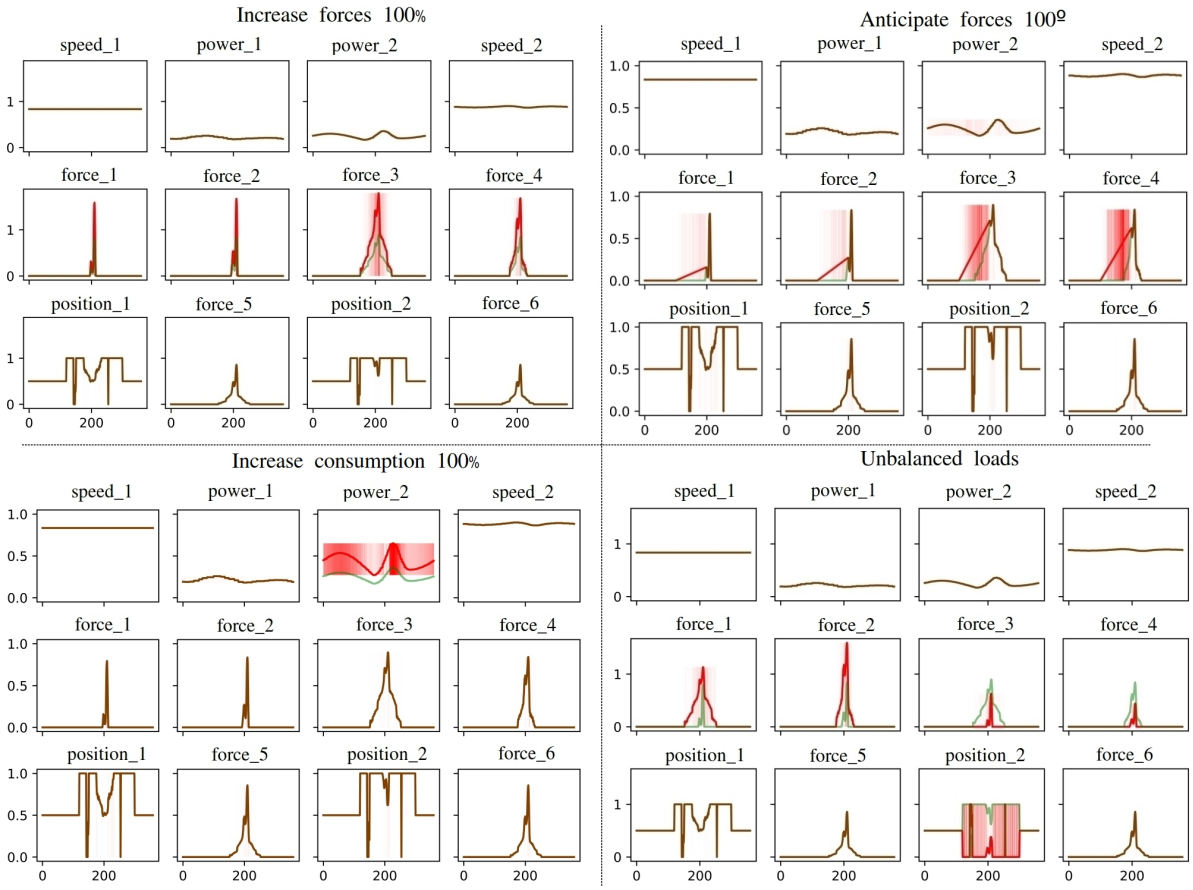


Figure 5.24: Diagnosis using XAI on the same stroke with 4 failure types, shadowing feature's cycle data that cause them in red colour. The green and red signals correspond to a normal stroke and its synthetic failure indicated in the subtitle, respectively.

This tool has also been used to diagnose several outliers detected in training data by the

anomaly detection algorithm, aiming to analyse their root cause. Figure 5.25 segments anomalous training points, and analyses one of these with the XAI-based diagnosis tool. The main difference of this outlier with respect to normal points is the increase of power consumption. This difference is clearly identified by the tool.

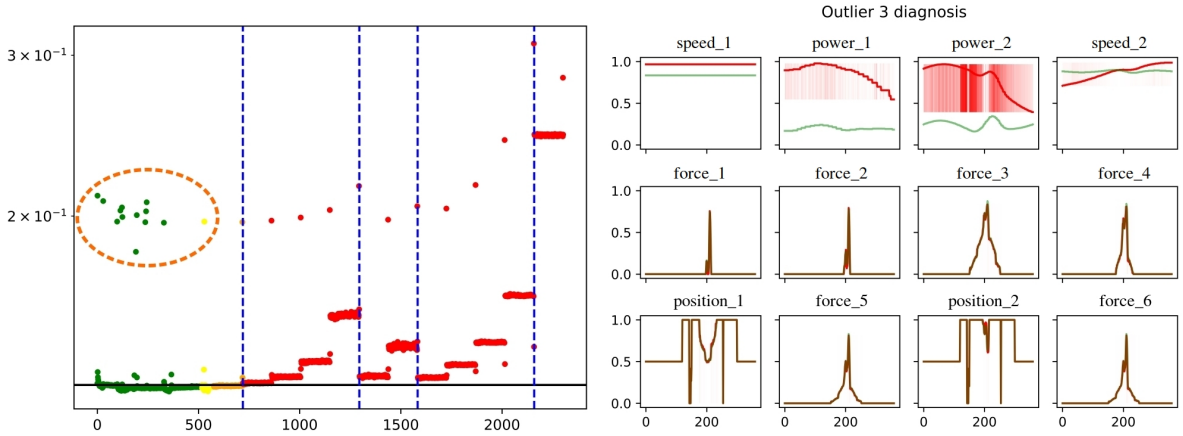


Figure 5.25: Right segment contains diagnosis of outliers in training data using XAI. Left segment contains predictions of 2D-CNN-AE on train, validation, test and synthetic failure data, and training outliers are outlined in a dashed circle. The green and red signals correspond to a normal stroke and an outlier stroke, respectively.

5.3.3.3 Adaptability

At this point, model's accuracy, novel identifiability, failure isolability, explanation facility, damage index estimation and online data processing capability have been tested. The final requirement for the model is to be reusable, adapting to data changes over time. This requirement has been tested using another data subset collected a week after the subset used for anomaly detection and diagnosis, which belongs to the same die.

Firstly, the 2D-CNN-AE model trained in Section 5.3.3.1 has been loaded and executed in the subset of data of the same die 7 days later, calculating F1 score using correct and all failure types on test data. The model does not work well with new data without being modified, obtaining an average F1 score of 0.77 on failure data. At this point, transfer learning experiments have been executed, expecting the model to adapt to new data given it belonged to similar EOC than data used for training the model. Transfer learning has been applied by only training linear layers of the model while keeping original convolutional layers.

The results showed that only 10 strokes are required for model retraining and 5 strokes for validation to achieve a F1 score of 1 in all failure types. Figure 5.26 contains images of damage indexes on test correct data before and after retraining, showing that all indexes moved below the anomaly threshold after transfer learning. This means that threshold adaptation is not required when internal layers of the autoenconder are retrained for new data of the same die.

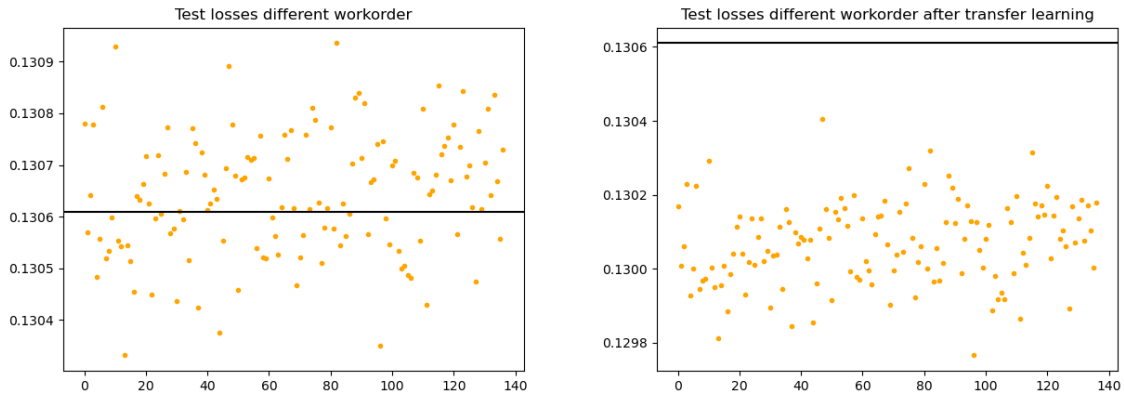


Figure 5.26: Damage indexes of correct test data before (left) and after transfer learning (right) using 2D-CNN-AE model in data of same die one week later. Horizontal black line represents the anomaly detection threshold.

These experiments show that transfer learning enables model reusability for data of the same die along different time periods. Retraining only linear layers of the model requires less data, training resources, and achieves better results than training the whole model from zero.

Finally, the ability of model adaptability to other die using transfer learning has been tested, by executing previous experiments with data of other die. When executing the model without retraining, all damage indexes are far above the anomaly threshold. Afterwards, transfer learning has been performed on linear layers with all training and validation strokes of this new die dataset, given that using a small number for retraining result in fast overfitting. After this transfer learning, damage indexes have been analysed and they are still far beyond the anomaly threshold, which indicate that different dies require different anomaly threshold.

Thus, the anomaly detection threshold has been adjusted in the retrained algorithm, using the techniques and procedure presented in Section 5.3.3.1 to facilitate comparisons.

Percentile 90 and percentile 95 thresholds have been calculated on correct validation data of the new stroke data, and their performance for anomaly detection has been tested using correct and failure test data of the new stroke. This analysis is presented in Figure 5.27, which contains train, validation, test correct and test failure damage indexes before and after transfer learning. None of these thresholds work correctly given that damage indexes on training and validation data are smaller than the ones for test correct data. Therefore, all test samples, either correct and failure, are above the selected thresholds, which is caused by model overfitting that is unable to differentiate anomalous and correct strokes.

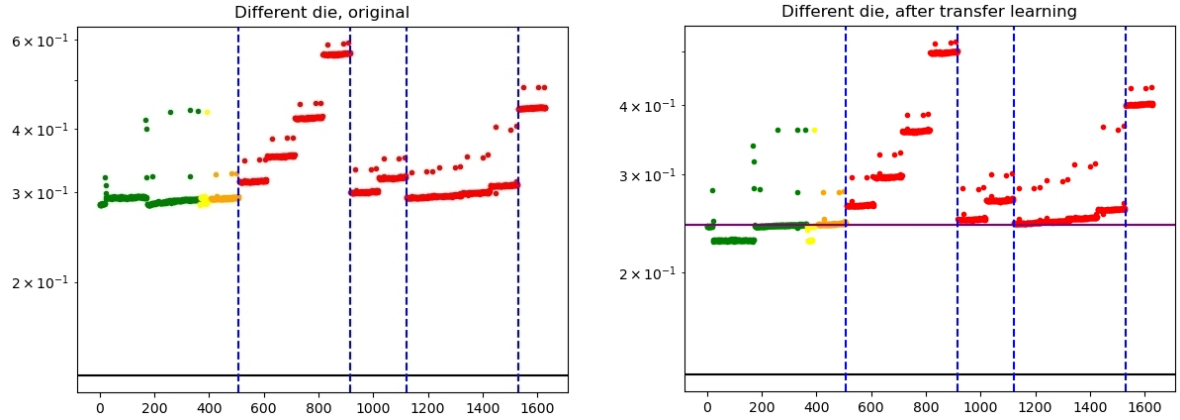


Figure 5.27: Damage indexes of train, validation, test correct and test failure test data before (left) and after transfer learning (right) using 2D-CNN-AE model in data of different die. Correct damage indexes are represented in orange and failure damage indexes in red. Horizontal black line represents the anomaly detection threshold before transfer learning and dashed blue line splits data by each failure type. Horizontal brown and purple lines represent percentile 90 and percentile 95 AD thresholds respectively, but only the purple line is visible given their values are near and they overlap in the figure.

All in all, transfer learning works correctly to adapt the model for data of the same die in different periods of time. However, it does not work to adapt the model trained with data of one die to data of different die. The reason is that data of the same die is collected for the same forming process and under similar operating conditions, but different dies perform different forming processes and have different operating conditions. Therefore, one model for each die should be created, but then they could be reused over time.

5.3.4 Discussion

In this use-case, an accurate semi-supervised deep learning-based anomaly detection model only trained with correct working machine data of production press machine data has been developed. It outperforms statistical semi-supervised anomaly detection models like PCA and null-space, and traditional machine learning models such as OC-SVM and ELM. This deep learning model can be executed online, monitoring the stamping process to identify anomalous strokes and avoid machine failures.

However, one drawback of deep learning models is their difficulty to be interpreted, given they are black box models. Therefore, several diagnosis tools have been developed to facilitate domain technicians in the identification of possible machine failure types once anomalies are detected. For that purpose, the encoder part of the autoencoder has been used to extract stroke features and obtain a 32 feature vector to experiment diagnosis. This data has been used with OPTICS and GMM techniques to cluster failure types, t-SNE for 2D projection and visualisation, and SOM for projection to new space using competitive learning. These techniques have successfully isolated different failure types with latent space features, even though these novel failures were not available for model training.

XAI techniques have also been integrated for diagnosis, demonstrating their ability to detect which signal parts of stroke data are responsible for causing the anomaly. In addition, a visual diagnosis tool based on XAI that highlights damaged signals has been created, with the objective to assist domain technicians in the diagnosis of failure causes based on their expertise.

In addition, this use-case has demonstrated that transfer learning enables model adaptability to data variations of same die in different workorders, which allows reusing models with small adjustments. In contrast, each die requires a model specifically trained for it, since their process and operational conditions are different and therefore, transfer learning does not work.

Following the methodology for data-driven PdM implementation presented in this thesis has facilitated the design and implementation of this use-case. Likewise, it has promoted the collaboration of business, domain technician and data-scientist profiles during the process. One relevant example is the validation of developed models, which has been

supported on domain knowledge by a questionnaire, synthetic failure creation, and interview for results evaluation. The methodology has successfully guided the project, and by adapting its tasks to use-case characteristics, the developed algorithms have correctly addressed use-case requirements.

All in all, this work that combines clustering, visualisation, projection and XAI techniques with a deep learning model designed to meet industrial requirements, has resulted in obtaining a high accurate yet explainable model. Future research will continue with the analysis of machine condition evolution over time and monitoring model performance.

Even though each industrial use-case has its own requirements and data characteristics, the techniques implemented in this work can be reused in other PdM use-cases after adaptations. This work's contributions on semi-supervised anomaly detection, semi-supervised diagnosis, and adaptability with transfer learning can increase stakeholders' confidence on developed models, facilitating the adoption of machine learning and deep learning-based predictive maintenance systems in industrial environments.

Chapter 6

Conclusions and future research lines

This chapter presents the main outcomes achieved on this thesis. Section 6.1 outlines the work accomplished during the realisation of the research work. Moreover, as a result of completing this investigation new future research lines open up, which could be taken up as research projects. These future lines are described in Section 6.2.

6.1 Conclusions

The research of this thesis focused on the design and validation of a modular methodology for data-driven predictive maintenance that integrates domain knowledge to systematise their life-cycle in industrial environments. The proposed methodology is supported on existing data-driven PdM methodologies, standards, and architectures. It is composed of the stages, steps and tasks required to design and develop data-driven PdM systems. The methodology's steps are specific to guide industrial companies, although they are general enough to provide the flexibility to adapt to different use-case requirements.

The methodology has been validated empirically with its implementation on three industrial use-cases: aircraft engine simulation, bushing testbed and press machine of a production line. It has systematised the life-cycle of data-driven PdM on each use-case, adapting to their requirements to address their corresponding industrial needs.

The first use-case turbofan (Section 5.1) validates the methodology by its application to develop semi-supervised data-driven anomaly detection models as part of PdM in the benchmark maintenance dataset. This use-case has validated the business analysis, resources analysis and model development stages of the methodology, but the deployment and monitoring stage was not implemented since the dataset was obtained from a challenge that does not have a running environment. As a result of this use-case, a

CNN autoencoder and OC-SVM algorithms were developed, which achieved a F_2 score of 0.98 in the first dataset and 0.92 and 0.98 F_2 scores in the third dataset, respectively. These algorithms obtain more accurate results than the other statistical and machine learning models developed for this use-case.

The second use-case bushing testbed (Section 5.2) validates the proposed methodology to create supervised RUL models. Several state-of-the-art data-driven algorithms are developed including statistical and traditional machine learning models, but excluding deep learning models to facilitate model explainability. Domain knowledge is used for feature selection and model interpretation supported on explainable artificial intelligence techniques that provided local and global explanations. XAI is also used to select the most relevant features based on data characteristics. Then, the selected model is a random forest regressor that uses the 10 most important features to estimate the remaining useful life of bushing tests, which achieves a RMSE of 61.64 and MAE of 36.29 seconds. Finally, this model is deployed to an industrial PC where it processes the data of bushing testbed, estimates the RUL of the experiment and sends the prediction to the operator's PC for visualisation. Domain technicians consider that this model's accuracy for the target application is suitable considering the average duration of fatigue experiments.

The third use-case, real machine (Section 5.3), validates the methodology on a press machine of a production line, covering all the stages and steps except for the model deployment to production step. This last step will be covered by uploading the model to the cloud platform in the future. In this use-case, a semi-supervised deep learning model based on the autoencoder structure achieved the best results for anomaly detection on a dataset of synthetic failure data created by modifying normal strokes. This two dimensional autoencoder achieved an average $F1$ score of 0.99 on data of the same die and workorder. Further steps of the experimentation consisted of perfoming diagnosis on the detected anomalies based on HITL, combining the anomaly detection model with clustering, projection, visualisation and XAI techniques. These techniques enabled to differentiate failure types semi-supervisedly. Furthermore, the model's adaptability to changes in EOC for data of the same die was successfully implemented using transfer learning, showing the potential of this technique for model reusability.

Finally, the adaptability of the methodology has been validated, demonstrating its robustness to address different industrial use-cases' requirements. Integrating the busi-

ness profile in the PdM life-cycle enabled to create PdM systems aligned with industrial requirements from business perspective. Moreover, adding domain knowledge by integrating HITL in the PdM system life-cycle, combined with clustering, projection and XAI techniques enabled to understand models' individual predictions and general behaviour. This process helps to create data-driven models whose behaviour can be linked to the physical meaning. Therefore, domain technicians can trust more complex data-driven PdM systems, which facilitates the implementation of these models in production environments.

6.2 Future research

This thesis has contributed to the data-driven PdM field, specially by providing a methodology for model life-cycle management supported on domain expertise. There are two additional contributions to the field regarding the integration of model explainability and model adaptability. Even so, there are still open issues that can be addressed as future research lines. The following subjects are outlined as future works, which are related to the proposed methodology and the technological research performed in this thesis.

- Analyse if the data-driven PdM methodology is applicable in other scopes beyond industrial environments. Validating its applicability to other fields such as telecommunications, computing or electric equipment could extend the methodology's scope of application.
- Monitor the evolution of the prognosis model's performance in the bushing testbed, evaluating its adaptability to changes in EOC and new experiments. When its performance declines, model retraining should be designed and implemented.
- As no real failure was available, synthetic failure types used in the third use-case of this thesis were based on domain knowledge. However, simulating the machine's failure types with digital twins may result into failure data that is more similar to real production failures. In addition, finding the limit of magnitude change that each synthetic failure type requires for the AD model to label them anomalous, and adapting model's threshold to learn from novel anomalies will facilitate further research.

- Deploy the press machine's anomaly detection model to analyse its performance in production. The possible alerts should be analysed by a domain technician supported on the diagnosis techniques that were developed in the use-case, to evaluate the anomalies and relate them with their failure causes.
- Evaluate how the transfer learning works on the press machine use-case with data collected on longer periods of time. This could model the wear of the die, and help to define when the model requires retraining or adjustments to continue working correctly.

Bibliography

[1] "Reactive Maintenance Vs Preventive Maintenance Vs Predictive Maintenance," [Accessed 08. Jul. 2021]. [Online]. Available: https://www.assetinfinity.com/blog/reactive-vs-preventive-vs-predictive

[2] "What is Predictive Maintenance? | TIBCO Software," [Accessed 08. Jul. 2021]. [Online]. Available: https://www.tibco.com/reference-center/what-is-predictive-maintenance

[3] scikit-learn 1.0.dev0 documentation, "One-Class SVM versus One-Class SVM using Stochastic Gradient Descent," [Accessed 02. Sep. 2021]. [Online]. Available: https://scikit-learn.org/dev/auto_examples/linear_model/plot_sgdocsvm_vs_ocsvm.html

[4] J. Nikulski, "The Ultimate Guide to AdaBoost, random forests and XGBoost," [Accessed 08. Sep. 2021]. [Online]. Available: https://towardsdatascience.com/the-ultimate-guide-to-adaboost-random-forests-and-xgboost-7f9327061c4f

[5] Angus Turner, "Gaussian Mixture Models in PyTorch," [Accessed 03. Sep. 2021]. [Online]. Available: https://angusturner.github.io/generative_models/2017/11/03/pytorch-gaussian-mixture-model.html

[6] Y. Zhang, Y. Liu, Z. Zhang, and Z. Zhang, "Classification of incomplete data based on evidence theory and an extreme learning machine in wireless sensor networks," *Sensors*, vol. 18, p. 1046, 2018.

[7] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020.

[8] Marco Tulio, "Lime: Explaining the predictions of any machine learning classifier," [Accessed 31. Jul. 2021]. [Online]. Available: https://github.com/marcotcr/lime

[9] "ELI5 0.11.0 documentation," [Accessed 28. Jul. 2021]. [Online]. Available: https://eli5.readthedocs.io/en/latest/

[10] "SHAP documentation," [Accessed 21. Jul. 2021]. [Online]. Available: https://shap.readthedocs.io/en/latest/index.html

[11] UE Systems, "Understanding the P-F curve and its impact on reliability centered maintenance," [Accessed 18. Sep. 2019]. [Online]. Available: http://www.uesystems.com/news/understanding-the-p-f-curve-and-its-impact-on-reliability-centered-maintenance

[12] D. Li, D. Chen, J. Goh, and S.-k. Ng, "Anomaly detection with generative adversarial networks for multivariate time series," *arXiv preprint arXiv:1809.04758*, 2018.

[13] C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, and N. V. Chawla, "A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 1409–1416.

[14] Z. Li, D. Wu, C. Hu, and J. Terpenny, "An ensemble learning-based prognostic approach with degradation-dependent weights for remaining useful life prediction," *Reliability Engineering and System Safety*, vol. 184, pp. 110–122, 2019.

[15] "MIMOSA OSA-CBM - MIMOSA," [Accessed 08. Jul. 2021]. [Online]. Available: https://www.mimosa.org/mimosa-osa-cbm/

[16] S. Khan and T. Yairi, "A review on the application of deep learning in system health management," *Mechanical Systems and Signal Processing*, vol. 107, pp. 241–265, 2018.

[17] "Cross Industry Standard Process for Data Mining - Wikipedia, la enciclopedia libre," [Accessed 08. Jul. 2021]. [Online]. Available: https://es.wikipedia.org/wiki/Cross_Industry_Standard_Process_for_Data_Mining

[18] D. Frederick, J. DeCastro, and J. Litt, "User's guide for the commercial modular

aero-propulsion system simulation (c-mapss)," *NASA Technical Manuscript*, vol. 2007-215026, 01 2007.

[19] J. Olaizola Alberdi, "Soft sensor-based servo press monitoring," Ph.D. dissertation, Mondragon Unibertsitatea. Goi Eskola Politeknikoa, 2020.

[20] D. Lukac, "The fourth ICT-based industrial revolution "industry 4.0" - HMI and the case of CAE/CAD innovation with EPLAN P8," in *2015 23rd Telecommunications Forum, TELFOR 2015*. IEEE, 2016, pp. 835–838.

[21] UNE-EN 13306, "Maintenance. maintenance terminology," Asociación Española de Normalización, Génova, Madrid, Standard, Jul. 2018.

[22] C. Colemen, S. Damodaran, M. Chandramoulin, and E. Deuel, "Making maintenance smarter," *Deloitte University Press*, 2017.

[23] M. Wiseman, "A History of CBM (condition based maintenance)," 2006, [Accessed 17. Sep. 2019]. [Online]. Available: http://www.omdec.com/moxie/Technical/Reliability/a-history-of-cbm.shtml

[24] S. Selcuk, "Predictive maintenance, its implementation and latest trends," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 231, no. 9, pp. 1670–1679, 2017.

[25] A. Prajapati, J. Bechtel, and S. Ganesan, "Condition based maintenance: a survey," *Journal of Quality in Maintenance Engineering*, vol. 18, no. 4, pp. 384–400, 2012.

[26] Vorne, "What Is OEE (Overall Equipment Effectiveness)," [Accessed 18. Sep. 2019]. [Online]. Available: https://www.oee.com/

[27] Y. Lavi, "The Rewards and Challenges of Predictive Maintenance," Jul 2018, [Accessed 05. Jun. 2020]. [Online]. Available: https://www.infoq.com/articles/predictive-maintenance-industrial-iot

[28] B. S. Dhillon, "Engineering maintenance: a modern approach," *cRc press*, pp. 1–224, 2002.

[29] D. Sanger, "Reactive, Preventive & Predictive Maintenance | IVC Technologies,"

[Accessed 16. Sep. 2017]. [Online]. Available: https://ivctechnologies.com/2017/08/29/reactive-preventive-predictive-maintenance/

[30] Dimensional Research, "Artificial Intelligence and Machine Learning Projects Are Obstructed by Data Issues Global Survey of Data Scientists, AI Experts and Stakeholders," no. May, pp. 1–31, 2019.

[31] H. Hotelling, "Analysis of a complex of statistical variables into principal components." *Journal of educational psychology*, vol. 24, no. 6, p. 417, 1933.

[32] M. M. Moya and D. R. Hush, "Network constraints and multi-objective optimization for one-class classification," *Neural Networks*, vol. 9, no. 3, pp. 463–474, 1996.

[33] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[34] T. Chen, T. He, M. Benesty, V. Khotilovich, Y. Tang, H. Cho *et al.*, "Xgboost: extreme gradient boosting," *R package version 0.4-2*, vol. 1, no. 4, pp. 1–4, 2015.

[35] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[36] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors)," *The annals of statistics*, vol. 28, no. 2, pp. 337–407, 2000.

[37] G. J. McLachlan and K. E. Basford, *Mixture models: Inference and applications to clustering*. M. Dekker New York, 1988, vol. 38.

[38] T. L. Bailey, C. Elkan *et al.*, "Fitting a mixture model by expectation maximization to discover motifs in bipolymers," *Proceedings of International Conference on Intelligent Systems for Molecular Biology*, no. 2, pp. 28–36, 1994.

[39] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015.

[40] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural networks*, vol. 4, no. 2, pp. 251–257, 1991.

[41] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[42] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. van der Laak, B. van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," *Medical Image Analysis*, vol. 42, pp. 60–88, 2017.

[43] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M. P. Reyes, M.-L. Shyu, S.-C. Chen, and S. Iyengar, "A survey on deep learning: Algorithms, techniques, and applications," *ACM Computing Surveys (CSUR)*, vol. 51, no. 5, p. 92, 2019.

[44] A. Géron, *Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2017.

[45] Scik-it learn, "Scik-it learn 0.23.2 documentation," Jun 2021, [Accessed 28. Jun. 2021]. [Online]. Available: https://scikit-learn.org/stable/whats_new/v0.23.html

[46] Keras, "Keras API reference," [Accessed 28. Jun. 2021]. [Online]. Available: https://keras.io/api/

[47] Tensorflow, "TensorFlow 2.3 documentation," [Accessed 28. Jun. 2021]. [Online]. Available: https://devdocs.io/tensorflow~2.3/

[48] P. J. Werbos, "Applications of advances in nonlinear sensitivity analysis," in *System Modeling and Optimization*. Springer, 2005, pp. 762–770.

[49] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[50] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

[51] Wikipedia, "Extreme learning machine," [Accessed 03. Aug. 2021]. [Online]. Available: https://en.wikipedia.org/wiki/Extreme_learning_machine

[52] G.-B. Huang, L. Chen, C. K. Siew *et al.*, "Universal approximation using incremental constructive feedforward networks with random hidden nodes," *IEEE Trans. Neural Networks*, vol. 17, no. 4, pp. 879–892, 2006.

[53] A. Robinson and F. Fallside, *The utility driven dynamic error propagation network*. University of Cambridge Department of Engineering Cambridge, 1987.

[54] P. J. Werbos, "Generalization of backpropagation with application to a recurrent gas market model," *Neural Networks*, vol. 1, no. 4, pp. 339–356, 1988.

[55] S. Hochreiter, "Untersuchungen zu dynamischen neuronalen Netzen," Master's thesis, Institut für Informatik, Technische Universität, Munchen, 1991.

[56] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[57] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," in *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 103–111.

[58] D. H. Ballard, "Modular Learning in Neural Networks," in *Aaai*, 1987, pp. 279–284.

[59] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," in *Numerische Mathematik*. Springer, 1970, vol. 14, no. 5, pp. 403–420.

[60] Y. Le Cun and F. Fogelman-Soulié, "Modèles connexionnistes de l'apprentissage," *Intellectica. Revue de l'Association pour la Recherche Cognitive*, vol. 2, no. 1, pp. 114–143, 1987.

[61] A. Makhzani and B. Frey, "K-sparse autoencoders," *arXiv preprint*, 2013.

[62] J. Sun, C. Yan, and J. Wen, "Intelligent bearing fault diagnosis method combining compressed data acquisition and deep learning," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 1, pp. 185–195, 2017.

[63] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, vol. 1, 2014.

[64] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, vol. 3, no. January, 2014, pp. 2672–2680.

[65] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[66] "Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures - Exsilio Blog," [Accessed 14. Jun. 2021]. [Online]. Available: https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/

[67] "Root Mean Squared Error Versus Mean Absolute Error," [Accessed 10. Oct. 2021]. [Online]. Available: https://jmlb.github.io/flashcards/2018/07/01/mae_vs_rmse/

[68] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big data*, vol. 3, no. 1, pp. 1–40, 2016.

[69] D. Gunning, "Explainable artificial intelligence (xai)," *Defense Advanced Research Projects Agency (DARPA)*, [Accessed 20. Oct. 2021]. [Online]. Available: https://www.darpa.mil/program/explainable-artificial-intelligence

[70] K. Amarasinghe, K. Kenney, and M. Manic, "Toward explainable deep neural network based anomaly detection," in *2018 11th International Conference on Human System Interaction (HSI)*. IEEE, 2018, pp. 311–317.

[71] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (xai)," *IEEE Access*, vol. 6, pp. 52 138–52 160, 2018.

[72] W. Samek, *Explainable AI: Interpreting, explaining and visualizing deep learning*. Springer Nature, 2019.

[73] M. T. Ribeiro, S. Singh, and C. Guestrin, """ why should i trust you?""" explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[74] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PloS one*, vol. 10, no. 7, 2015.

[75] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K.-R.

Müller, "Unmasking clever hans predictors and assessing what machines really learn," *Nature communications*, vol. 10, no. 1, pp. 1–8, 2019.

[76] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in neural information processing systems*, 2017, pp. 4765–4774.

[77] L. S. Shapley, *17. A value for n-person games.* Princeton University Press, 2016.

[78] L. Liao and F. Köttig, "A hybrid framework combining data-driven and model-based methods for system remaining useful life prediction," *Applied Soft Computing*, vol. 44, pp. 191–199, 2016.

[79] A. Guzman, "Analisis de modos y efectos de falla para mejorar la disponibilidad operacional en la linea de produccion de gaseosas," Bachelor's Thesis, Universidad Libre, 2014.

[80] R. Gorgin, Y. Luo, and Z. Wu, "Environmental and operational conditions effects on lamb wave based structural health monitoring systems: A review," *Ultrasonics*, vol. 105, p. 106114, 2020.

[81] J. Brownlee, *Introduction to time series forecasting with python: how to prepare data and develop models to predict the future.* Machine Learning Mastery, 2017.

[82] M. R. Mehrjou, N. Mariun, M. H. Marhaban, and N. Misron, "Rotor fault condition monitoring techniques for squirrel-cage induction machine-a review," *Mechanical Systems and Signal Processing*, vol. 25, no. 8, pp. 2827–2848, 2011.

[83] V. Saxena, N. Chowdhury, and S. Devendiran, "Assessment of Gearbox Fault DetectionUsing Vibration Signal Analysis and Acoustic Emission Technique," *IOSR Journal of Mechanical and Civil Engineering*, vol. 7, no. 4, pp. 52–60, 2013.

[84] F. P. G. Márquez, A. M. Tobias, J. M. P. Pérez, and M. Papaelias, "Condition monitoring of wind turbines: Techniques and methods," *Renewable Energy*, vol. 46, pp. 169–178, 2012.

[85] M. Albano, E. Jantunen, G. Papa, and U. Zurutuza, *The MANTIS book: Cyber physical system based proactive collaborative maintenance.* River Publishers, 2018.

[86] F. Civerchia, S. Bocchino, C. Salvadori, E. Rossi, L. Maggiani, and M. Petracca, "Industrial internet of things monitoring solution for advanced predictive main-

tenance applications," *Journal of Industrial Information Integration*, vol. 7, pp. 4–12, 2017.

[87] K. Borodulin, G. Radchenko, A. Shestakov, L. Sokolinsky, A. Tchernykh, and R. Prodan, "Towards digital twins cloud platform: Microservices and computational workflows to rule a smart factory," in *Proceedings of the10th International Conference on Utility and Cloud Computing*. ACM, 2017, pp. 209–210.

[88] P. Aivaliotis, K. Georgoulias, and G. Chryssolouris, "The use of digital twin for predictive maintenance in manufacturing," *International Journal of Computer Integrated Manufacturing*, vol. 32, no. 11, pp. 1067–1080, 2019.

[89] W. Zhang, D. Yang, and H. Wang, "Data-driven methods for predictive maintenance of industrial equipment: A survey," *IEEE Systems Journal*, 2019.

[90] B. Dhillon, "Failure modes and effects analysis-bibliography," *Microelectronics Reliability*, vol. 32, no. 5, pp. 719–731, 1992.

[91] R. C. Bromley and E. Bottomley, "Failure modes, effects and criticality analysis (FMECA)," *IEE Colloquium (Digest)*, no. 52, 1994.

[92] D. Li and J. Gao, "Study and application of reliability-centered maintenance considering radical maintenance," *Journal of Loss Prevention in the Process Industries*, vol. 23, no. 5, pp. 622–629, 2010.

[93] Z. A. Welz, "Integrating Disparate Nuclear Data Sources for Improved Predictive Maintenance Modeling : Maintenance-Based Prognostics for Long-Term Equipment Operation," 2017.

[94] Y. Peng, M. Dong, and M. J. Zuo, "Current status of machine prognostics in condition-based maintenance: a review," *The International Journal of Advanced Manufacturing Technology*, vol. 50, no. 1, pp. 297–313, 2010.

[95] Y. Li, T. Kurfess, and S. Liang, "Stochastic prognostics for rolling element bearings," *Mechanical Systems and Signal Processing*, vol. 14, no. 5, pp. 747–762, 2000.

[96] C. H. Oppenheimer and K. A. Loparo, "Physically based diagnosis and prognosis of cracked rotor shafts," in *Component and Systems Diagnostics, Prognostics, and*

Health Management II, vol. 4733. International Society for Optics and Photonics, 2002, pp. 122–133.

[97] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri, "A review of process fault detection and diagnosis: Part i: Quantitative model-based methods," *Computers & chemical engineering*, vol. 27, no. 3, pp. 293–311, 2003.

[98] O. Blancke, A. Combette, N. Amyot, D. Komljenovic, M. Lévesque, C. Hudon, A. Tahan, and N. Zerhouni, "A predictive maintenance approach for complex equipment based on petri net failure mechanism propagation model," in *Proceedings of the European Conference of the PHM Society*, vol. 4, no. 1, 2018.

[99] L. Li, M. Liu, W. Shen, and G. Cheng, "An expert knowledge-based dynamic maintenance task assignment model using discrete stress–strength interference theory," *Knowledge-Based Systems*, vol. 131, pp. 135–148, 2017.

[100] F. Zhao, Z. Tian, and Y. Zeng, "Uncertainty quantification in gear remaining useful life prediction through an integrated prognostics method," *IEEE Transactions on Reliability*, vol. 62, no. 1, pp. 146–159, 2013.

[101] Z. Li and Y. Wang, *Domain Knowledge in Predictive Maintenance for Water Pipe Failures*. Springer International Publishing, 2018, pp. 437–457.

[102] M. Lebold, K. Reichard, C. S. Byington, and R. Orsagh, "Osa-cbm architecture development with emphasis on xml implementations," in *Maintenance and Reliability Conference (MARCON)*, 2002, pp. 6–8.

[103] E. Zugasti, P. Arrillaga, J. Anduaga, M. A. Arregui, and F. Martínez, "Sensor fault identification on laboratory tower," *Proceedings of the 6th European Workshop - Structural Health Monitoring 2012, EWSHM 2012*, vol. 2, pp. 1093–1100, 2012.

[104] B. Kroll, D. Schaffranek, S. Schriegel, and O. Niggemann, "System modeling based on machine learning for anomaly detection and predictive maintenance in industrial plants," *19th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2014*, 2014.

[105] V. Cerqueira, F. Pinto, C. Sá, and C. Soares, "Combining boosted trees with metafeature engineering for predictive maintenance," in *Advances in Intelligent*

Data Analysis XV, H. Boström, A. Knobbe, C. Soares, and P. Papapetrou, Eds. Springer International Publishing, 2016, pp. 393–397.

[106] C. F. Costa and M. A. Nascimento, "Ida 2016 industrial challenge: Using machine learning for predicting failures," in *Advances in Intelligent Data Analysis XV*, H. Boström, A. Knobbe, C. Soares, and P. Papapetrou, Eds. Springer International Publishing, 2016, pp. 381–386.

[107] L. Perini, "Predictive maintenance for off-road vehicles based on hidden markov models and autoencoders for trend anomaly detection." Ph.D. dissertation, Politecnico di Torino, 2019.

[108] J. Martinez, C. Dennison, and Z. Lian, "Sequence Based Classification for Predictive Maintenance," [Accessed 13. Dec. 2020]. [Online]. Available: https://cs229.stanford.edu/proj2017/final-reports/5238029.pdf

[109] M. Lo, N. Valot, F. Maraninchi, and P. Raymond, "Real-time on-board manycore implementation of a health monitoring system: Lessons learnt," in *9th European Congress on Embedded Real Time Software and Systems (ERTS 2018)*, 2018.

[110] M. Sanayha and P. Vateekul, "Fault detection for circulating water pump using time series forecasting and outlier detection," in *2017 9th International Conference on Knowledge and Smart Technology (KST)*, 2017, pp. 193–198.

[111] M. Pratama, E. Dimla, T. Tjahjowidodo, W. Pedrycz, and E. Lughofer, "Online tool condition monitoring based on parsimonious ensemble+," *IEEE transactions on cybernetics*, 2018.

[112] C. Cernuda, "On the relevance of preprocessing in predictive maintenance for dynamic systems," *Predictive Maintenance in Dynamic Systems: Advanced Methods, Decision Support Tools and Real-World Applications*, pp. 53–92, 2019.

[113] Z. Zhou, J. Zhao, and F. Cao, "A novel approach for fault diagnosis of induction motor with invariant character vectors," *Information Sciences*, vol. 281, pp. 496 – 506, 2014, multimedia Modeling.

[114] A. Garg, V. Vijayaraghavan, K. Tai, P. M. Singru, V. Jain, and N. Krishnakumar, "Model development based on evolutionary framework for condition monitoring of a lathe machine," *Measurement*, vol. 73, pp. 95 – 110, 2015.

[115] T. dos Santos, F. J. T. E. Ferreira, J. M. Pires, and C. Damásio, "Stator winding short-circuit fault diagnosis in induction motors using random forest," in *2017 IEEE International Electric Machines and Drives Conference (IEMDC)*, 2017, pp. 1–8.

[116] M. Canizo, E. Onieva, A. Conde, S. Charramendieta, and S. Trujillo, "Real-time predictive maintenance for wind turbines using big data frameworks," in *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*, 2017, pp. 70–77.

[117] J. Lee, J. Ni, D. Djurdjanovic, H. Qiu, and H. Liao, "Intelligent prognostics tools and e-maintenance," *Computers in Industry*, vol. 57, no. 6, pp. 476–489, 2006.

[118] O. Serradilla, E. Zugasti, C. Cernuda, A. Aranburu, J. Ramirez de Okariz, and U. Zurutuza, "Interpreting remaining useful life estimations combining explainable artificial intelligence and domain knowledge in industrial machinery," in *2020 IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2020*, 2020.

[119] H. Wang, M. J. Bah, and M. Hammad, "Progress in outlier detection techniques: A survey," *IEEE Access*, vol. 7, pp. 107 964–108 000, 2019.

[120] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Processing*, vol. 99, pp. 215–249, 2014.

[121] T. P. Carvalho, F. A. Soares, R. Vita, R. d. P. Francisco, J. P. Basto, and S. G. Alcalá, "A systematic literature review of machine learning methods applied to predictive maintenance," *Computers and Industrial Engineering*, vol. 137, p. 106024, 2019.

[122] G. A. Susto, A. Schirru, S. Pampuri, S. McLoone, and A. Beghi, "Machine learning for predictive maintenance: A multiple classifier approach," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 3, pp. 812–820, 2014.

[123] V. Mathew, T. Toby, V. Singh, B. M. Rao, and M. G. Kumar, "Prediction of remaining useful lifetime (rul) of turbofan engine using machine learning," in *2017 IEEE International Conference on Circuits and Systems (ICCS)*. IEEE, 2017, pp. 306–311.

[124] A. Diez-Olivan, J. A. Pagan, R. Sanz, and B. Sierra, "Data-driven prognostics

using a combination of constrained k-means clustering, fuzzy modeling and lof-based score," *Neurocomputing*, vol. 241, pp. 97 – 107, 2017.

[125] S. Zhai, A. Riess, and G. Reinhart, "Formulation and solution for the predictive maintenance integrated job shop scheduling problem," in *2019 IEEE International Conference on Prognostics and Health Management (ICPHM)*, 2019, pp. 1–8.

[126] C. Okoh, R. Roy, and J. Mehnen, "Predictive maintenance modelling for through-life engineering services," *Procedia CIRP*, vol. 59, pp. 196–201, 2017.

[127] G. K. Durbhaka and B. Selvaraj, "Predictive maintenance for wind turbine diagnostics using vibration signal analysis based on collaborative recommendation approach," in *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2016, pp. 1839–1842.

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

[129] P. Castagliola, G. Celano, and S. Psarakis, "Monitoring the coefficient of variation using ewma charts," *Journal of Quality Technology*, vol. 43, no. 3, pp. 249–265, 2011.

[130] G. A. Susto, A. Beghi, and C. De Luca, "A predictive maintenance system for epitaxy processes based on filtering and prediction techniques," *IEEE Transactions on Semiconductor Manufacturing*, vol. 25, no. 4, pp. 638–649, 2012.

[131] Z. Zhao, F. li Wang, M. xing Jia, and S. Wang, "Predictive maintenance policy based on process data," *Chemometrics and Intelligent Laboratory Systems*, vol. 103, no. 2, pp. 137 – 143, 2010.

[132] J. Wen and H. Gao, "Degradation assessment for the ball screw with variational autoencoder and kernel density estimation," *Advances in Mechanical Engineering*, vol. 10, no. 9.

[133] M. Munir, S. Erkel, A. Dengel, and S. Ahmed, "Pattern-based contextual anomaly detection in hvac systems," in *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, 2017, pp. 1066–1073.

[134] C. M. Able, A. H. Baydush, C. Nguyen, J. Gersh, A. Ndlovu, I. Rebo, J. Booth, M. Perez, B. Sintay, and M. T. Munley, "A model for preemptive maintenance of medical linear accelerators-predictive maintenance," *Radiation Oncology*, vol. 11, no. 1, p. 36, 2016.

[135] Fuzhou Feng, Guoqiang Rao, Pengcheng Jiang, and Aiwei Si, "Research on early fault diagnosis for rolling bearing based on permutation entropy algorithm," in *Proceedings of the IEEE 2012 Prognostics and System Health Management Conference (PHM-2012 Beijing)*, 2012, pp. 1–5.

[136] S. Radhakrishnan and S. Kamarthi, "Complexity-entropy feature plane for gear fault detection," in *2016 IEEE International Conference on Big Data (Big Data)*, 2016, pp. 2057–2061.

[137] C. Carlsson, M. Heikkilä, and J. Mezei, *Fuzzy entropy used for predictive analytics*. Springer International Publishing, 2016, vol. 341, pp. 187–209.

[138] N. Bolbolamiri, M. S. Sanai, and A. Mirabadi, "Time-domain stator current condition monitoring: Analyzing point failures detection by kolmogorov-smirnov (ks) test," *Int. J. Electr. Comput. Energ. Electron. Commun. Eng*, vol. 6, no. 6, pp. 587–592, 2012.

[139] S. Eke, T. Aka-Ngnui, G. Clerc, and I. Fofana, "Characterization of the operating periods of a power transformer by clustering the dissolved gas data," vol. 2017-January, 2017, pp. 298–303.

[140] J. Fernandez-Anakabe, E. Z. Uriguen, and U. Z. Ortega, "An Attribute Oriented Induction based Methodology for Data Driven Predictive Maintenance," *arXiv preprint*, 2019.

[141] Z. Chen, S. Deng, X. Chen, C. Li, R.-V. Sanchez, and H. Qin, "Deep neural networks-based rolling bearing fault diagnosis," *Microelectronics Reliability*, vol. 75, pp. 327–333, 2017.

[142] P. Zhao, M. Kurihara, J. Tanaka, T. Noda, S. Chikuma, and T. Suzuki, "Advanced correlation-based anomaly detection method for predictive maintenance," in *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*. IEEE, 2017, pp. 78–83.

[143] E. Zugasti, M. Iturbe, I. Garitano, and U. Zurutuza, "Null is not always empty: Monitoring the null space for field-level anomaly detection in industrial IoT environments," *2018 Global Internet of Things Summit, GIoTS 2018*, 2018.

[144] W. O. L. Vianna and T. Yoneyama, "Predictive Maintenance Optimization for Aircraft Redundant Systems Subjected to Multiple Wear Profiles," *IEEE Systems Journal*, vol. 12, no. 2, pp. 1170–1181, 2018.

[145] N. Kolokas, T. Vafeiadis, D. Ioannidis, and D. Tzovaras, "Forecasting faults of industrial equipment using machine learning classifiers," in *2018 Innovations in Intelligent Systems and Applications (INISTA)*, 2018, pp. 1–6.

[146] J. Yuan, Y. Wang, and K. Wang, "LSTM based prediction and time-temperature varying rate fusion for hydropower plant anomaly detection: A case study," in *Lecture Notes in Electrical Engineering*, vol. 484. Springer, 2019, pp. 86–94.

[147] R. Jegadeeshwaran and V. Sugumaran, "Fault diagnosis of automobile hydraulic brake system using statistical features and support vector machines," *Mechanical Systems and Signal Processing*, vol. 52-53, pp. 436 – 446, 2015.

[148] M. K. Rad, M. Torabizadeh, and A. Noshadi, "Artificial neural network-based fault diagnostics of an electric motor using vibration monitoring," in *Proceedings 2011 International Conference on Transportation, Mechanical, and Electrical Engineering, TMEE 2011*. IEEE, 2011, pp. 1512–1516.

[149] Y. O. Lee, J. Jo, and J. Hwang, "Application of deep neural network and generative adversarial network to industrial maintenance: A case study of induction motor fault detection," in *2017 IEEE International Conference on Big Data (Big Data)*. IEEE, 2017, pp. 3248–3253.

[150] H. Xu, Y. Feng, J. Chen, Z. Wang, H. Qiao, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, and D. Pei, "Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Applications," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 187–196.

[151] "Plant and Equipment Health Measurement, Assessment and Management," Oct 2019, [Accessed 9. Oct. 2019]. [Online]. Available: https://www.lifetime-reliability.com/cms/machinery-health-measurement

[152] A. Bakar, H. Illias, M. Othman, and H. Mokhlis, "Identification of failure root causes using condition based monitoring data on a 33kv switchgear," *International Journal of Electrical Power & Energy Systems*, vol. 47, pp. 305 – 312, 2013.

[153] T. Boutros and M. Liang, "Detection and diagnosis of bearing and cutting tool faults using hidden markov models," *Mechanical Systems and Signal Processing*, vol. 25, no. 6, pp. 2102–2124, 2011.

[154] H. Cortes, J. Daaboul, J. Le Duigou, and B. Eynard, "Strategic lean management: integration of operational performance indicators for strategic lean management," *IFAC-PapersOnLine*, vol. 49, no. 12, pp. 65–70, 2016.

[155] S. Pradhan, R. Singh, K. Kachru, and S. Narasimhamurthy, "A bayesian network based approach for root-cause-analysis in manufacturing process," in *2007 International Conference on Computational Intelligence and Security (CIS 2007)*. IEEE, 2007, pp. 10–14.

[156] F. Ansari, R. Glawar, and W. Sihn, "Prescriptive maintenance of cpps by integrating multimodal data with dynamic bayesian networks," in *Machine Learning for Cyber Physical Systems*, J. Beyerer, A. Maier, and O. Niggemann, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2020, pp. 1–8.

[157] T. T. Ademujimi, M. P. Brundage, and V. V. Prabhu, "A review of current machine learning techniques used in manufacturing diagnosis," in *Advances in Production Management Systems. The Path to Intelligent, Collaborative and Sustainable Manufacturing*, H. Lödding, R. Riedel, K.-D. Thoben, G. von Cieminski, and D. Kiritsis, Eds. Springer International Publishing, 2017, pp. 407–415.

[158] M. Demetgul, "Fault diagnosis on production systems with support vector machine and decision trees algorithms," *The International Journal of Advanced Manufacturing Technology*, vol. 67, no. 9-12, pp. 2183–2194, 2013.

[159] M. Yuwono, Y. Qin, J. Zhou, Y. Guo, B. G. Celler, and S. W. Su, "Automatic bearing fault diagnosis using particle swarm clustering and hidden markov model," *Engineering Applications of Artificial Intelligence*, vol. 47, pp. 88–100, 2016.

[160] R. Murugan and R. Ramasamy, "Failure analysis of power transformer for effective

maintenance planning in electric utilities," *Engineering Failure Analysis*, vol. 55, pp. 182 – 192, 2015.

[161] P. Adhikari, H. G. Rao, and D.-I. M. Buderath, "Machine learning based data driven diagnostics & prognostics framework for aircraft predictive maintenance," in *10th International Symposium on NDT in Aerospace Dresden, Germany*, 2018.

[162] Z. Wu, H. Luo, Y. Yang, X. Zhu, and X. Qiu, "An unsupervised degradation estimation framework for diagnostics and prognostics in cyber-physical system," in *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, 2018, pp. 784–789.

[163] M. Sharp, T. Sexton, and M. P. Brundage, "Toward semi-autonomous information," in *IFIP International Conference on Advances in Production Management Systems*. Springer, 2017, pp. 425–432.

[164] E. Ramasso, "Investigating computational geometry for failure prognostics." *International Journal of prognostics and health management*, vol. 5, no. 1, p. 005, 2014.

[165] M. Baptista, S. Sankararaman, I. P. de Medeiros, C. Nascimento Jr, H. Prendinger, and E. M. Henriques, "Forecasting fault events for predictive maintenance using data-driven techniques and arma modeling," *Computers & Industrial Engineering*, vol. 115, pp. 41–53, 2018.

[166] Z. Zhang and P. Zhang, "Seeing around the corner: an analytic approach for predictive maintenance using sensor data," *Journal of Management Analytics*, vol. 2, no. 4, pp. 333–350, 2015.

[167] A. Kanawaday and A. Sane, "Machine learning for predictive maintenance of industrial machines using iot sensor data," in *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, 2017, pp. 87–90.

[168] D. Lee and R. Pan, "Evaluating reliability of complex systems for predictive maintenance," in *2016 Industrial and Systems Engineering Research Conference, ISERC 2016*, 2020.

[169] ——, "Predictive maintenance of complex system with multi-level reliability struc-

ture," *International Journal of Production Research*, vol. 55, no. 16, pp. 4785–4801, 2017.

[170] X. Zhang, R. Xu, C. Kwan, S. Y. Liang, Q. Xie, and L. Haynes, "An integrated approach to bearing fault diagnostics and prognostics," in *Proceedings of the 2005, American Control Conference, 2005.*, 2005, pp. 2750–2755.

[171] E. Zio and F. Di Maio, "A data-driven fuzzy approach for predicting the remaining useful life in dynamic failure scenarios of a nuclear system," *Reliability Engineering & System Safety*, vol. 95, no. 1, pp. 49–57, 2010.

[172] C. Gutschi, N. Furian, J. Suschnigg, D. Neubacher, and S. Voessner, "Log-based predictive maintenance in discrete parts manufacturing," *Procedings of Conference on Intelligent Computation in Manufacturing Engineering 2018 (CIRP)*, vol. 79, pp. 528–533, 2019.

[173] H. Liao, W. Zhao, and H. Guo, "Predicting remaining useful life of an individual unit using proportional hazards model and logistic regression model," in *RAMS'06. Annual Reliability and Maintainability Symposium, 2006.* IEEE, 2006, pp. 127–132.

[174] T. Benkedjouh, K. Medjaher, N. Zerhouni, and S. Rechak, "Remaining useful life estimation based on nonlinear feature reduction and support vector regression," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 7, pp. 1751–1760, 2013.

[175] G. S. Babu, P. Zhao, and X. L. Li, "Deep convolutional neural network based regression approach for estimation of remaining useful life," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, S. B. Navathe, W. Wu, S. Shekhar, X. Du, X. S. Wang, and H. Xiong, Eds., vol. 9642. Springer International Publishing, 2016, pp. 214–228.

[176] X.-S. Si, W. Wang, C.-H. Hu, M.-Y. Chen, and D.-H. Zhou, "A wiener-process-based degradation model with a recursive filter algorithm for remaining useful life estimation," *Mechanical Systems and Signal Processing*, vol. 35, no. 1-2, pp. 219–237, 2013.

[177] M. Yuan, Y. Wu, and L. Lin, "Fault diagnosis and remaining useful life estimation of aero engine using lstm neural network," in *2016 IEEE International Conference on Aircraft Utility Systems (AUS)*. IEEE, 2016, pp. 135–140.

[178] X. Liu, F. Feng, and A. Si, "Condition based monitoring, diagnosis and maintenance on operating equipments of a hydraulic generator unit," in *IOP Conference Series: Earth and Environmental Science*, vol. 15, no. 4. IOP Publishing, 2012, p. 042014.

[179] M. T. Yildirim and B. Kurt, "Engine health monitoring in an aircraft by using Levenberg-Marquardt Feedforward Neural Network and Radial Basis Function Network," in *Proceedings of the 2016 International Symposium on INnovations in Intelligent SysTems and Applications, INISTA 2016*. IEEE, 2016, pp. 1–5.

[180] K. F. Al-Raheem and W. Abdul-Karem, "Rolling bearing fault diagnostics using artificial neural networks based on Laplace wavelet analysis," *International Journal of Engineering, Science and Technology*, vol. 2, no. 6, 2011.

[181] L. Liao, W. Jin, and R. Pavel, "Enhanced restricted boltzmann machine with prognosability regularization for prognostics and health assessment," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 11, pp. 7076–7083, 2016.

[182] S. Hwang, J. Jeong, and Y. Kang, "SVM-RBM based predictive maintenance scheme for IoT-enabled smart factory," in *2018 13th International Conference on Digital Information Management, ICDIM 2018*. IEEE, 2018, pp. 162–167.

[183] X. Wang, J. Huang, G. Ren, and D. Wang, "A hydraulic fault diagnosis method based on sliding-window spectrum feature and deep belief network." *Journal of Vibroengineering*, vol. 19, no. 6, 2017.

[184] J. Deutsch and D. He, "Using deep learning-based approach to predict remaining useful life of rotating components," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 1, pp. 11–20, 2017.

[185] K. Peng, R. Jiao, J. Dong, and Y. Pi, "A deep belief network based health indicator construction and remaining useful life prediction using improved particle filter," *Neurocomputing*, vol. 361, pp. 19–28, 2019.

[186] S. Y. Shao, W. J. Sun, R. Q. Yan, P. Wang, and R. X. Gao, "A Deep Learning

Approach for Fault Diagnosis of Induction Motors in Manufacturing," *Chinese Journal of Mechanical Engineering (English Edition)*, vol. 30, no. 6, pp. 1347–1356, 2017.

[187] W. Yang, C. Liu, and D. Jiang, "An unsupervised spatiotemporal graphical modeling approach for wind turbine condition monitoring," *Renewable Energy*, vol. 127, pp. 230–241, 2018.

[188] P. Chemweno, I. Morag, M. Sheikhalishahi, L. Pintelon, P. Muchiri, and J. Wakiru, "Development of a novel methodology for root cause analysis and selection of maintenance strategy for a thermal power plant: A data exploration approach," *Engineering Failure Analysis*, vol. 66, pp. 19–34, 2016.

[189] J. Lacaille, A. Gouby, W. Bense, T. Rabenoro, and M. Abdel-Sayed, "Turbofan engine monitoring with health state identification and remaining useful life anticipation," *International Journal of Condition Monitoring*, vol. 5, no. 2, pp. 8–16, 2015.

[190] K. Prabakaran, S. Kaushik, R. Mouleeshuwarapprabu, and A. B. Singh, "Self-organizing map based fault detection and isolation scheme for pneumatic actuator," *International Journal of Innovation and Applied Studies*, vol. 8, no. 3, p. 1361, 2014.

[191] L. Ren, Y. Sun, J. Cui, and L. Zhang, "Bearing remaining useful life prediction based on deep autoencoder and deep neural networks," *Journal of manufacturing systems*, vol. 48, pp. 71–77, 2018.

[192] H. Ahmed, M. D. Wong, and A. K. Nandi, "Intelligent condition monitoring method for bearing faults from highly compressed measurements using sparse over-complete features," *Mechanical Systems and Signal Processing*, vol. 99, pp. 459–477, 2018.

[193] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*. ACM, 2014, p. 4.

[194] J. P. Cabezas Rodríguez, "Generative adversarial network based model for multi-domain fault diagnosis," Bachelor's Thesis, Universidad de Chile, 2019.

[195] X. Guo, L. Chen, and C. Shen, "Hierarchical adaptive deep convolution neural network and its application to bearing fault diagnosis," *Measurement*, vol. 93, pp. 490–502, 2016.

[196] R. Liu, G. Meng, B. Yang, C. Sun, and X. Chen, "Dislocated time series convolutional neural architecture: An intelligent fault diagnosis approach for electric machine," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1310–1320, 2016.

[197] P. Wang, R. Yan, R. X. Gao *et al.*, "Virtualization and deep recognition for system fault classification," *Journal of Manufacturing Systems*, vol. 44, pp. 310–316, 2017.

[198] X. Li, Q. Ding, and J.-Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Reliability Engineering & System Safety*, vol. 172, pp. 1–11, 2018.

[199] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed, "Deepant: A deep learning approach for unsupervised anomaly detection in time series," *IEEE Access*, vol. 7, pp. 1991–2005, 2018.

[200] P. Li, X. Jia, J. Feng, F. Zhu, M. Miller, L.-Y. Chen, and J. Lee, "A novel scalable method for machine degradation assessment using deep convolutional neural network," *Measurement*, p. 107106, 2019.

[201] D. Bruneo and F. De Vita, "On the use of LSTM networks for predictive maintenance in smart industries," in *Proceedings - 2019 IEEE International Conference on Smart Computing, SMARTCOMP 2019*. IEEE, 2019, pp. 241–248.

[202] O. Aydin and S. Guldamlasioglu, "Using LSTM networks to predict engine condition on large scale data processing framework," in *2017 4th International Conference on Electrical and Electronics Engineering, ICEEE 2017*. IEEE, 2017, pp. 281–285.

[203] K. K. Reddy, S. Sarkar, V. Venugopalan, and M. Giering, "Anomaly detection and fault disambiguation in large flight data: a multi-modal deep auto-encoder approach," in *Annual Conference of the Prognostics and Health Management Society*, 2016.

[204] E. Anderlini, G. Salavasidis, C. A. Harris, P. Wu, A. Lorenzo, A. B. Phillips, and

G. Thomas, "A remote anomaly detection system for slocum underwater gliders," *Ocean Engineering*, vol. 236, p. 109531, 2021.

[205] S. Tao, T. Zhang, J. Yang, X. Wang, and W. Lu, "Bearing fault diagnosis method based on stacked autoencoder and softmax regression," in *2015 34th Chinese Control Conference (CCC)*. IEEE, 2015, pp. 6331–6335.

[206] M. Roy, S. K. Bose, B. Kar, P. K. Gopalakrishnan, and A. Basu, "A stacked autoencoder neural network based automated feature extraction method for anomaly detection in on-line condition monitoring," in *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2018, pp. 1501–1507.

[207] G. S. Galloway, V. M. Catterson, T. Fay, A. Robb, and C. Love, "Diagnosis of Tidal Turbine Vibration Data through Deep Neural Networks," *Proceedings of the Third European Conference of the Prognostics and Health Management Society 2016*, pp. 172–180, 2016.

[208] C. Lu, Z.-Y. Wang, W.-L. Qin, and J. Ma, "Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification," *Signal Processing*, vol. 130, pp. 377–388, 2017.

[209] R. Chen, S. Chen, M. He, D. He, and B. Tang, "Rolling bearing fault severity identification using deep sparse auto-encoder network with noise added sample expansion," *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, vol. 231, no. 6, pp. 666–679, 2017.

[210] M. Xia, T. Li, L. Liu, L. Xu, and C. W. de Silva, "Intelligent fault diagnosis approach with unsupervised feature learning by stacked denoising autoencoder," *IET Science, Measurement and Technology*, vol. 11, no. 6, pp. 687–695, 2017.

[211] S. Lygren, M. Piantanida, and A. Amendola, "Unsupervised, deep learning-based detection of failures in industrial equipments: The future of predictive maintenance," in *Society of Petroleum Engineers - Abu Dhabi International Petroleum Exhibition and Conference 2019, ADIP 2019*. Society of Petroleum Engineers, 2019.

[212] M. Arias Chao, B. T. Adey, and O. Fink, "Implicit supervision for fault detection

and segmentation of emerging fault types with deep variational autoencoders," *Neurocomputing*, vol. 454, pp. 324–338, 2021.

[213] R. Chalapathy, A. K. Menon, and S. Chawla, "Anomaly detection using one-class neural networks," *arXiv preprint arXiv:1802.06360*, 2018.

[214] D. Arifoglu and A. Bouchachia, "Activity Recognition and Abnormal Behaviour Detection with Recurrent Neural Networks," *Procedia Computer Science*, vol. 110, pp. 86–93, 2017.

[215] A. Nanduri and L. Sherry, "Anomaly detection in aircraft data using Recurrent Neural Networks (RNN)," in *ICNS 2016: Securing an Integrated CNS System to Meet Future Challenges*. IEE, 2016, pp. 5C2—-1.

[216] D. You, X. Shen, G. Liu, and G. Wang, "Signal anomaly identification strategy based on bayesian inference for nuclear power machinery," *Mechanical Systems and Signal Processing*, vol. 161, p. 107967, 2021.

[217] F. Xu, W. t. P. Tse, and Y. L. Tse, "Roller bearing fault diagnosis using stacked denoising autoencoder in deep learning and Gath-Geva clustering algorithm without principal component analysis and data label," *Applied Soft Computing Journal*, vol. 73, pp. 898–913, 2018.

[218] C. Aytekin, X. Ni, F. Cricri, and E. Aksu, "Clustering and Unsupervised Anomaly Detection with l2 Normalized Deep Auto-Encoder Representations," in *Proceedings of the International Joint Conference on Neural Networks*, vol. 2018-July. IEEE, 2018, pp. 1–6.

[219] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding Gaussian mixture model for unsupervised anomaly detection," *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 2018.

[220] T. Amarbayasgalan, B. Jargalsaikhan, and K. H. Ryu, "Unsupervised novelty detection using deep autoencoders with density based clustering," *Applied Sciences (Switzerland)*, vol. 8, no. 9, p. 1468, 2018.

[221] Z. Li, H. Fang, M. Huang, Y. Wei, and L. Zhang, "Data-driven bearing fault iden-

tification using improved hidden Markov model and self-organizing map," *Computers and Industrial Engineering*, vol. 116, pp. 37–46, 2018.

[222] R. Rustum and S. Forrest, "Fault Detection in the Activated Sludge Process using the Kohonen Self-Organising Map," in *8th International Conference on Urban Planning, Architecture, Civil and Environment Engineering. Dubai, UAE*, 2018.

[223] S. Schwartz, J. J. Montero Jimenez, M. Salaün, and R. Vingerhoeds, "A fault mode identification methodology based on self-organizing map," *Neural Computing and Applications*, pp. 1–19, 2020.

[224] L. Hao, X. Xin, W. Xiaojing, G. Jiayu, and S. Jiexi, "Health Assessment of Rolling Bearing based on Self-organizing Map and Restricted Boltzmann Machine," *Journal of Mechanical Transmission*, no. 6, p. 5, 2017.

[225] Q. Niu, "Remaining useful life prediction of bearings based on health index recurrent neural network," *Boletin Tecnico/Technical Bulletin*, vol. 55, no. 16, pp. 585–590, 2017.

[226] B. Zhang, S. Zhang, and W. Li, "Bearing performance degradation assessment using long short-term memory recurrent network," *Computers in Industry*, vol. 106, pp. 14–29, 2019.

[227] J. C. Chen, T.-L. Chen, W.-J. Liu, C. Cheng, and M.-G. Li, "Combining empirical mode decomposition and deep recurrent neural networks for predictive maintenance of lithium-ion battery," *Advanced Engineering Informatics*, vol. 50, p. 101405, 2021.

[228] Y. Zhang, R. Xiong, H. He, and M. G. Pecht, "Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 7, pp. 5695–5705, 2018.

[229] J. Wang, Y. Liang, Y. Zheng, R. X. Gao, and F. Zhang, "An integrated fault diagnosis and prognosis approach for predictive maintenance of wind turbine bearing with limited samples," *Renewable Energy*, vol. 145, pp. 642–650, 2020.

[230] M. Kraus and S. Feuerriegel, "Forecasting remaining useful life: Interpretable deep learning approach via variational Bayesian inferences," *Decision Support Systems*, vol. 125, 2019.

[231] P. D. Paraschos, G. K. Koulinas, and D. E. Koulouriotis, "Reinforcement learning for combined production-maintenance and quality control of a manufacturing system with deterioration failures," *Journal of Manufacturing Systems*, vol. 56, pp. 470–483, 2020.

[232] R. Rocchetta, L. Bellani, M. Compare, E. Zio, and E. Patelli, "A reinforcement learning framework for optimal operation and maintenance of power grids," *Applied Energy*, vol. 241, pp. 291–301, 2019.

[233] K. S. H. Ong, D. Niyato, and C. Yuen, "Predictive maintenance for edge-based sensor networks: A deep reinforcement learning approach," in *2020 IEEE 6th World Forum on Internet of Things (WF-IoT)*. IEEE, 2020, pp. 1–6.

[234] G. Michau, Y. Hu, T. Palmé, and O. Fink, "Feature learning for fault detection in high-dimensional condition monitoring signals," *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, vol. 234, no. 1, pp. 104–115, 2020.

[235] S. K. Bose, B. Kar, M. Roy, P. K. Gopalakrishnan, and A. Basu, "Adepos: anomaly detection based power saving for predictive maintenance using edge computing," in *Proceedings of the 24th Asia and South Pacific Design Automation Conference*. ACM, 2019, pp. 597–602.

[236] D. Cabrera, F. Sancho, C. Li, M. Cerrada, R.-V. Sánchez, F. Pacheco, and J. V. de Oliveira, "Automatic feature extraction of time-series applied to fault severity assessment of helical gearbox in stationary and non-stationary speed operation," *Applied Soft Computing*, vol. 58, pp. 53–64, 2017.

[237] Y. Huang, C.-H. Chen, and C.-J. Huang, "Motor fault detection and feature extraction using rnn-based variational autoencoder," *IEEE Access*, 2019.

[238] P. Malhotra, V. TV, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Multi-sensor prognostics using an unsupervised health index based on lstm encoder-decoder," *arXiv preprint arXiv:1608.06154*, 2016.

[239] J. Pereira, "Unsupervised anomaly detection in time series data using deep learning," Master's thesis, Instituto Superior Técnico (IST), University of Lisbon, 2018.

[240] B. Lindemann, F. Fesenmayr, N. Jazdi, and M. Weyrich, "Anomaly detection in

discrete manufacturing using self-learning approaches," *Procedia CIRP*, vol. 79, pp. 313–318, 2019.

[241] A. S. Yoon, T. Lee, Y. Lim, D. Jung, P. Kang, D. Kim, K. Park, and Y. Choi, "Semi-supervised learning with deep generative models for asset failure prediction," *arXiv preprint arXiv:1709.00845*, 2017.

[242] C. Zhang and Y. Chen, "Time series anomaly detection with variational autoencoders," *arXiv preprint arXiv:1907.01702*, 2019.

[243] H. Oh, J. H. Jung, B. C. Jeon, and B. D. Youn, "Scalable and unsupervised feature engineering using vibration-imaging and deep learning for rotor system diagnosis," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 4, pp. 3539–3549, 2017.

[244] A. Elsheikh, S. Yacout, and M.-S. Ouali, "Bidirectional handshaking lstm for remaining useful life prediction," *Neurocomputing*, vol. 323, pp. 148–156, 2019.

[245] H. Shao, H. Jiang, H. Zhang, and T. Liang, "Electric locomotive bearing fault diagnosis using a novel convolutional deep belief network," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 3, pp. 2727–2736, 2017.

[246] R. Zhao, R. Yan, J. Wang, and K. Mao, "Learning to monitor machine health with convolutional bi-directional lstm networks," *Sensors*, vol. 17, no. 2, p. 273, 2017.

[247] R. Zhao, D. Wang, R. Yan, K. Mao, F. Shen, and J. Wang, "Machine health monitoring using local feature-based gated recurrent unit networks," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 2, pp. 1539–1548, 2017.

[248] H. Shao, H. Jiang, H. Zhao, and F. Wang, "A novel deep autoencoder feature learning method for rotating machinery fault diagnosis," *Mechanical Systems and Signal Processing*, vol. 95, pp. 187–204, 2017.

[249] B. Lu, J. Stuber, and T. F. Edgar, "Data-driven adaptive multiple model system utilizing growing self-organizing maps," *Journal of Process Control*, vol. 67, pp. 56–68, 2018.

[250] T. Guo, Z. Xu, X. Yao, H. Chen, K. Aberer, and K. Funaya, "Robust online time series prediction with recurrent neural networks," in *Proceedings - 3rd IEEE*

International Conference on Data Science and Advanced Analytics, DSAA 2016. IEEE, 2016, pp. 816–825.

[251] E. Lejon, P. Kyösti, and J. Lindström, "Machine learning for detection of anomalies in press-hardening: Selection of efficient methods," *Procedia CIRP*, vol. 72, pp. 1079–1083, 2018.

[252] M. Unal, M. Onat, M. Demetgul, and H. Kucuk, "Fault diagnosis of rolling bearings using a genetic algorithm optimized neural network," *Measurement: Journal of the International Measurement Confederation*, vol. 58, pp. 187–196, 2014.

[253] L. Wen, L. Gao, and X. Li, "A new deep transfer learning based on sparse auto-encoder for fault diagnosis," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 1, pp. 136–144, 2017.

[254] T. Wen and R. Keyes, "Time series anomaly detection using convolutional neural networks and transfer learning," *arXiv preprint arXiv:1905.13628*, 2019.

[255] G. Martinez Arellano and S. Ratchev, "Towards an active learning approach to tool condition monitoring with bayesian deep learning," *Nottingham repository*, 2019.

[256] D. Kateris, D. Moshou, X.-E. Pantazi, I. Gravalos, N. Sawalhi, and S. Loutridis, "A machine learning approach for the condition monitoring of rotating machinery," *Journal of Mechanical Science and Technology*, vol. 28, no. 1, pp. 61–71, 2014.

[257] Z. Li, K. Goebel, and D. Wu, "Degradation Modeling and Remaining Useful Life Prediction of Aircraft Engines Using Ensemble Learning," *Journal of Engineering for Gas Turbines and Power*, vol. 141, no. 4, 2019.

[258] H. Shao, H. Jiang, Y. Lin, and X. Li, "A novel method for intelligent fault diagnosis of rolling bearings using ensemble deep auto-encoders," *Mechanical Systems and Signal Processing*, vol. 102, pp. 278–297, 2018.

[259] P. S. Mashhadi, S. Nowaczyk, and S. Pashami, "Stacked ensemble of recurrent neural networks for predicting turbocharger remaining useful life," *Applied Sciences (Switzerland)*, vol. 10, no. 1, 2020.

[260] C. Zhang, C. Gupta, A. Farahat, K. Ristovski, and D. Ghosh, "Equipment health

indicator learning using deep reinforcement learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2018, pp. 488–504.

[261] P. Koprinkova-Hristova, "Reinforcement Learning for Predictive Maintenance of Industrial Plants," *Information Technologies and Control*, vol. 11, no. 1, pp. 21–28, 2014.

[262] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, "Deep learning and its applications to machine health monitoring," *Mechanical Systems and Signal Processing*, vol. 115, pp. 213–237, 2019.

[263] O. Fink, Q. Wang, M. Svensén, P. Dersin, W.-J. Lee, and M. Ducoffe, "Potential, challenges and future directions for deep learning in prognostics and health management applications," *Engineering Applications of Artificial Intelligence*, vol. 92, p. 103678, 2020.

[264] T. Rieger, S. Regier, I. Stengel, and N. Clarke, "Fast predictive maintenance in Industrial Internet of Things (IIoT) with Deep Learning (DL): A review," in *CEUR Workshop Proceedings*, vol. 2348, 2019, pp. 69–79.

[265] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," *Data Mining and Knowledge Discovery*, vol. 33, no. 4, pp. 917–963, 2019.

[266] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *Pattern Recognition Letters*, vol. 119, pp. 3–11, 2019.

[267] "Prognostics Center - Data Repository," Jul 2018, [Accessed 11. Oct. 2019]. [Online]. Available: https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository

[268] "UCI Machine Learning Repository: Data Sets," [Accessed 11. Oct. 2019]. [Online]. Available: http://archive.ics.uci.edu/ml/datasets.php

[269] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," *2008 International Conference on Prognostics and Health Management, PHM 2008*, 2008.

[270] E. Ramasso and A. Saxena, "Performance benchmarking and analysis of prognostic methods for CMAPSS datasets," *International Journal of Prognostics and Health Management*, vol. 5, no. 2, pp. 1–15, 2014.

[271] C. Zhang, P. Lim, A. K. Qin, and K. C. Tan, "Multiobjective Deep Belief Networks Ensemble for Remaining Useful Life Estimation in Prognostics," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2306–2318, 2017.

[272] S. Zheng, K. Ristovski, A. Farahat, and C. Gupta, "Long Short-Term Memory Network for Remaining Useful Life estimation," in *2017 IEEE International Conference on Prognostics and Health Management, ICPHM 2017*. IEEE, 2017, pp. 88–95.

[273] A. Listou Ellefsen, E. Bjørlykhaug, V. Æsøy, S. Ushakov, and H. Zhang, "Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture," *Reliability Engineering and System Safety*, vol. 183, pp. 240–251, 2019.

[274] P. Kakati, D. Dandotiya, and B. Pal, "Remaining useful life predictions for turbofan engine degradation using online long short-term memory network," *ASME 2019 Gas Turbine India Conference, GTINDIA 2019*, vol. 2, p. 34, 2019.

[275] ISO 17359, "Condition monitoring and diagnostics of machines - general guidelines," International Organization for Standardization, pub-ISO:adr, Standard, 2018.

[276] P. O'Donovan, K. Leahy, K. Bruton, and D. T. O'Sullivan, "An industrial big data pipeline for data-driven analytics maintenance applications in large-scale smart manufacturing facilities," *Journal of Big Data*, vol. 2, no. 1, pp. 1–26, 2015.

[277] A. Rana, A. K. Verma, and A. Srividya, "Maintenance of Large Engineering Systems," in *Current Trends in Reliability, Availability, Maintainability and Safety*, U. Kumar, A. Ahmadi, A. K. Verma, and P. Varde, Eds. Springer International Publishing, 2016, pp. 599–609.

[278] Deloitte Analytics Institute, "Predictive Maintenance: Taking pro-active measures

based on advanced data analytics to predict and avoid machine failure," *Deloitte Consulting GmbH*, 2017.

[279] D. L. Nuñez and M. Borsato, "Ontoprog: An ontology-based model for implementing prognostics health management in mechanical machines," *Advanced Engineering Informatics*, vol. 38, pp. 746–759, 2018.

[280] A. Bousdekis, B. Magoutas, D. Apostolou, and G. Mentzas, "A proactive decision making framework for condition-based maintenance," *Industrial Management & Data Systems*, vol. 115, no. 7, pp. 1225–1250, 2015.

[281] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, and R. Wirth, "CRISP-DM 1.0: Cross Industry Standard Process for Data Mining," *CRISP-DM Consortium*, 2000.

[282] Reliasoft, "Basic Steps of Applying Reliability Centered Maintenance (RCM)," p. Issue 72, 2007, [Accessed 29. Sep. 2020]. [Online]. Available: https://www.weibull.com/hotwire/issue76/relbasics76.htm

[283] M. Mishali and Y. C. Eldar, "Blind multiband signal reconstruction: Compressed sensing for analog signals," *IEEE Transactions on signal processing*, vol. 57, no. 3, pp. 993–1009, 2009.

[284] M. Almaged and J. Hale, "Virtual instruments based approach to vibration monitoring, processing and analysis," *International Journal of Instrumentation and Measurement*, vol. 4, 2019.

[285] O. Serradilla, E. Zugasti, and U. Zurutuza, "Deep learning models for predictive maintenance: a survey, comparison, challenges and prospect," *arXiv preprint arXiv:2010.03207*, 2020.

[286] C. P. Ley and M. E. Orchard, "Simultaneous inference of lithium-ion battery polarising impedance surface and capacity degradation using a hybrid neural adaptive state space model," *Journal of Energy Storage*, vol. 36, p. 102370, 2021.

[287] B. Andersen and T. Fagerhaug, *Root cause analysis: simplified tools and techniques*. ASQ Quality Press, 2006.

[288] "Turbofan - Wikipedia," [Accessed 15. Jul. 2021]. [Online]. Available: https://en.wikipedia.org/wiki/Turbofan

[289] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[290] M. Hejazi and Y. P. Singh, "One-class support vector machines approach to anomaly detection," *Applied Artificial Intelligence*, vol. 27, no. 5, pp. 351–366, 2013.

[291] L. Domingo Colomer, "Deep learning for predictive maintenance of rolling bearings," Master's thesis, Universitat de Barcelona, jun 2020.

[292] S. Zhang, F. Ye, B. Wang, and T. G. Habetler, "Semi-supervised bearing fault diagnosis and classification using variational autoencoder-based deep generative models," *IEEE Sensors Journal*, vol. 21, no. 5, pp. 6476–6486, 2020.

[293] G. Michau and O. Fink, "Transferring complementary operating conditions for anomaly detection," *arXiv e-prints*, pp. arXiv–2008, 2020.

[294] G. R. Garcia, G. Michau, M. Ducoffe, J. S. Gupta, and O. Fink, "Time series to images: Monitoring the condition of industrial assets with deep learning image processing algorithms," *arXiv preprint arXiv:2005.07031*, 2020.

[295] B. Iglewicz and D. C. Hoaglin, *How to detect and handle outliers*. Asq Press, 1993, vol. 16.

[296] National Institute of Standards and Technology (NIST), "NIST/SEMATECH e-Handbook of Statistical Methods," [Accessed 30. Jul. 2021]. [Online]. Available: https://doi.org/10.18434/M32189

[297] O. A. Hassin, "Condition monitoring of journal bearings for predictive maintenance management based on high frequency vibration analysis," Ph.D. dissertation, University of Huddersfield, 2017.

[298] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 8, pp. 1226–1238, 2005.

[299] M. Meire and P. Karsmakers, "Comparison of deep autoencoder architectures for

real-time acoustic based anomaly detection in assets," in *2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, vol. 2. IEEE, 2019, pp. 786–790.

[300] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[301] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: Ordering points to identify the clustering structure," *ACM Sigmod record*, vol. 28, no. 2, pp. 49–60, 1999.

[302] B. G. Lindsay, "Mixture models: theory, geometry and applications," in *NSF-CBMS regional conference series in probability and statistics*. JSTOR, 1995, pp. i–163.

[303] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[304] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 3319–3328.

[305] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "Smoothgrad: removing noise by adding noise," *arXiv preprint arXiv:1706.03825*, 2017.