Alma Mater Studiorum – Università di Bologna

DOTTORATO DI RICERCA IN

Meccanica e Scienze Avanzate dell'Ingegneria (DIMSAI)

Ciclo XXXIV

Settore Concorsuale: 09/B2

Settore Scientifico Disciplinare: ING-IND/17

INTEGRATING MACHINE LEARNING PARADIGMS FOR PREDICTIVE MAINTENANCE IN THE FOURTH INDUSTRIAL REVOLUTION ERA

Integrazione dei paradigmi di apprendimento automatico per l'implementazione della manutenzione predittiva nell'era della quarta rivoluzione industriale

Presentata da: Francesca Calabrese

Coordinatore Dottorato **Supervisore**

Prof. Marco Carricato **Prof. Alberto Regattieri**

Esame finale anno 2022

ABSTRACT

In the last decade, manufacturing companies have been facing two significant challenges that push towards a remarkable transformation of their factories. First, digitalization imposes adopting the technologies that fall into the principles of Industry 4.0 and that allow creating smart, connected, self-aware, and self-predictive factories. Second, the incredible attention on sustainability imposes to evaluate and reduce the impact of the implemented solutions from economic and social points of view. Production processes transform raw materials into finite products, involving considerable investments in materials, machinery, technologies, and people. The main goal of manufacturing companies is to remunerate such investments by increasing productivity while keeping qualitative standards of produced goods and ensuring the safety of working environments. At the center of these conflicting goals, the maintenance of physical assets assumes a critical role.

On the one hand, increasing the reliability and the availability of production systems leads to the minimization of systems' downtimes; on the other hand, the proper system functioning avoids production wastes and potentially catastrophic accidents. Digitalization and new ICT technologies have assumed a relevant role in maintenance strategies. First of all, they allow assessing the health condition of machinery at any point in time. Moreover, they allow predicting the future behavior of machinery so that maintenance interventions and the spare parts supply can be planned, and the useful life of components can be exploited, ideally, until the time instant before their fault.

Predictive Maintenance typically involves four steps, i.e., data acquisition, feature extraction, diagnostics, and prognostics. First, signals like vibrations, currents, acoustic emissions, and others are collected through sensors installed on the most critical components. Then, signal processing techniques are applied to extract relevant and synthetic information revealing the health condition of the monitored components. Finally, in the case of data-driven Predictive Maintenance, Machine Learning algorithms are applied to find the relationships between the extracted parameters and the health condition (diagnostics) and predict the Remaining Useful Life (RUL) of components or systems under analysis. This process highlights the importance of two aspects, i.e., the data collection and the data analysis. Data acquisition is critical because of the high volume of structured and unstructured data generated by equipment in Smart Factories. The data analysis is critical because of the lousy quality of collected data, which are often unprovided with additional context-related information required by typical machine learning approaches.

This dissertation aims to provide insights on Predictive Maintenance goals and tools in Industry 4.0 and propose a novel data acquisition, processing, sharing, and storage framework that addresses typical issues encountered by machine producers and users. In particular, the research elaborates on two research questions that narrow down the set of potential approaches to the problem of data acquisition, processing, and analysis for fault diagnostics in evolving environments. The research activity is developed according to a research framework, where the research questions are addressed by research levers that are explored according to research topics. Each topic requires a specific set of methods and approaches; however, the overarching methodological approach presented in this dissertation includes three fundamental aspects: the maximization of the quality level of input data, the use of Machine Learning methods for data analysis, and the use of case studies deriving from both controlled environments (laboratory) and real-world instances.

Initially, this dissertation depicts the state-of-the-art literature and industrial practice in fault diagnosis in the Industry 4.0 era and identifies the main research trends over the last years. The other two chapters illustrate the research activity. The first focuses on developing a framework for data collection and processing at the edge and data storage in the cloud, and the second narrow down the research focus to diagnostics in evolving environments. The explored research topics lead to theoretical, methodological, and practical contributions highlighted in the last chapter.

SOMMARIO

Negli ultimi decenni le aziende manifatturiere hanno affrontato, e stanno ancora affrontando, due grandi sfide che le spingono verso una profonda trasformazione dei loro impianti di produzione. In primo luogo, la digitalizzazione, che impone l'utilizzo di una serie di tecnologie che possono essere racchiuse nel concetto di Industria 4.0 e che consentono di creare fabbriche intelligenti, connesse, consapevoli e in grado di prevedere il futuro. In secondo luogo, l'attenzione in tema di sostenibilità impone di valutare e ridurre l'impatto delle soluzioni adottate sia dal punto di vista economico che sociale. I processi produttivi trasformano materie prime in prodotti finiti, richiedendo quindi enormi investimenti in materiali, impianti, tecnologie e persone. L'obiettivo principale delle aziende manufatturiere è la remunerazione di tali investimenti, resa possibile dalla massimizzazione della produttività, cercando, al tempo stesso, di mantenere lo standard qualitativo promesso al cliente e assicurando la sicurezza degli ambienti lavorativi. Al centro di questi tre obiettivi in conflitto tra loro, la manutenzione degli impianti assume un ruolo centrale. Da un lato, l'aumento dell'affidabilità e della disponibilità delle macchine porta alla minimizzazione dei fermi impianto dovuti a guasti improvvisi; dall'altro lato, il loro funzionamento corretto evita scarti di produzione e incidenti potenzialmente catastrofici. La digitalizzazione e le nuove tecnologie ICT (Information and Communication Technology) hanno un impatto notevole sulle strategie manutentive. Non solo consentono di conoscere lo stato di salute dell'impianto in qualsiasi momento, ma consentono di predirne il comportamento futuro, in modo che gli interventi manutentivi, così come l'approvvigionamento delle parti di ricambio, possano essere pianificati con un certo anticipo, e la vita utile di componenti e sistemi possa essere sfruttata, idealmente, fino all'istante di tempo immediatamente precedente al guasto.

La Manutenzione Predittiva in genere si articola in quattro step, che sono la raccolta dei segnali, l'estrazione di parametri rilevanti, la diagnostica e la prognostica. Prima, segnali come vibrazioni, correnti, emissioni acustiche, e altri, vengono raccolti tramite opportuni sensori montati su componenti critici. Poi, il segnale grezzo viene processato in modo da estrarre delle grandezze rilevanti e sintetiche che sono in grado di rivelare lo stato di salute del componente o sistema monitorato. Infine, nel caso di approcci *data-driven*, vengono utilizzati algoritmi di apprendimento automatico per individuare le relazioni che sussistono tra le grandezze estratte e la condizione di salute (diagnostica) e predire la vita utile residua. In questo processo, possono essere individuate due aspetti critici: la raccolta dati e l'analisi dati. La raccolta dati è particolarmente ostica data l'elevata

mole di dati, più o meno strutturati, generati da un impianto. La parte di analisi dei dati, invece, si complica a causa della difficoltà nell'ottenere dati di qualità, ovvero corredati di tutte le informazioni al contorno necessarie per l'applicazione degli algoritmi di apprendimento automatico.

L'obiettivo di questa tesi è quello di fornire degli approfondimenti sugli obiettivi e strumenti per la Manutenzione Predittiva nell'era dell'Industria 4.0 e di proporre una metodologia per l'acquisizione, l'analisi, la condivisione e la memorizzazione dei che risponde ai problemi tipici incontrati sia dai produttori che dagli utilizzatori delle macchine. La ricerca si basa sue due *Research Questions*, che focalizzano l'attenzione sull'insieme di possibili approcci ai problemi dell'acquisizione dati e della loro analisi per la diagnostica in ambienti dinamici. In particolare, per rispondere a tali domande, la ricerca presentata approfondisce alcuni argomenti all'interno di una cornice ben definita, esplorati rispetto ad alcune leve. Sebbene ciascun argomento richieda metodi ed approcci specifici, si può definire un quadro metodologico comune, basato su tre aspetti fondamentali: la massimizzazione del livello di qualità dei dati in input, l'utilizzo di algoritmi di apprendimento automatico per l'analisi dei dati, e l'utilizzo di casi studio sia forniti da aziende che condotti in laboratorio.

All'inizio, la tesi illustra lo stato dell'arte dell'attuale letteratura e pratica industriale sul tema della diagnostica nell'era dell'industria 4.0 e identifica le direzioni della ricerca negli ultimi anni. I successivi due capitoli illustrano le attività di ricerca. Il primo si focalizza sullo sviluppo della metodologia per la raccolta e l'analisi dei dati a bordo macchina e la loro memorizzazione in un cloud centrale, mentre il secondo si concentra sulla diagnostica in real-time nei contesti dinamici. I risultati ottenuti, che portano contributi alla ricerca, alla metodologia e supportano le aziende a livello tattico e strategico, saranno illustrati nell'ultimo capitolo.

Integrating Machine Learning Paradigms for Predictive Maintenance in the Fourth Industrial Revolution era

By Francesca Calabrese

Submitted in fulfilment of the requirements for the Degree of

Doctor of Philosophy

University of Bologna

Meccanica e Scienze Avanzate dell’Ingegneria (DIMSAI)

2022

“*There are known knowns; there are things we know we know. We also know there are known unknowns; that is to say, we know there are some things we do not know. But there are also unknown unknowns—the ones we don’t know we don’t know*”

Donald Rumsfeld, US Secretary of Defense

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Almost all physical objects in our lives are subject to degradation or are willing to fail at a certain point in their lives. This statement is particularly true for industrial assets, consisting of several components that perform a determined function until one or more of them do not function properly anymore. When a component or system reaches its end of life, it needs to be repaired or replaced. In industries, the maintenance function is the business unit that deals with the health management of physical assets.

Maintenance is a collection of actions intended to retain an item or restore it to a state where it can perform a required function (Brundage et al. 2019). Therefore, maintenance directly impacts productivity since the failure of a component may cause unplanned production downtimes, whose duration may vary depending on the type of needed actions and the availability of spare parts. Besides, the output quality of a production system may be compromised when an item does not work correctly, and the safety of working environments is also compromised. In other words, maintenance impacts the company's economic, environmental, and social dimensions (Holgado, Macchi, and Evans 2020).

In 2016, maintenance expenditures and preventable losses in discrete manufacturing were estimated at $193.6 billion U.S. dollars (Thomas and Weiss 2021). Lemes and Hvam (2019) divided maintenance costs into eight different families, including production loss, wages, assets, components, institutional, energy, IT, and outsourcing. Maintenance assumes a critical role in the firm's profitability (Alsyouf 2007) (Maletič et al. 2014) too, and from a cost center, it becomes a profit center. From a necessary evil, it becomes a critical aspect in the achievement of the company's goals.

Since maintenance cannot be eliminated, it must be well-managed. Planned downtimes may be less expensive than unplanned interventions. For this reason, reactive maintenance, which is performed after a failure occurs, is replaced by preventive maintenance first and predictive maintenance then (Poór, Ženíšek, and Basl 2019). Preventive maintenance aims to schedule interventions on physical assets depending on statistical parameters extracted from historical databases. The problem of preventive maintenance is twofold: the lack of historical data and the potential loss of useful life without eliminating the possibility of breakdowns during the defined optimal interval between two consecutive interventions. On the contrary, predictive maintenance does not rely on historical data only. Instead, it considers the actual condition of machinery to decide whether and when performing a maintenance intervention.

Predictive Maintenance (PdM) is the process of determining maintenance actions according to regular inspections of equipment's physical parameters, degradation mechanisms, and stressors to correct problems before machine-down occurs (Fraser, Hvolby, and Watanabe 2011). In other words, relevant parameters, like vibrations, acoustic emissions, currents, pressures, temperatures, and others, can be monitored in a continuous or discontinuous way, and the health condition of the system can be assessed at any point in time based on their values. Finally, predictions on future health conditions and the Remaining Useful Life (RUL) of the monitored component or system can be made (Lei et al., 2018). These predictions will be the basis for scheduling maintenance interventions and spare parts supply, and optimizing maintenance actions (de Jonge and Scarf 2020).

Manufacturing companies and equipment manufacturers face two significant changes affecting their business: digitalization and sustainability (Jasiulewicz-Kaczmarek, Legutko, and Kluk 2020). The development of sensor technologies and Industry 4.0 tools and principles offers a remarkable opportunity for health system management. Machinery is equipped with sensors able to collect data at high frequencies; machinery in the same shop floor communicate with each other and are connected to a central server, where the data of machinery installed in other plants can be gathered. The differences between the current Industries and Industry 4.0 are divided into three main lines: components, which become self-aware and self-predictive; machines, which become self-aware, self-predictive, and self-compare; productive systems, which become self-configure, self-maintain, and self-organize (Zonta et al. 2020). Therefore, the type of data and their analysis objectives are the differentiation key of digitalized industries compared to traditional industries.

In this context, considering data science into maintenance management is fundamental. Data science takes place in five different steps: data retrieval, data preparation, exploration data analysis, modeling, and presentation (Sajid et al., 2021). Data are collected from multiple sources and need to be integrated and converted in a structured way; raw data are high in volume and can be redundant or irrelevant. Data science helps discover possible trends by extracting statistical measures; data science helps model the historical data to make reference trends available for future decisions. These steps basically can be associated with the steps of a typical PdM system: data collection (data retrieval), feature extraction (data preparation and EDA), diagnostics, and prognostics (modeling).

In the literature, there are plenty of Machine Learning algorithms applied to PdM of specific components, like rolling bearings, gearboxes, wind turbines, and others. Given the multi-disciplinary nature of the problem, and the high number of models and tools that can be adopted, a literature review covering all topics is not trivial nor helpful. For this reason, this dissertation will analyze the research background and review the literature topic by topic, with the aim to provide general concepts and guidelines for further investigation.

Although the high number of publications, the use of AI in industrial environments is still in its infancy. The large body of existing literature on diagnostics and predictive maintenance focuses on improving the accuracy of ML models. However, the problem with Industrial AI is not the technology but how to produce application value (Lee, 2020). As Lee (2020) states in his book on Industrial AI, the typical elements of AI, i.e., algorithms, Big Data, and computational power, become algorithms, analytics, and computational platform in Industrial AI. Unlike algorithms, analytics refers to modeling the algorithms in specific application scenarios and for specific objectives. Contrary to Internet Big Data[1], Industrial Big Data needs to manage the 3Bs challenges, named Broken data (data integrity), Bad quality data, and Background (context and domain of data). Finally, besides computational power, the computational platform for Industrial AI should also encompass cloud, edge, and fog computing to integrate endpoints with the cloud and integrate distributed systems with centralized ones.

Within the big challenge to fit the AI in industries, there are more specific issues concerning Big Data and Machine Learning for PdM in Industry 4.0 (Dalzochio et al., 2020). Concerning Big Data, the amount of the data and their acquisition are of a significant concern. A large amount of generated data limits the opportunity of real-time monitoring because of latency, bandwidth, and network scalability. Moreover, data acquisition deals with unstructured data, which may have missing values and annotations (contextual data). Concerning ML, the main issues are the data quality (heterogeneity, unbalanced data, lack of run to failure data), the training of large datasets, the dynamic nature of industrial environments, and the lack of a universal model that applies to multiple scenarios. To deal with issues, evolved approaches like deep learning and transfer learning have been proposed. Although the promising results of these tools, how to construct deep learning-based diagnosis models subject to particular issues in the revolution of big data, and how to improve the interpretability of the deep learning-based diagnosis models are worthy of being further investigated (Lei et al. 2020).

Finally, Original Equipment Manufacturers (OEMs), who are interested in a PdM as a post-sale service, show several limitations and requirements, including the difficulty to get the data from their clients because of transferring and storage limited space, the difficulty to get labeled and contextual data, and the difficulty to conduct, in the same time, PdM at a component-level and a system-level (Calabrese et al., 2021).

[1] The 5 V’s of Internet Big Data are volume, velocity, variety, veracity, and value

To summarize the illustrated research background, Figure 1 locates the issues mentioned above into three boxes, corresponding to different categories:

- Technology-related issues (1) concern the Industry 4.0 technologies, like IoT, Big Data, and Machine Learning. They include the complexity of Big Data collection and analysis in IoT environments, the data sharing among different actors (clients and machine manufacturers) under privacy constraints, the quality of the data that will input intelligent algorithms, and their integration into structured databases;
- Domain-specific issues (2) concern the specific domain of application of PdM, i.e., production plants. They include the lack of historical data related to faulty conditions (unbalanced datasets), the lack of operating and environmental data (unlabelled datasets), the dynamic nature of industrial environments, and the economic and temporal constraints that limit the collection of the necessary data and information;
- Methodology-related issues (3) concern the methodology to realize a PdM system. They include the lack of a comprehensive methodology, including component-level and system-level PdM, and a structured information storage and sharing system.

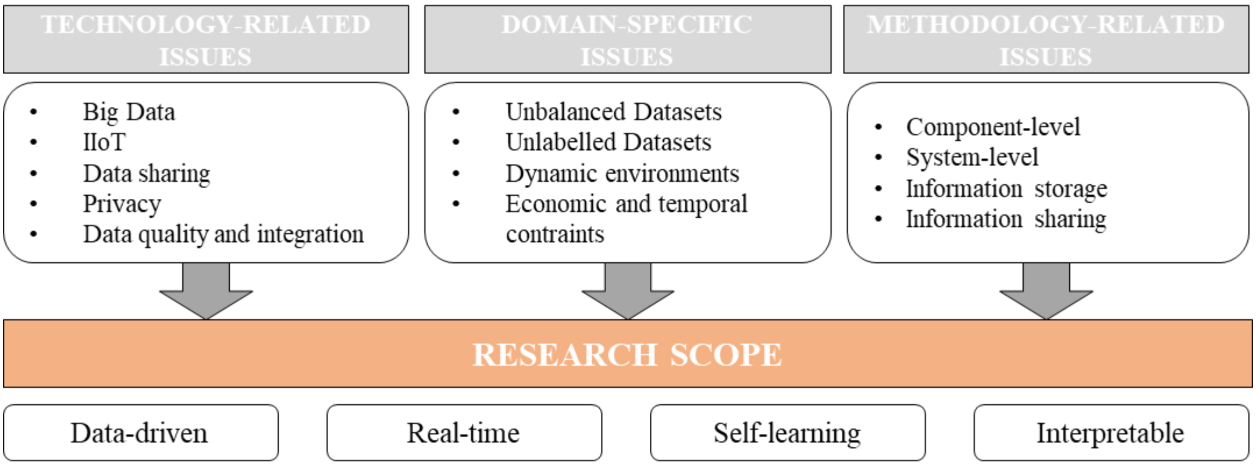


Figure 1 Research background framework

Given the three categories, the research project presented in this dissertation falls within the scope of creating a framework including all elements needed for data collection, sharing, storage, and analysis, that allows industries to monitor and assess the health condition of machinery continuously either in their plants or spread worldwide to their clients' plants. In particular, according to (R. Li, Verhagen, and Curran 2020), the framework includes physical, logical, and functional architectures, which allow implementing a data-driven methodology that (1) provides real-time feedback on the machinery health condition, (2) autonomously recognizes novel behaviors based on historical and current data, (3) gives interpretable results that will feed optimization models for maintenance activities optimization.

Based on these statements, the research is motivated by three research questions discussed in detail in the next section, followed by the research purpose and objectives. Then, the scope and demarcation behind this thesis, the research framework, and the adopted methodology are illustrated.

1.1 Research questions and objectives

This thesis is primarily motivated by the following questions.

> *RQ 1.* How can a Predictive Maintenance system that takes advantage of Industry 4.0 technologies and data-driven solutions be implemented to extract information about the health condition of machinery in real-time, and provide interpretable results to schedule maintenance activities from the double perspective of machinery producers and users?

Such a question is broad and can be approached from a variety of angles and standpoints. In addition, the introduction of the I4.0 concepts and technologies increases the complexity of *RQ 1*, making it a multi-disciplinary problem that requires acting at different levels. In particular, the core of a health management system is transforming raw data into useful information to support the decision-making process. However, data-driven solutions impose challenging issues from two main points of view, mainly derived from the constraints of the current real-world factories. First, the lack of a systematic methodology for choosing the right tools or sources for data collection, and the lack of proper infrastructure for managing the massive data could be potentially available from one or more plants. Hence, *RQ 2* is formulated as follows:

> *RQ 2.* How can industries collect and store enough but essential condition monitoring data from their machinery?

Underpinned by this research question, the first purpose of this thesis is to exploit the experience derived from several collaborations with industries in the Emilia Romagna region and the knowledge acquired from the existing literature for proposing a classification of the available data sources and explaining how each kind of extracted data can contribute to building a health management system. Furthermore, defining how to share the data among different actors, which data need to be shared, and how data have to be organized are also the objects of this dissertation. In other words, this research question acts at the basic level of a predictive maintenance system, which collects the correct data and transforms them into a suitable form to be transferred and analyzed.

> *RQ 3*. How can condition monitoring data be processed and analyzed for real-time detection and identification of machinery faults in evolving environments?

The second issue posed by data-driven solutions is related to the lack of historical data and the dynamic nature of industrial environments. Hence, *RQ 3* requires acting at the data analysis level, which focuses on different approaches and learning paradigms to integrate historical information with streaming data to provide real-time feedback on the machinery health condition for the maintenance actions scheduling.

The second and third research questions can encompass a wide range of related sub-issues. For this reason, this thesis includes a set of research topics that are underpinned by specific objectives. These will be explicated in the following sections.

In addition, a transversal goal of this thesis is to provide knowledge to both scholars and practitioners on the criticalities affecting the realization of a predictive maintenance system for the health management of industrial assets.

1.2 Scope and demarcations

The breadth of the research scope as it was previously introduced makes it essential to delimit the research area by defining the demarcations. The research presented in this dissertation presents limitations related to the hardware technologies, the set of selected topics, and the optimization models for the maintenance decision-making process.

Firstly, research does not intend to discuss in detail the technical characteristics of the physical architecture of the PdM system. This thesis focuses on concepts and principles of Industrial IoT, cloud and edge computing, giving attention to how the different tasks should be distributed among these actors, which kind of data should be stored in the corresponding memories, and when data and information should be transferred. The definition of the connection types and the performance evaluation in terms of required bandwidth, latency, and scalability are left out from this dissertation.

Secondly, the research topics included in this dissertation gravitate around three macro areas only. These are feature extraction and reduction, diagnostics, and novelty detection. Even the approaches and methods to investigate these macro-areas reflect a specific standpoint. This thesis introduces a framework for predictive maintenance in manufacturing and assembling industries, which are still beginning their transformation toward the smart factory. Hence, complicated and innovative approaches are left out from this dissertation. Similarly, the lack of historical industrial data about failure progressions made it hard to perform prognostics and RUL prediction. However, the proposed framework also includes modules for prognostics, for which, best models and solutions will be investigated and validated in the future.

Finally, given the limitations mentioned above, the multi-disciplinary nature of the research topic, and the lack of a complete implementation of the framework in the industrial context, this dissertation does not perform an economic evaluation of results. In general, the estimation of the savings achievable through maintenance optimization is challenging. In addition, in the literature, very few works deal with this issue, and most of them are focused on maintenance strategies optimizations rather than the evaluation of predictive maintenance investments and costs. Finally, since the proposed framework and methodology consider implementing an infrastructure that also affects other business units, a more comprehensive study should have been conducted to assess the economic impact of an IIoT-based Predictive Maintenance system.

The introduced demarcations both identify the research path and expose the limitations of this thesis. Further limitations deal with the specific methodologies applied to study the following research topics. Therefore, they are discussed within each specific chapter.

1.3 Research framework

The research presented in this dissertation has been developed within the framework illustrated in Figure 2, where the research topics and levers corresponding to each research question are identified.

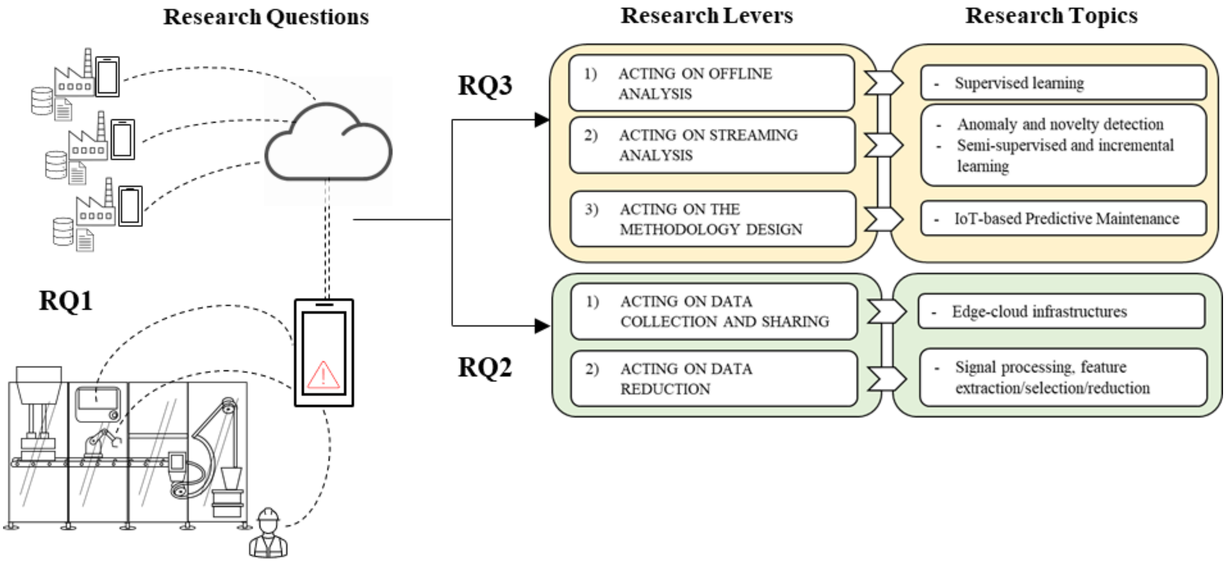

Research Questions
Research Levers
Research Topics
RQ3
RQ1
RQ2
1) ACTING ON OFFLINE ANALYSIS
2) ACTING ON STREAMING ANALYSIS
3) ACTING ON THE METHODOLOGY DESIGN
- Supervised learning
- Anomaly and novelty detection
- Semi-supervised and incremental learning
- IoT-based Predictive Maintenance
1) ACTING ON DATA COLLECTION AND SHARING
2) ACTING ON DATA REDUCTION
- Edge-cloud infrastructures
- Signal processing, feature extraction/selection/reduction

Figure 2 Research Framework

Two research levers are proposed to address *RQ 2*. The first one explores the collection of the data from machinery that are able to provide helpful information about their health status. Hence, it explores the infrastructure for data collection and sharing among different actors to improve the quality and quantity of raw data and the real-time monitoring of assets. In particular, given the demarcations mentioned in the previous section, research is conducted on existing edge-cloud infrastructures, the benefits they bring to the whole plant, and where performing computation and storage. The second research lever deals with reducing the amount of collected data to improve the informative content of the data and facilitate their transmission and storage. This lever explores signal processing and dimensionality reduction techniques that extract or select synthetic and relevant information from raw data.

Three research levers are proposed to address *RQ 3*. The first one deals with using supervised ML approaches for batch analysis to improve the system's diagnostic capability. The second lever deals with the use of unsupervised and incremental learning for streaming data to identify the unknown behaviors of the machine under analysis, face the dynamic nature of the industrial environment, and deal with the different operating conditions of machinery. In other words, the problem of novelty detection in evolving environments is addressed. Finally, the third research lever deals with integrating the outputs of the offline and streaming analyses to provide a methodology to design a general framework for the realization of Predictive Maintenance, including the logical connections between the different functions of the health management system, and addressing defined stakeholders’ requirements.

1.4 Methodology

The methodology used during the research activity and described in this dissertation includes different approaches, models, and research tools according to each specific research topic. In particular, three main pillars can be identified. These will be illustrated with more details in the following.

Firstly, integrating different kinds of input data from physical assets (1) assumes a critical role in addressing the research questions formulated in Section 1.1. In particular, a first distinction can be made between sensor data and other data types, like event data and log data. In the first case, the data are collected from sensors installed on the machinery at high frequencies and generally specific to each component or sub-systems. On the other hand, event and log data are low-frequency data that may contain general information about the maintenance interventions carried out during the operation and the machinery setting parameters (contextual data). The second distinction of input data can be made between historical data and streaming data. Historical data are generally collected from the machinery in several known health conditions and stored in an offline database, from which they will be picked for batch analysis aiming to build predictive models. On the contrary, streaming data are collected during the machinery functioning and are generally transmitted to a cloud or temporarily stored in edge devices. The development of proper tools and architectures for data collection, integration, and storage represents one of the biggest challenges in Smart Factories.

Second, research activity focuses on Machine Learning (ML) and Deep Learning (DL) techniques for analyzing the collected data (2). In particular, the goal is twofold. On the one hand, ML and DL aim to build accurate models on historical data, where accurate stands for capable of well-separating data referring to each known health condition of machinery and classifying them according to the health condition. Then, the so-built models are modified and integrated to make inferences on streaming data to provide real-time feedback on the current health condition of machinery and adapt themselves to novel conditions.

To achieve (2), the use of case studies and proof-of-concept deriving from real-world instances is preferential to validate the proposed diagnostic solutions. The case studies constitute the third pillar (3).

Based on these pillars, several research topics are explored according to a specific protocol that deserves special attention in this chapter. Such protocol is illustrated by the mean of a framework represented in Figure 3. As shown in the figure, the real-world complex system considered in this dissertation consists of machinery (potentially) distributed in different plants located all over the world. Machinery produces a significant amount of data, which are temporarily stored in the edge devices. The protocol suggests reducing data at the edge and gathering and storing such reduced data into a central cloud. Here, collected data can be organized into relational databases to support the decision-making process. For instance, the relevant information can be grouped for component, machinery type, product family, and client.

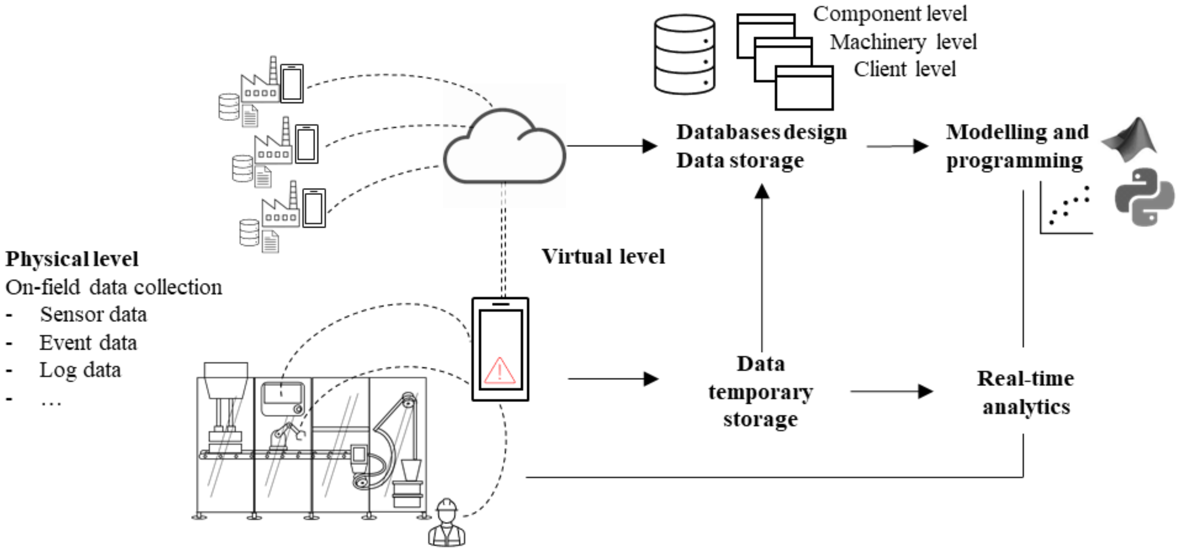
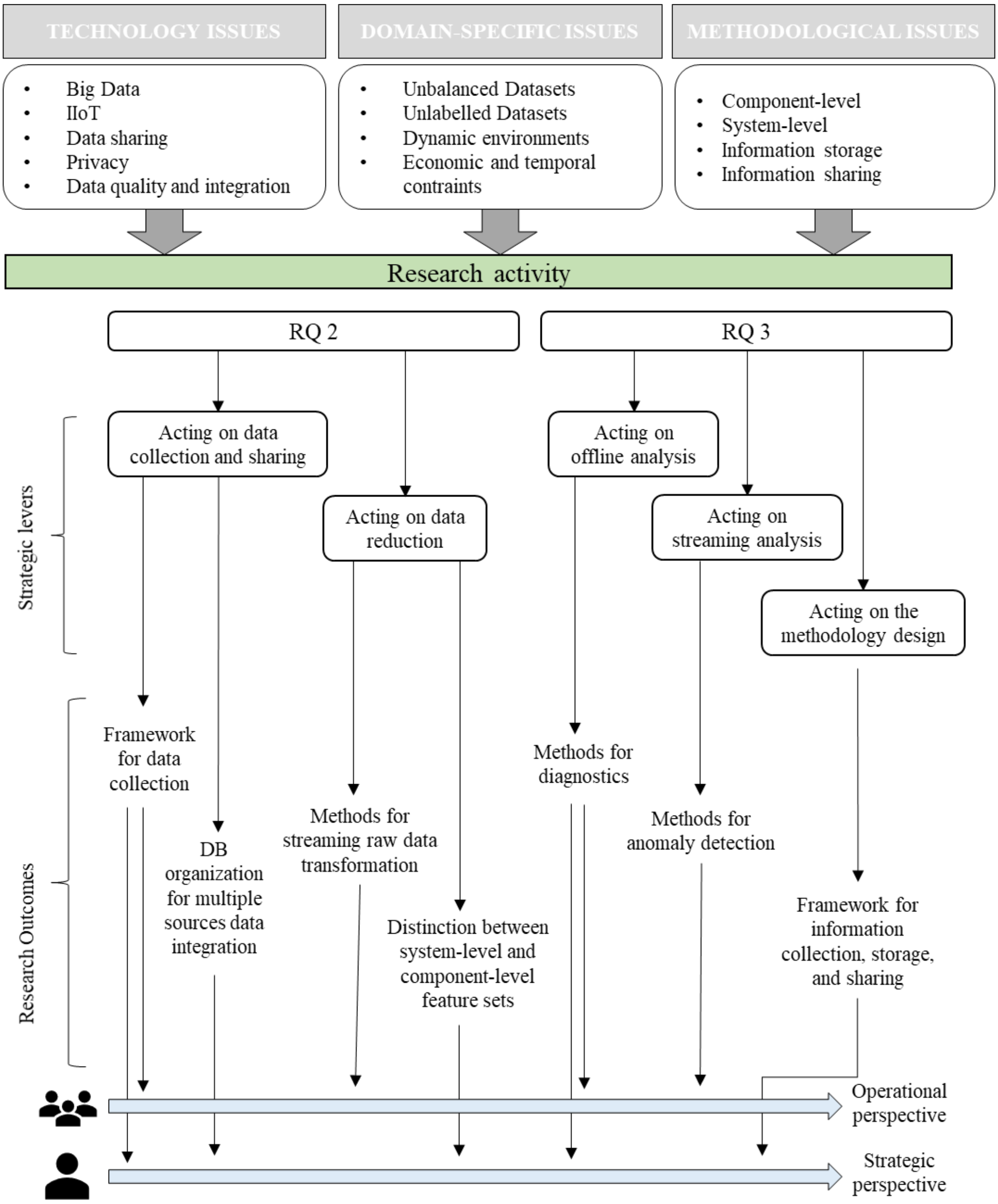


Figure 3 Research Methodology

Data in this form can successively be investigated through models that explain the behavior of the observed system.

1.5 Research activities

The research activity performed with respect to the introduced research levers led to a set of outcomes summarized in Figure 4. These outcomes contribute to the research scope following the two research sub-questions and have been obtained by adopting different perspectives.



TECHNOLOGY ISSUES
DOMAIN-SPECIFIC ISSUES
METHODOLOGICAL ISSUES
Big Data
IIoT
Data sharing
Privacy
Data quality and integration
Unbalanced Datasets
Unlabelled Datasets
Dynamic environments
Economic and temporal contraints
Component-level
System-level
Information storage
Information sharing
Research activity
RQ 2
RQ 3
Strategic levers
Acting on data collection and sharing
Acting on data reduction
Acting on offline analysis
Acting on streaming analysis
Acting on the methodology design
Research Outcomes
Framework for data collection
DB organization for multiple sources data integration
Methods for streaming raw data transformation
Distinction between system-level and component-level feature sets
Methods for diagnostics
Methods for anomaly detection
Framework for information collection, storage, and sharing
Operational perspective
Strategic perspective

Figure 4 Outcomes of the research activity

According to an operational perspective, some outcomes support the performance improvement of the daily information. In contrast, others support the strategic decision-making on how to use the daily data and information to maintain and enhance the remote monitoring of plants in the long term (i.e., strategic perspective). In addition, the two perspectives can be associated with the two main actors interested in the proposed PdM system, i.e., Original Equipment Manufacturers (OEMs) and clients. OEMs assume a more strategic perspective and are interested in the information extracted from raw data rather than receiving continuous data flows. On the contrary, clients assume an operational perspective since they are interested in continuous monitoring and real-time feedback to avoid machinery shutdown. Finally, these two perspectives and actors also require different times of data collection and storage. Hence, computing and storage can be divided between edge and cloud. At the edge, the tasks to achieve the daily clients' objectives can be realized, while the data storage and organization and the models training can be realized into the cloud, which acts as a service for edge computing and provides information for strategic purposes.

The research outcomes generated to address *RQ 2* include the framework for data collection, the DB organization in the cloud for information collection and sharing, the evaluation of different methods for data processing, and the distinction between system-level and component-level features that allow performing different kinds of task. As illustrated in Figure 4, the first outcome impacts both perspectives since the data are collected at the edge, processed, and then sent to the cloud. The third outcome impacts only the operational perspective since data reduction is performed at the edge, while the remaining two outcomes impact only the strategic perspective since OEMs take the most advantage from the contained information.

The research outcomes generated to address *RQ 3* include evaluating several offline and batch diagnostics models, streaming anomaly detection and incremental novelty detection approaches, and their integration into the framework for information collection, storage, and sharing. The first two outcomes impact both perspectives since they are based on models training in the cloud and models inference at the edge. Instead, the last outcome only impacts the strategic perspective since it represents a way to achieve specific goals and can be integrated with the functionalities required by the system owner.

1.6 Thesis outline

This thesis has been developed following the research framework presented in previous sections. The defined research levers and research topics have been arranged in a sequence of chapters, as shown in Figure 5.

Chapter 1 introduces this dissertation by outlining the context investigation, the research questions and objectives, the research framework, and the methodological approach.

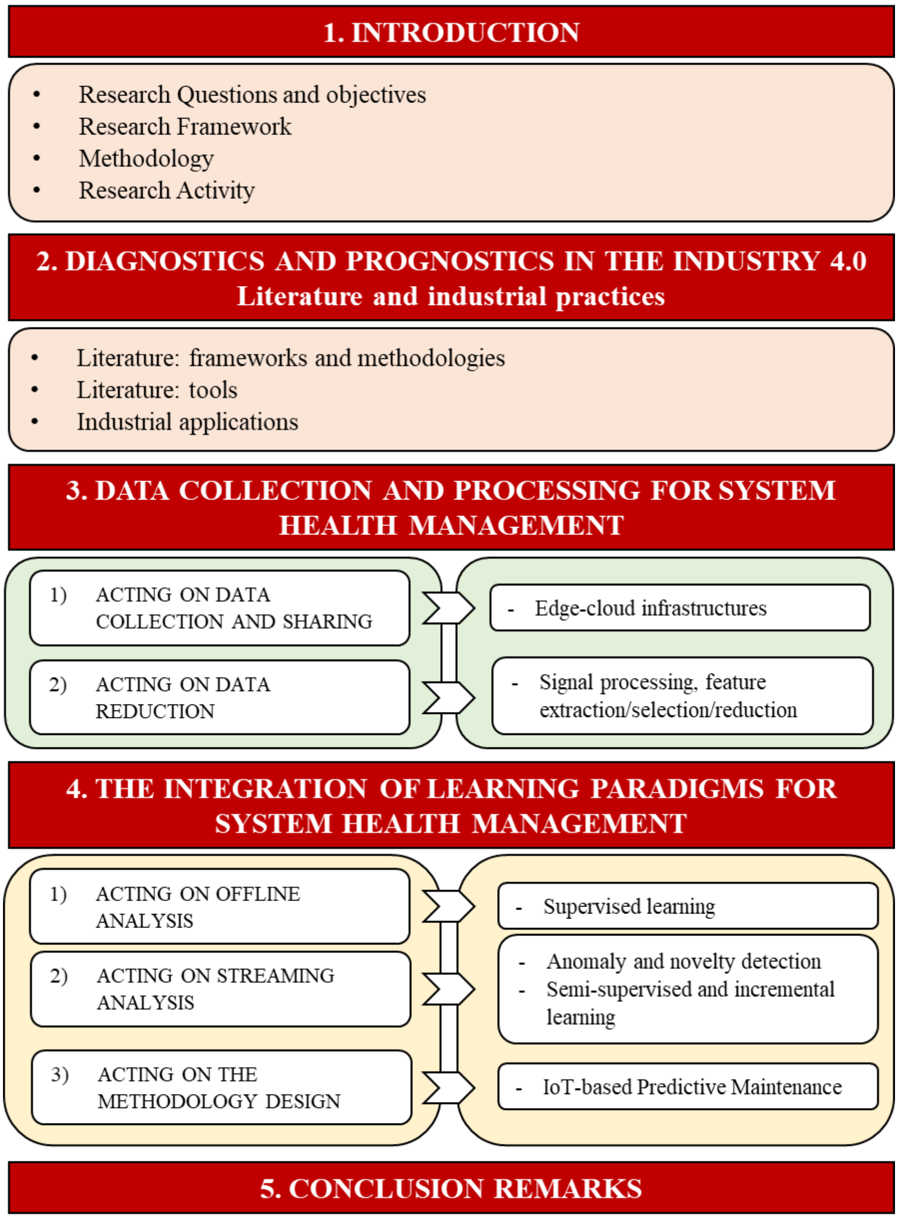

1. INTRODUCTION
Research Questions and objectives
Research Framework
Methodology
Research Activity
2. DIAGNOSTICS AND PROGNOSTICS IN THE INDUSTRY 4.0
Literature and industrial practices
Literature: frameworks and methodologies
Literature: tools
Industrial applications
3. DATA COLLECTION AND PROCESSING FOR SYSTEM HEALTH MANAGEMENT
1) ACTING ON DATA COLLECTION AND SHARING
2) ACTING ON DATA REDUCTION
- Edge-cloud infrastructures
- Signal processing, feature extraction/selection/reduction
4. THE INTEGRATION OF LEARNING PARADIGMS FOR SYSTEM HEALTH MANAGEMENT
1) ACTING ON OFFLINE ANALYSIS
2) ACTING ON STREAMING ANALYSIS
3) ACTING ON THE METHODOLOGY DESIGN
- Supervised learning
- Anomaly and novelty detection
- Semi-supervised and incremental learning
- IoT-based Predictive Maintenance
5. CONCLUSION REMARKS

Figure 5 Thesis outline

Chapter 2 discusses the current industrial practice on maintenance of production systems in the era of Industry 4.0 to provide the reader with the reasons driving the choice of the research topics (see section 2.5). The chapter illustrates a literature review on the problem, facing it from three points of view, i.e., the object (diagnostics and prognostics), the tools (Machine Learning), and the technologies (Industrial IIoT, edge, and cloud computing). Section 2.2 introduces the concept of diagnostics and prognostics, which represent the primary goals of a PdM system. Section 2.3 introduces the main concepts and tasks of Machine Learning, which is the primary tool adopted for data-driven diagnostics and prognostics. Finally, section 2.4 introduces definitions and characteristics of emerging Industry 4.0 technologies, like Industrial Big Data, Industrial Internet of Things, edge computing, and cloud computing.

Chapter 3 addresses *RQ 2* by exploring the two research topics underpinned by two strategic levers, as illustrated in the figure. First, a literature review on existing edge-cloud architecture for Predictive Maintenance, signal processing techniques, and dimensionality reduction models is provided in Section 3.1. Then, The first lever, acting on data collection and sharing, is reported in section 3.2, while the second, acting on data reducing, is in section 3.3. Subsections from 3.2.1 to 3.2.6 illustrate the data collection process of several case studies. In particular, the first case study considers a test rig built by the research team of the Department of Industrial Engineering of the University of Bologna. It is considered operating in a controlled environment since several scenarios are simulated according to a specific object, and datasets are labeled, balanced, and complete. The other three case studies consider sub-systems of automatic machinery, and the data have been collected by machinery producers during simulations or by their clients during production. Hence, they are considered operating in uncontrolled environments. Case studies on Data Processing are reported in sections 3.3.1, 3.3.2, and 3.3.3, in which applications of signal processing techniques, feature selection methods, and feature construction methods are described. Finally, in section 3.4, relevant conclusions are drawn, and the outcomes of the research activity are provided.

Chapter 4 addresses *RQ 3* by exploring three research topics underpinned by three strategic levers, as illustrated in the figure. First, a literature review on classification models for fault diagnosis and detection and novelty detection approaches is provided in section 4.1. Then, the first level, acting on offline and supervised ML models for diagnostics, is reported in subsection 4.2. Sections 4.3 and 4.4 report the strategic levers of online and streaming analysis and the methodology design, respectively. In particular, the offline analysis is divided according to two objectives, i.e., operating setting recognition and fault diagnosis, which are addressed in sections 4.2.1 and 4.2.2, respectively. Section 4.3.1 describes the deep learning approach for novelty detection, and results of the analysis are provided in section 4.3.2.

Finally, chapter 5 concludes the dissertation by illustrating the obtained results and the managerial insights and proposing potential future developments.

The readers interested in exploring some of the presented research topics are referred to the list of appended papers.

1.7 References

Alsyouf, Imad. 2007. “The Role of Maintenance in Improving Companies’ Productivity and Profitability.” *International Journal of Production Economics* 105 (1): 70–78. https://doi.org/10.1016/j.ijpe.2004.06.057.

Brundage, Michael P., Thurston Sexton, Melinda Hodkiewicz, K. C. Morris, Jorge Arinez, Farhad Ameri, Jun Ni, and Guoxian Xiao. 2019. “Where Do We Start? Guidance for Technology Implementation in Maintenance Management for Manufacturing.” *Journal of Manufacturing Science and Engineering, Transactions of the ASME* 141 (9): 1–16. https://doi.org/10.1115/1.4044105.

Calabrese, Francesca, Alberto Regattieri, Marco Bortolini, Mauro Gamberi, and Francesco Pilati. 2021. “Predictive Maintenance : A Novel Framework for a Data-Driven , Semi-Supervised , and Partially Online Prognostic Health Management Application in Industries.” *Applied Sciences (Switzerland)* 11: 1–28.

Dalzochio, Jovani, Rafael Kunst, Edison Pignaton, Alecio Binotto, Srijnan Sanyal, Jose Favilla, and Jorge Barbosa. 2020. “Machine Learning and Reasoning for Predictive Maintenance in Industry 4.0: Current Status and Challenges.” *Computers in Industry* 123: 103298. https://doi.org/10.1016/j.compind.2020.103298.

Fraser, Kym, Hans Henrik Hvolby, and Chihiro Watanabe. 2011. “A Review of the Three Most Popular Maintenance Systems: How Well Is the Energy Sector Represented?” *International Journal of Global Energy Issues* 35 (2–4): 287–309. https://doi.org/10.1504/IJGEI.2011.045024.

Holgado, Maria, Marco Macchi, and Stephen Evans. 2020. “Exploring the Impacts and Contributions of Maintenance Function for Sustainable Manufacturing.” *International Journal of Production Research* 58 (23): 7292–7310. https://doi.org/10.1080/00207543.2020.1808257.

Jasiulewicz-Kaczmarek, Małgorzata, Stanisław Legutko, and Piotr Kluk. 2020. “Maintenance 4.0 Technologies - New Opportunities for Sustainability Driven Maintenance.” *Management and Production Engineering Review* 11 (2): 74–87. https://doi.org/10.24425/mper.2020.133730.

Jonge, Bram de, and Philip A. Scarf. 2020. “A Review on Maintenance Optimization.” *European Journal of Operational Research* 285 (3): 805–24. https://doi.org/10.1016/j.ejor.2019.09.047.

Lee, Jay. 2020. *Industrial AI*. *Industrial AI*. https://doi.org/10.1007/978-981-15-2144-7.

Lei, Yaguo, Naipeng Li, Liang Guo, Ningbo Li, Tao Yan, and Jing Lin. 2018. “Machinery Health Prognostics: A Systematic Review from Data Acquisition to RUL Prediction.” *Mechanical Systems and Signal Processing* 104: 799–834. https://doi.org/10.1016/j.ymssp.2017.11.016.

Lei, Yaguo, Bin Yang, Xinwei Jiang, Feng Jia, Naipeng Li, and Asoke K. Nandi. 2020. “Applications of Machine Learning to Machine Fault Diagnosis: A Review and Roadmap.” *Mechanical Systems and Signal Processing* 138: 106587. https://doi.org/10.1016/j.ymssp.2019.106587.

Lemes, L. C., and L. Hvam. 2019. “Maintenance Costs in the Process Industry: A Literature Review.” *IEEE International Conference on Industrial Engineering and Engineering Management*, 1481–85. https://doi.org/10.1109/IEEM44572.2019.8978559.

Li, Rui, Wim J.C. Verhagen, and Richard Curran. 2020. “A Systematic Methodology for Prognostic and Health Management System Architecture Definition.” *Reliability Engineering and System Safety* 193 (August 2019): 106598. https://doi.org/10.1016/j.ress.2019.106598.

Maletič, Damjan, Matjaž Maletič, Basim Al-Najjar, and Boštjan Gomišček. 2014. “The Role of Maintenance in Improving Company’s Competitiveness and Profitability A Case Study in a Textile Company.” *Journal of Manufacturing Technology Management* 25 (4): 441–56. https://doi.org/10.1108/JMTM-04-2013-0033.

Poór, Peter, David Ženíšek, and Josef Basl. 2019. “Historical Overview of Maintenance Management Strategies: Development from Breakdown Maintenance to Predictive Maintenance in Accordance with Four Industrial Revolutions.” *Proceedings of the International Conference on Industrial Engineering and Operations Management*, no. July: 495–504.

Sajid, Sufiyan, Abid Haleem, Shashi Bahl, Mohd Javaid, Tarun Goyal, and Manoj Mittal. 2021. “Data Science Applications for Predictive Maintenance and Materials Science in Context to Industry 4.0.” *Materials Today: Proceedings* 45: 4898–4905. https://doi.org/10.1016/j.matpr.2021.01.357.

Thomas, Douglas, and Brian Weiss. 2021. “Maintenance Costs and Advanced Maintenance Techniques in Manufacturing Machinery: Survey and Analysis.” *International Journal of Prognostics and Health Management* 12 (1): 1–13. https://doi.org/10.36001/IJPHM.2021.V12I1.2883.

Zonta, Tiago, Cristiano André da Costa, Rodrigo da Rosa Righi, Miromar José de Lima, Eduardo Silveira da Trindade, and Guann Pyng Li. 2020. “Predictive Maintenance in the Industry 4.0: A Systematic Literature Review.” *Computers and Industrial Engineering* 150 (August): 106889. https://doi.org/10.1016/j.cie.2020.106889.

2. Diagnostics And Prognostics In Industry 4.0: Literature, Tools, And Industrial Requirements

In the last decades, scientific and industrial interest in asset health management has grown, together with Information and Communication Technologies (ICT) and digital innovation of factories, whose elements fall into the concept of Industry 4.0. In this context, machines are connected into an Industrial Internet of Things (IIoT) that enables the collection of as much data as possible from all complex physical assets on the shop floor, their transmission into the cloud, and their processing with intelligent algorithms to make factories more productive, less costly to operate, and more reliable (Balogh et al. 2018). From a strategic point of view, the knowledge of the exact assets' health condition in a given time instant has become an essential driver in the maintenance management, since it provides the opportunity of setting efficient, just-in-time, and just-right maintenance strategies, resulting in the maximization of the production profits and minimization of all costs and losses, including asset ones (Compare, Baraldi, and Zio 2020). From an operational point of view, this implies the collection of large volumes of data from multiple sources and requires proper Big Data processing technologies to build an integrated environment in which the production process, and the maintenance process, can be represented transparently and managed in a more efficient way (Yan et al. 2017). Hence, two areas of interest can be highlighted among Industry 4.0 technologies.

The first area corresponds to the scope of the analysis itself, i.e., maintenance of physical industrial assets, and concerns the analysis of the collected data to increase the knowledge about assets' health management. In this context, Karim et al. (Karim et al. 2016) developed a concept for Maintenance Analytics, which aims to improve information extraction and knowledge discovery from industrial Big Data to support the maintenance decision-making process. The proposed concept is based on four interconnected and sequential phases:

1. Maintenance Descriptive Analytics. It uses retrospective analysis of maintenance processes and the evaluation of proper KPI (e.g., the machine with less availability) to answer the question "what happened?"
2. Maintenance Diagnostic Analytics. It identifies the relationships between the event (failure) and its cause (e.g., the gearboxes failed because of an overload of the braking force) to answer the question "why it happened?"

3. Maintenance Predictive Analytics. It uses historical data and forecasting analysis to answer the question "what and when will happen?"
4. Maintenance Prescriptive Analytics. It aims to prescribe optimal actions in real-time and analyze potential decisions and their interactions. In other words, it answers the question "what should be done?"

The second area of interest is related to the infrastructure to collect, transmit, and store data that allows monitoring the equipment's technological processes and health condition. Considering that a large volume of data is generated on the shop floor, creating systems that enable data collection from different data sources automatically and transparently is necessary (Yan et al., 2017). For this purpose, the development of data collection systems demands an efficient selection of technologies to collect data based on IoT technologies (Cachada et al. 2019). In particular, the hardware architecture should allow (Vlasov et al. 2018):

1. Transfering signals from the field level devices (sensors) to engineering units and transfer of information to a data collection station (monitoring of engineering system);
2. Executing algorithms for calculating physical quantities to take into account the obtained data (processing of obtained data);
3. Controlling technical systems, taking into account the built-in algorithms (self-diagnostics of the central nodes);
4. Creating and transmitting warning and alarm signals (timely notification and assistance in decision-making).

Within the Industry 4.0 context, these areas have become strictly dependent. Traditionally, processing methods relied on local high-performance computers and simple parallel operations to improve computational power. However, these conventional in-house computers with constrained resources, i.e., storage, memory, and processing power, and also cloud data centers, are not suitable to address the Big Data challenges, mainly due to scalability and computational complexity, and the higher latency and network usage due to the vast geographical distance between IIoT devices and cloud data center (Teoh, Gill, and Parlikad 2021). Hence, there is a need to overcome the limitations of original relational databases and serial algorithms by developing new types of distributed processing systems (Madden 2012). That goal can be achieved by considered the current Big Data processing systems, which are are mainly focused on batch processing of historical monitoring data and real-time processing of online data (Y. Xu et al. 2017):

1. Offline batch processing aims to uncover valuable information from accumulated massive data for fault prediction, and it is more suitable for mass stored data, as it focuses more on the accuracy and comprehensiveness of analysis rather than real-time diagnosis.
2. Real-time stream processing, needed in variable and complex operating conditions to realize a real-time diagnosis and ensure timely maintenance.

Within the context generated by the interconnection between hardware and software elements that realize different tasks to achieve the digitalization of the factory and impact all industrial functions, defining the burden of the research, and the elements that contribute to the defined object, is necessary. Figure 6 depicts the main elements of research, listed according to 4 Ws questions:

1. Why, to express the ultimate goal of this research, that is the real-time health assessment of physical assets installed in one single or more production plants;
2. What, to express the main research activities to achieve the goal mentioned above. Diagnostics and prognostics, which aim to detect and predict faults, represent the object.
3. How, to express the tools to address diagnostics and prognostics. In this research, only data-driven approaches based on Machine Learning and Deep Learning are considered;
4. Where/When, to express the computing location, i.e., where ML algorithms are trained and applied to make the inference, and the time in which they are used (batch or streaming).

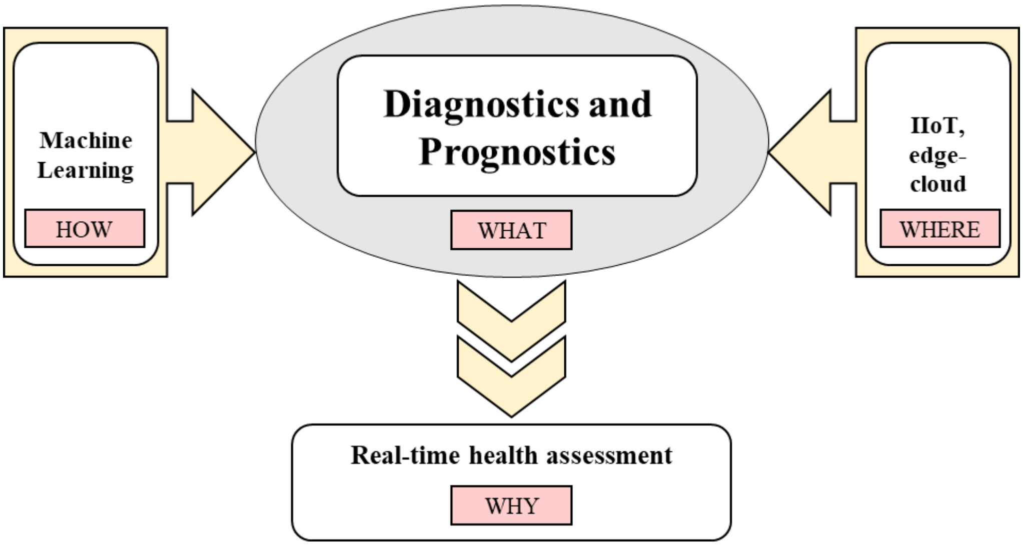


Figure 6 Research objects

The following sections address the defined topics. First, an analysis of the existing literature is presented to discover the research trends and the state-of-the-art. Then, an overview on prognostics and diagnostics, Machine Learning algorithms, and IIoT, edge, and cloud computing technologies is provided to describe general concepts of such tools.

2.1 Research trends

In order to support companies and managers during the initial step of a predictive maintenance system implementation, scholars are called to develop accessible and data-driven diagnostics and prognostics models to handle the large amount of data generated by connected machines. This chapter presents a bibliometric analysis to explore the scientific landscape on data-driven Predictive Maintenance and provide the reader with an initial macroscopic overview of the literature on the topic.

A database containing all the papers addressing the topic was analyzed through a pattern classification tool, identifying main research trends over the last decade. The dataset was retrieved on September 13th, 2021, from Scopus, one of the most widely used search engines among the scientific community. The keywords included in the search query, i.e., condition monitoring & maintenance, have been chosen because they represent the most general and fundamental concepts of Predictive Maintenance, allowing collecting all papers addressing the equipment health assessment based on its actual condition while leaving out other kinds of maintenance strategies. In addition, the research query has been built to include only engineering and computer science areas to limit the search to the engineering field and to the data-driven approach. Finally, keywords related to structural health monitoring, which belongs to the civil engineering subject area, have been excluded to consider only industrial applications. In total, 9473 were obtained, ranging from 1964 to 2022. The time span from the first publication to the last one has been divided into seven bins of different duration to find research trends. As shown in Figure 7, in the first 28 years, from 1964 to 1991, only 10% of papers were published. A relevant increase in the number of publications can be seen from 2007. Indeed,

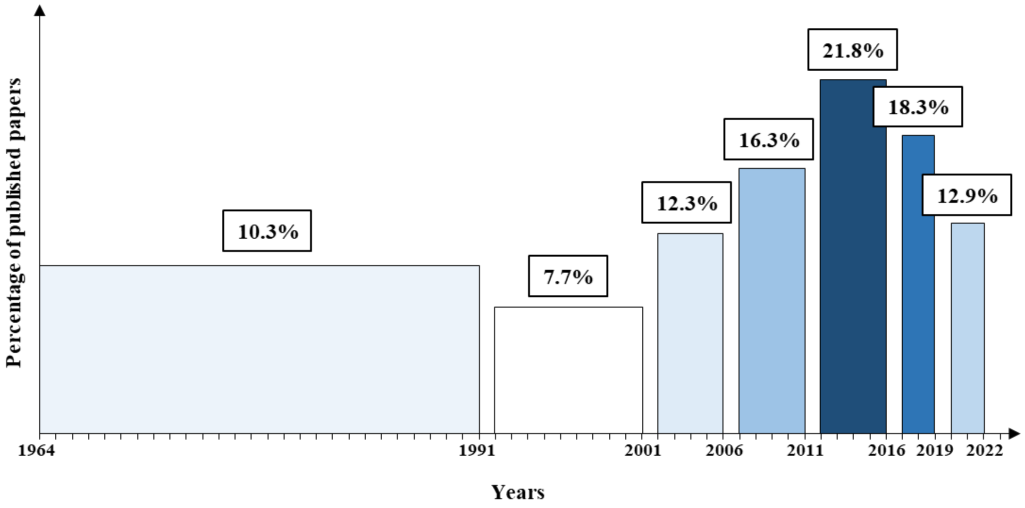


Figure 7 Distribution of published papers on Predictive Maintenance using condition monitoring techniques

from 2007 to today, 70% of papers were published. In particular, 21.8 % of papers were published in the period 2012-2016.

In order to track the evolution of maintenance approaches using condition monitoring data and the adopted tools from 1964 to today, a keyword analysis has been conducted. In particular, three categories of topics have been defined, i.e., condition-based maintenance, predictive maintenance, and technologies related to Industry 4.0. For each period, the most adopted keywords have been assigned to each category. For instance, diagnostics, fault diagnosis, Condition-Based Maintenance are assigned to the first category. Similarly, prognostics, Remaining Useful Life, fault prognosis have been assigned to the second category. Finally, keywords related to Machine Learning models, Big Data, sensor networks, Internet of Things have been assigned to the third category. Results are shown in Figure 8. It can be seen that Condition-Based Maintenance was subject to notable growth until 2016. In the meantime, Predictive Maintenance began to receive attention while ICT technologies were still in their infancy. In 2016, the trend completely changed: papers related to Condition-Based maintenance decreased, while papers addressing Industry 4.0 technologies increased notably. This trend seems to continue in the following years, too.

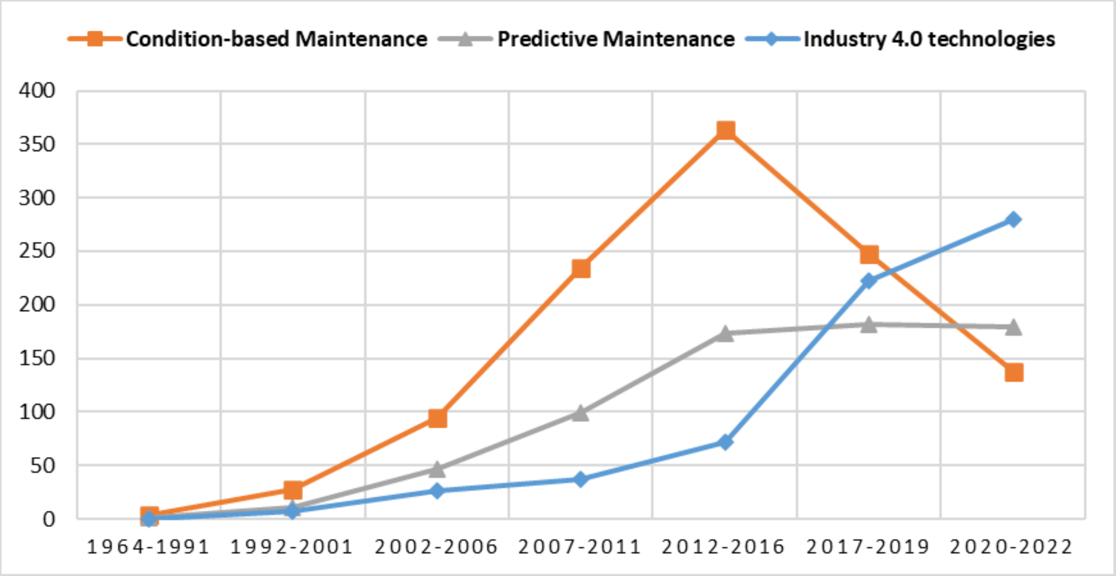


Figure 8 Publishing trends on maintenance strategies and tools

Although the subjective assignment of keywords to the three categories biases the results, an increased use of ICT technologies in maintenance is undeniable. The new paradigm of Industry 4.0 promoted the factory's digitalization, enabled a faster and more reliable collection of data, and made diagnostics and prognostics the core of maintenance optimization. On the other side, improvements in infrastructure technologies, like sensors and computational power, made Artificial Intelligence

(AI) a valuable opportunity for several industrial activities. Consequently, we are experiencing exponential growth of published papers that belong to several knowledge domains.

A co-occurrence analysis of all the keywords in the database was conducted to provide a demonstration of this statement. The citation information and keywords of the collected papers were exported and analyzed through the freely available software VOSviewer to construct and visualize bibliometric maps (van Eck and Waltman 2010). The software can draw bibliometric distance-based maps, paying much attention to their graphical appearance and facilitating the users' understanding. Results are visualized in a map, where the relatedness of pairs of items is calculated with respect to the number of documents in which they occur together. The fractional counting approach was implemented since the focus was to identify all documents in which each keyword occurs at least once, rather than the count of the total number of occurrences of a keyword. Given the high number of papers and keywords, a minimum number of occurrences equal to 10 were set, and the study was limited to the set of the most related keywords. All other settings have been given by default. The distance among items was calculated based on the association strength (van Eck and Waltman 2010), representing the ratio between the number of co-occurrences of term i and term j and the total number of occurrences of i multiplied for the total number of occurrences of j. The higher the association

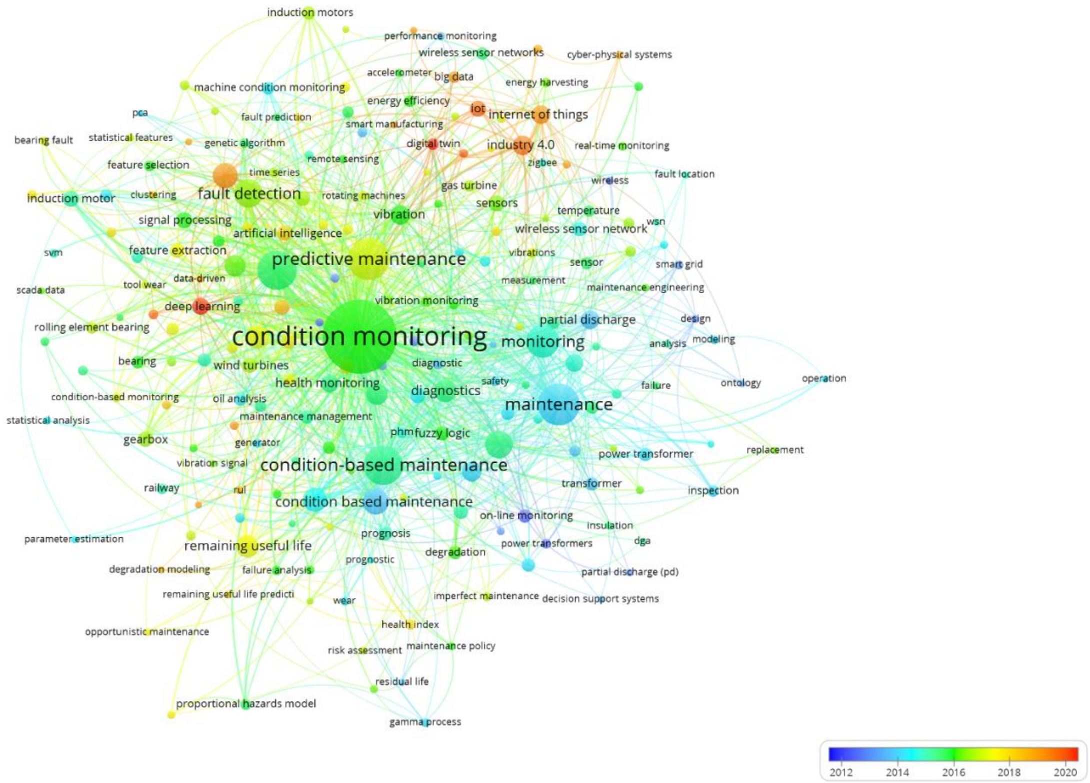


Figure 9 Map of Keywords from 2007 to 2022

strength between two nodes of the network, the shorter their distance is. The color scale refers to the average year of publications. Therefore, Figure 9 provides an overview of the most cited keywords over the years, from 2007 to 2022, enabling to identify the most debated research topics.

A preliminary analysis showed how the highest color variation is limited to the period between 2012 and 2020. Therefore, the color scale was re-set to this period. This phenomenon is partly influenced by the selected threshold on the occurrences number and the publication trend (see Figure 8). However, it is worth noting how a similar phenomenon is obtained by reducing the minimum number of occurrences to 15. Therefore, it is possible to assume an initial phase of warmup of the literature on condition monitoring-based maintenance (i.e., from 2007 to 2012) characterized by exploratory studies on different themes. Based on the map of keywords, the identified research trends are summarized in Table 1.

The obtained results provide the reader with a macroscopic overview of the extant literature on the topic. Moreover, Figure 9 allows visualizing past research trends and supports researchers in identifying potential literature gaps. In particular, the distance among nodes in the map could show new links among research topics, while a little bubble size could indicate a still little debated topic.

Table 1 Research Trends

Year	Research Trends
2012 – 2013	*Condition-based maintenance* for increasing the *availability* of machinery
2014 – 2015	*Prognostics and health management*
2016 – 2017	*Vibration*-based *fault diagnosis*
	Predictive Maintenance based on *Remaining Useful Life* prediction
	Fault Detection
2018 – 2019	*Machine Learning* for *Predictive Maintenance* in the *Industry 4.0* era
	Internet of Things for *Big Data* collection
2020	*Deep Learning* and *Digital Twin* for Predictive Maintenance

From the keywords co-occurrence analysis, it is possible to see that keywords like novelty detection, or streaming analysis, do not appear. The reason behind this result may depend on the threshold on the minimum number of keyword occurrences, which in turn may happen because it is a still unexplored topic. Indeed, although the studies on incremental learning and novelty detection are abundant in other fields, like data science, they still find little application in fault diagnosis and detection. However, considering the emergent Industry 4.0 technologies and the necessity of real-time monitoring of physical assets, they are necessary tasks to consider when dealing with Predictive Maintenance in industrial contexts.

The present dissertation addresses fault diagnosis and detection using data-driven approaches relying on data collection through sensor networks and the analysis using edge and cloud computing. Hence, diagnostics, prognostics, and the existing architectures for their implementation are first described. Then, Industrial Artificial Intelligence, and in particular Machine Learning models, for diagnostics are introduced. Finally, Industry 4.0 technologies, potentially involved in a predictive maintenance strategy, with a particular focus on edge and cloud computing, are reviewed. Finally, the last sub-chapter explores Predictive Maintenance from the point of view of industries. In particular, it illustrates the current industrial expectations and requirements on data-driven predictive maintenance in the context of Industry 4.0, with a focus on condition monitoring data collection, sharing, and analysis.

2.2 Overview on Diagnostics and Prognostics

According to (Lei 2016), diagnostics consists of three steps: (1) Fault detection, which is to indicate whether a fault has already occurred in the monitored machines; (2) Fault isolation, which is to find the faulty component and the position of the fault; (3) Fault identification, which attempts to determine the pattern and severity of the fault. Similarly, prognostic involves (1) State estimation, which relies on the output of fault identification in diagnostics and quantitatively identifies the severity of the fault and the degradation state of machinery; (2) State prediction, which predicts the degradation trend and speed of the state according to the information of the historical degradation curve; (3) RUL prediction, which is carried out by calculating the time length of the degradation curve from the current state to the final failure based on a predetermined Failure Threshold.

In other words, when monitoring the health condition of a component or system, the first step is to extract relevant parameters that allow detecting the fault. For instance, by monitoring the frequency spectrum of a rolling bearing instead of raw signals, a faulty bearing is easier to recognize. Hence, fault detection basically deals with the extraction of parameters revealing the component's health condition. Then, fault isolation deals with the recognition of where the fault occurred. In rolling bearings, different frequency value ranges correspond to different fault locations, e.g., inner race and outer race. Finally, the value of the frequency spectrum in a specific faulty mode allows identifying the severity of that fault, which corresponds to the fault identification phase. Once the relevant parameter (e.g., the frequency spectrum) and its specific value are known for a specific fault (e.g., a crack on the inner race), it is possible to conduct prognostics. In the state estimation phase, the main activity is to identify the severity of the fault based on the value assumed by the extracted Health

Indicator[2] at a specific moment. If what happened in the past is known, i.e., the degradation curve for that fault is available, then it is possible to predict the future values of the Health Indicator with a certain probability. Finally, the Remaining Useful Life can be computed as the difference between the time instant in which the Health Indicator is expected to achieve the known Failure Threshold (FT) and the current time.

From the above definitions, it emerges that while diagnostics is an a posteriori activity that detects and recognizes the fault after its occurrence, prognostics is an a priori activity that predicts the fault before its occurrence by computing at any point in time the time left before the failure, i.e., its Remaining Useful Life, or the system shutdown. Hence, diagnostics is the basis of Condition-Based Maintenance, while prognostics is the basis of Predictive Maintenance. The two activities are strictly correlated since the exact failure has to be known before predicting its occurrence. Hence, fault diagnosis is a prerequisite of prognostics. The opposite is not always true. Not all failures' cause is degradation. For example, a fault in an electric motor caused by a voltage drop cannot be predicted in any way.

Diagnostics and prognostics can be faced through physics-based, data-driven, or hybrid approaches (An, Kim, and Choi 2015). Physics-based modeling is preferred for simple components whose degradation behavior is readily available. In this case, mathematical models describing the degradation processes based on physics principles are built. However, it is difficult to describe complex systems physically and to know every possible failure mode for the component. These reasons, and the advances in the data analytics field, suggest using data-driven approaches, which can build degradation models only depending on data collected during the component functioning, i.e., CM data. Even if highly promising, data-driven approaches require data to be reliable and consistent, which may be a problem when considering peculiarities of CM data, like noise and measurement errors. Hybrid models, combining both model-based and data-driven prognostic, represent a valid alternative to solve issues related to each approach.

Concerning data-driven approaches, several frameworks and architectures have been proposed in the literature for diagnostics and prognostics, which consider the whole process from data acquisition to knowledge development.

First, the BS ISO 13374-1:2003 (ISO 2003) establishes general guidelines for software specifications related to data processing, communication, and presentation of machine condition monitoring and diagnostic information. According to that standard, machine condition assessment

[2] The Health Indicator is a particular feature, characterized by a monotonic trend, which can reveal the fault progression during the componnet's life

can be broken into six distinct, layered processing blocks (Figure 10). The first three blocks are technology-specific, requiring signal processing and data analysis functions targeted to a particular technology. The final three blocks usually combine monitoring technologies to assess the machine's current health, predict future failures, and provide recommended action steps to operations and maintenance personnel. The technology-specific blocks and the functions they should provide are the following:

1. Data Acquisition (DA) block: converts an output from the transducer to a digital parameter representing a physical quantity and related information (such as the time, calibration, data quality, data collector utilized, sensor configuration).
2. Data Manipulation (DM block): performs signal analysis, computes meaningful descriptors, and derives virtual sensor readings from the raw measurements.
3. State Detection (SD block): facilitates the creation and maintenance of normal baseline “profiles”, searches for abnormalities whenever new data are acquired, and determines in which abnormality zone, if any, the data belong (e.g. “alert” or “alarm”).

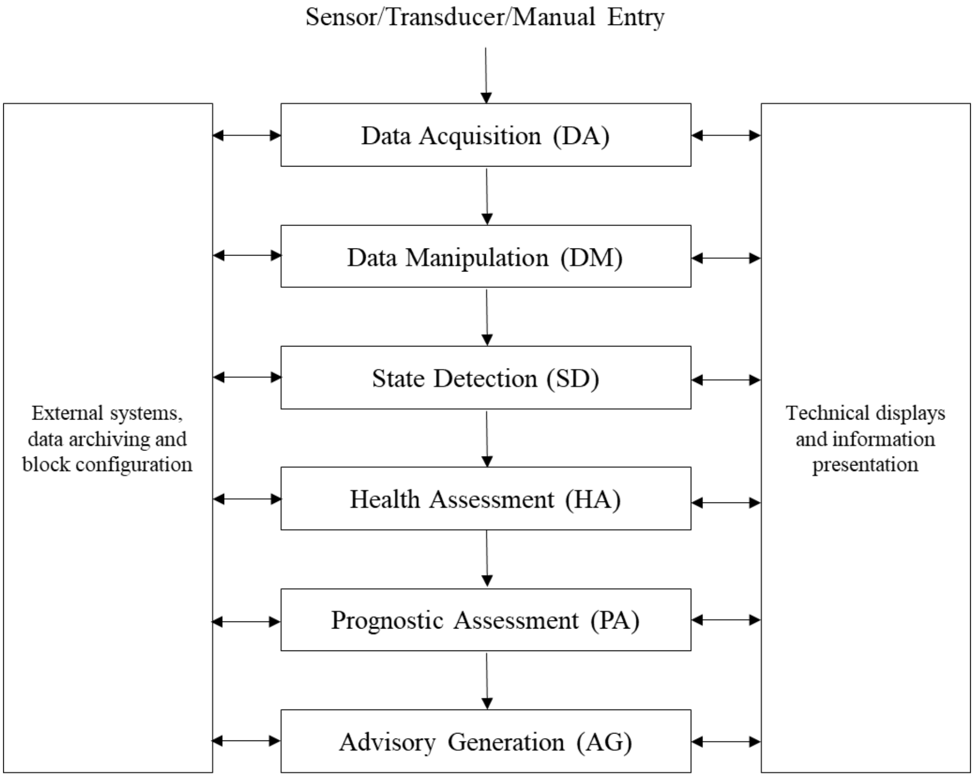

Sensor/Transducer/Manual Entry
External systems, data archiving and block configuration
Data Acquisition (DA)
Data Manipulation (DM)
State Detection (SD)
Health Assessment (HA)
Prognostic Assessment (PA)
Advisory Generation (AG)
Technical displays and information presentation

Figure 10 Data-processing and information-flow blocks (BS ISO 13374-1:2003)

The final three blocks and the functions they should support are the following:

1. Health Assessment (HA) block: diagnoses any faults and rates the current health of the equipment or process, considering all state information.

2. Prognostic Assessment (PA) block: determines future health states and failure modes based on the current health assessment and projected usage loads on the equipment and process and remaining useful life predictions.
3. Advisory Generation (AG) block: provides actionable information regarding maintenance or operational changes required to optimize the life of the process and/or equipment.

Jardine et al. (2006) consider the process for Condition-based Maintenance made of 4 steps, named data collection, feature extraction, diagnostics, and prognostics. According to the definition given in the paper, data collection is the process of acquiring data through condition monitoring techniques; feature extraction aims to extract and select relevant parameters that are able to distinguish the different health conditions of a component or system; diagnostics aims to find the relationships between the extracted features and the health condition; finally, prognostics deals with the RUL prediction.

More recently, the Prognostics and health management (PHM) approach has assumed a crucial role in the research on Predictive Maintenance. It enables real-time health assessment of a system under its actual operating conditions and the prediction of its future state based on up-to-date information by incorporating various disciplines, including sensing technologies, failure physics, Machine Learning, modern statistics, and reliability engineering (Kim Dawn An Joo-Ho Choi 2017). In (R. Li, Verhagen, and Curran 2020), functional architecture for PHM implementation is proposed. Similarly to the BS ISO 13374-1:2003, it includes several blocks with the same functions, which are named as Data Acquisition (DA), Data Processing (DP), Diagnostic Assessment (FDA), Prognostic Assessment (PA), Health Management (HM) (See Figure 11). In addition to the functions specified in the standard BS ISO 13374-1:2003, this architecture also includes the temporary storage memory as a function in the DA block and the data transmission into the DP block.

Lei et al. (2018) proposed a PHM approach from a different perspective. Similar to the PHM architecture proposed in (Jardine, Lin, and Banjevic 2006), it includes four steps named data collection, Health Indicator (HI) construction, Health Stage (HS) division, and the Remaining Useful Life (RUL) prediction. Contrary to previous architectures, the variable time is considered in this approach. Indeed, the HS division aims to divide the continuous degradation processes of machinery into different HSs according to the varying trends of the Health Indicator (HI) in each HS. The HI is a particular feature with peculiar characteristics that reveals the component behavior during the degradation. In other words, this architecture creates the basis for a streaming implementation of the PHM approach, in which the traditional blocks of feature extraction, diagnostics, and prognostics are performed in a cyclic way to assess the machinery health condition at any point in time.

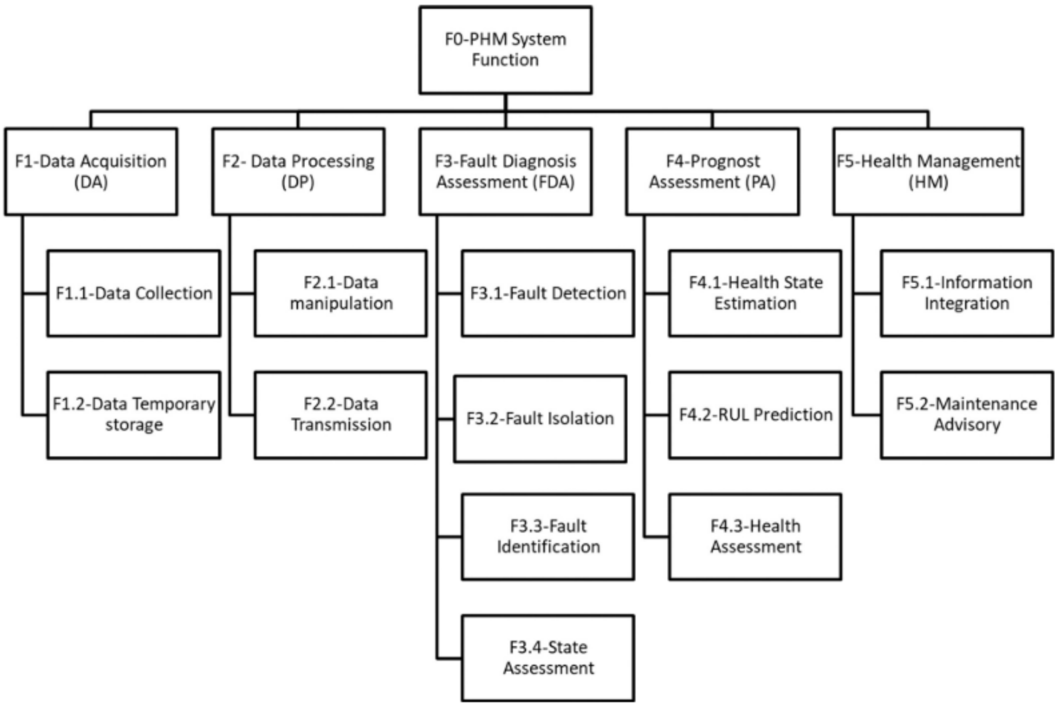


Figure 11 PHM function hierarchy diagram (R. Li, Verhagen, and Curran 2020)

Shin et al. (2018) provide a guideline for PHM application to six common core modules of intelligent manufacturing systems, classified depending on their main functional roles: power, hydraulic, control, drivers, transmission, and machining. In particular, the authors suggest four different PHM approaches depending on the amount of domain (physical) knowledge and sensing data. For instance, when high sensing data are available, and the domain knowledge is low, the data-driven approach is preferred, while the physics-based approach is suggested in the opposite case.

In the last two years, frameworks and architectures for the implementation of Predictive Maintenance using Industry 4.0 technologies have been proposed. In (Ran et al. 2019), a framework including five modules is introduced. The Data Acquisition module is responsible for the data collection from several sources via “wireless sensor networks” and the data storage in a data warehouse. The Data Pre-processing module is responsible for the data cleaning, integration, transformation, and feature extraction. The output of this module will be used as the input of the Data Analysis module, where advanced data analytics and machine/deep learning are used to perform the knowledge generation. The Decision Support module will visualize the Data Analysis module's result and provide an optimized maintenance schedule. Finally, the Maintenance Implementation module reacts to the physical world according to the maintenance decision and implements maintenance activities to achieve a specific purpose. The primary Industry 4.0 elements involved in this framework are comprise Cyber-Physical Systems (CPS), Internet of Things (IoT), Big Data, Data Mining, and Internet of Services (IoS), which will be defined in sub-section 2.4.

Similarly, Yu et al. (2021) developed an integrated framework for health state monitoring in Smart Factories employing IoT and Big Data techniques. The framework consists of four phases: (1) data ingestion, which is responsible for extracting different types of data from various data sources into a system; (2) data management, which is responsible for data storage; (3) data preparation, or data cleaning, which includes sensor selection, noise detection, and high-variance feature removal to produce high-quality data and avoid bias in predictive model training; (4) predictive modeling, in which the embedded prediction model acts as a real-time anomaly detector which takes the pre-processed data as inputs, yielding diagnostic results for maintenance decision support.

In conclusion, several frameworks for Condition-based Maintenance, Prognostics and Health Management, and Predictive Maintenance have been developed in the literature. Despite the different terminology adopted by the authors, four main activities are involved in these processes. First, data must be collected from machinery and processed to extract relevant information from weak signals; then, the outputs feed advanced data analytics techniques for fault classification and RUL prediction. Finally, the extracted information is used for maintenance activities optimization.

The following sub-sections provide the basic knowledge of data analytics techniques, particularly Machine Learning techniques, involved in diagnostics and prognostics.

2.3 Overview of Machine Learning for Predictive Maintenance[3]

Machine Learning is a computer science discipline that is part of the broader branch of Artificial Intelligence, whose goal is to design general-purpose methodologies to extract valuable patterns from data, ideally without much domain-specific expertise (Deisenroth, Faisal, and Ong 2020). Contrary to traditional systems that receive input values, process them, and produce output results, an intelligent system continuously adapts its model parameters and architectures to external stimuli (datasets or real-time inputs) and predicts the future using uncertain and fragmentary pieces of information. The ability to change according to external stimuli and remember most of the previous experience is what learning means. More precisely, the main goal of machine learning is to study, engineer, and improve mathematical models that can be trained (once or continuously) with context-related data (provided by a generic environment) to infer the future and to make decisions without complete knowledge of all influencing elements (external factors) (Bonaccorso 2018). Learning can

[3] The section is based on (Bonaccorso 2018) and (Deisenroth, Faisal, and Ong 2020).

be understood as a way to automatically find patterns and structures in data by optimizing the model's parameters. In other words, given a dataset and a suitable model, training the model means using the data available to optimize some model parameters with respect to a utility function that evaluates how well the model predicts the training data.

As illustrated in Figure 12, Machine Learning has four pillars, i.e., regression, dimensionality reduction, density estimation, and classification, which lie on several mathematical foundations. This section will focus on the basic concepts, i.e., input data format, learning paradigms, and goals, of Machine Learning without going deep into the mathematical formulations. In particular, the focus will be on classification and dimensionality reduction since they are the main involved tasks in the goals of the present dissertation. In addition, basic concepts of clustering will also be provided.

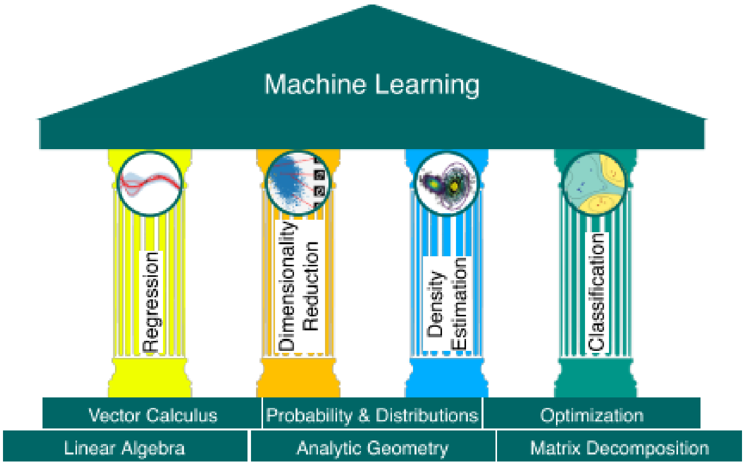
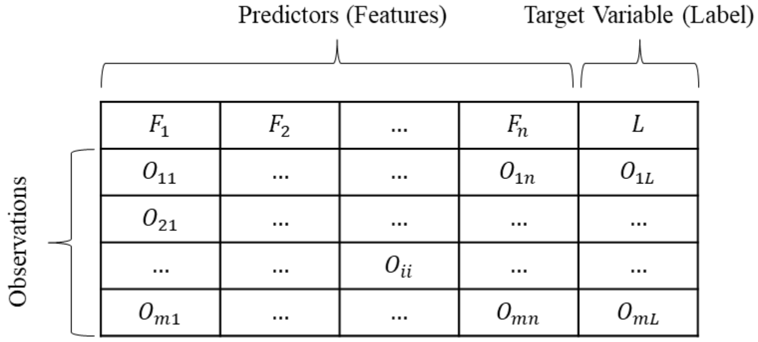


Figure 12 The foundations and four pillars of machine learning (Deisenroth, Faisal, and Ong 2020)

First of all, to build a Machine Learning model, input data need to be in a tabular format. Having data in this format does not mean that data in the form of images or text cannot be processed. However, it is always necessary to manipulate the data to find numerical representation and transform them into a tabular format. A typical input dataset for Machine Learning is represented in Figure 13, where each row corresponds to an example or observation, and each column represents a particular feature of interest of the example. In addition, in some cases, the column containing the output to predict, also named label or target variable, is included within the dataset. The consideration of the label in the learning process is what distinguishes supervised learning from unsupervised learning. While supervised learning aims to learn a mapping from the input to output, unsupervised learning aims to discover hidden patterns in data and is helpful to learn how a set of elements can be grouped (clustered) according to their similarity (or distance measure). A trade-off between supervised and unsupervised learning, named semi-supervised learning, involves using both labeled and unlabeled data. It can be adopted when it is necessary to categorize a large amount of data with a few labeled observations or when there is the need to impose some constraints to a clustering algorithm (for example, assigning some elements to a specific cluster or excluding others).



F_1	F_2	...	F_n	L
O_{11}	...	...	O_{1n}	O_{1L}
O_{21}	...	...	...	...
...	...	O_{ii}	...	...
O_{m1}	...	...	O_{mn}	O_{mL}

Figure 13 The structure of a training dataset for ML problems

During the supervised learning process, three phases can be distinguished (Deisenroth, Faisal, and Ong 2020):

1. Training or parameter estimation consists of finding a function and adjusting its parameters based on training data to find a good predictor. For non-probabilistic models[4], the principle of empirical risk minimization is commonly adopted, which directly provides an optimization problem for finding suitable parameters
2. Hyperparameter tuning or model selection allows the model to perform well on training data and unseen data. Hyper-parameters are different from the parameters learned during the training (Luo 2016). They are set before training and can either be set manually by the user or optimized through automatic selection methods (L. Yang and Shami 2020).
3. The prediction or inference phase consists of the inference by the trained predictor on previously unseen data.

Given a sufficiently rich class of functions for the predictor, the training data can be memorized to obtain zero empirical risk, i.e., the error. However, such predictors will not be expected to generalize well to unseen data. Since the training aims to build a predictor that also performs well on unseen data, i.e., has good generalization ability, a portion of the whole dataset, that is called test set, is held out during the training phase. Therefore, the training set is used to fit the model, i.e., to learn the relationships between input and output, and the test set (not seen by the machine learning algorithm during training) is used to evaluate generalization performance. A predictor that fits too closely the training data, i.e., having a small error on the training set, and that does not generalize well to new data, i.e., having a high error on the test set, presents an issue commonly referred to as

[4] Instead of considering a predictor as a single function, predictors can be considered probabilistic models, i.e., models describing the distribution of possible functions.

overfitting. This may result from too complex models, i.e., models having too many parameters to learn. On the other hand, the opposite problem, named underfitting and leading to opposite results, may occur if model complexity is low. Hence, a predictor can be considered good enough if it finds the optimal trade-off between the complexity and the generalization ability, also named bias-variance trade-off (Dougherty 2013). When the input dataset is not large enough, splitting it into training and test sets may reduce the number of samples used to fit the model. As the assumption is that the dataset represents an underlying data generating process, it is necessary that both the training and test sets obey this rule. Small datasets can have only a few samples representing a specific area of the data generating process, and it is essential to include them in the training set to avoid the lack of generalization ability. K-fold cross-validation can help solve this problem with a different strategy. According to this technique, the whole dataset is divided into k non-overlapping sets, and the training-test cycle is repeated k times. One of the k sets is held for testing at each iteration, while training is performed on the remaining $k-1$ sets (Rauber et al., 2021).

A classification problem assigns the input to one or more classes based on classification rules learned from training data. In some cases, the probability that a given input belongs to a class can also be calculated. The application of classification in Predictive Maintenance is mainly related to fault diagnosis. The training data set for fault diagnosis is represented by signals (condition monitoring data) collected during the machine nominal functioning and one or more faulty conditions. Hence, the label is represented by the different modes in which the system worked during data collection. Similar to classification, regression is a supervised learning problem that aims to find a mapping from the input to the output. Contrary to classification, the output is numerical instead of categorical. Regression problems in Predictive Maintenance are related to fault prognosis. Hence, the target variable is represented by the Remaining Useful Life.

Clustering is an unsupervised learning problem that aims to find groups of similar inputs without knowing the target value. Clustering may be used instead of classification when labels are not available. Therefore, it is mainly used for fault detection. The difference between classification, clustering, and regression is shown in Figure 14.

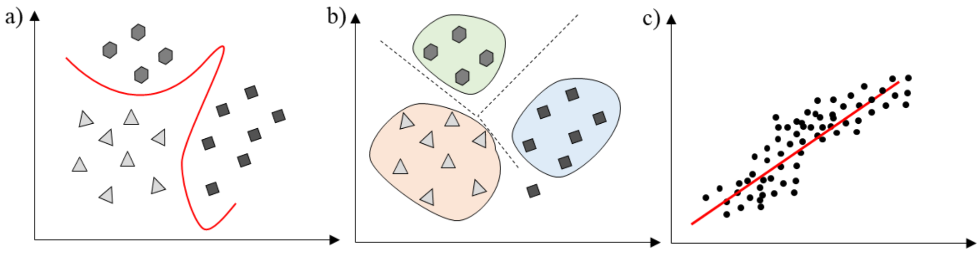


Figure 14 a) Classification; b) Clustering; c) Regression

Dimensionality Reduction aims to reduce the number of variables in a dataset to increase the accuracy of classification, regression, and clustering models and avoid the “curse of dimensionality” problem. High-dimensional data are hard to analyze, interpretation is problematic, visualization is nearly impossible, and (from a practical point of view) storage of the data vectors can be expensive. In addition, many dimensions may be redundant and can be explained by a combination of other dimensions, or are correlated, so that the data possesses an intrinsic lower-dimensional structure. Dimensionality reduction exploits structure and correlation and allows us to work with a more compact data representation, ideally without losing information. Dimensionality Reduction techniques may adopt supervised and unsupervised learning paradigms and, in the field of fault diagnosis and fault prognosis, are often applied after feature extraction to construct or select the most relevant variable for classification, clustering, or regression (F. Calabrese et al., 2022).

Another application of machine learning is anomaly detection, which finds instances that do not obey the general rule and are exceptions. The idea is that typical instances share characteristics that can be stated, and instances that do not have those characteristics are atypical. In such a case, the goal is to find a simple rule that covers as large a proportion of our typical instances as possible (Ahmad et al., 2017). Any instance that falls outside is an exception or a novel, previously unseen but valid case, and hence the other name novelty detection. Anomaly detection can be seen as a classification problem with two classes, i.e., nominal and anomalous, or may adopt a semi-supervised approach, in which only the nominal class is used in model training.

Details on specific algorithms for each of the Machine Learning tasks will be provided in the following chapters.

2.4 Overview on Industry 4.0 technologies for Predictive Maintenance

The new vision for the manufacturing systems provided by Industry 4.0 produces a manufacturing environment composed of product, intelligence, communication, and networking (Oztemel and Gursev 2020). In Smart Factoris, all machines on the shop floor are connected among each other and with a central data cloud, communicate and share data and information. They are self-aware and self-maintained, meaning that machines can self-assess their past or current health condition and further use similar information from other peers for smart maintenance decisions to avoid potential issues (J. Lee, Kao, and Yang 2014). Data-driven models are built to analyze data/information collected from machines whose health condition is fed back to the machine controller in real-time for adaptive control and in-time maintenance.

The core technologies of Industry 4.0 include the Internet of Things (IIoT), Big Data analysis, horizontal and vertical integration of systems, simulations, clouds, augmented reality, autonomous robots, 3D printing, and Cyber security (Bona et al. 2021). For the aim of this dissertation, four main elements will be considered, CPS, IIoT, cloud computing, edge computing.

Industrial Cyber-Physical Systems (CPS) integrate the physical world, made of sensors, actuators, and equipment deployed in the industrial plant, with the cyber world, composed by the computation, networking, and control systems, that collect and analyze data from both the physical and digital worlds to enable the operation, interconnection and provide industrial systems with intelligence. In Industrial CPS, machines devices exchange information with each other through the Machine-to-Machine (M2M) communication model, realized with either "Wired" or "Wireless" networks (G. Y. Lee et al. 2018). In (Bagheri et al. 2015), a five-level architecture for the industrial CPS implementation is proposed. First, in the smart connection layer, accurate and reliable data are acquired from machines from sensors. A second layer is dedicated to the data-to-information conversion, in which algorithms transform data into meaningful information to bring self-awareness to machines. The third layer is the cyber layer, which acts as a central information hub, which receives information from all connected devices and integrates them with historical information, providing machines with self-comparison ability. Then, in the cognition level, the priority of tasks to optimize the maintenance process is decided. Finally, a supervisory control from the cyber to physical devices makes the machine self-configurable and self-adaptive. As shown in Figure 15, in an IoT-based CPS, smart connection can be realized through the IoT devices, the data-to-information conversion at the edge for data pre-processing and real-time analytics, the cyber layer could be represented by only the

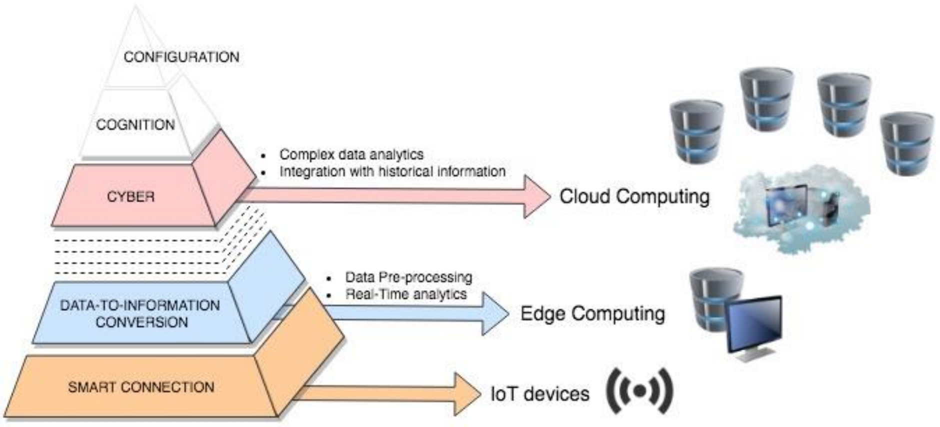


Figure 15 A CPS architecture based on IoT technology

cloud computing, in which complex data analytics and the integration with historical information can occur.

According to (W. Z. Khan et al. 2020), IIoT is the network of intelligent and highly connected industrial components deployed to achieve high production rate with reduced operational costs through real-time monitoring, efficient management, and control of industrial processes and assets operational time. In other words, IIoT makes possible the interconnection between intelligent industrial devices and control and management platforms. In Industrial IoT systems, the data source is represented by the sensors deployed in a wide area that provides different data types. The requirements generated by end devices are carried out into the cloud through the core network. Thanks to its significant storage and computational capabilities, the cloud is responsible for data processing. Therefore, the Industrial IoT offers ubiquitous access to entities on the Internet by using a variety of sensing, location tracking, and monitoring devices and enables “objects” to interact with each other and cooperate with their “smart” components (Ran et al. 2019). In other words, the IoT corresponds to the integration of communication layers of Industrial CPS (Fei et al. 2019) and provides the cyber space components, i.e., control, networking, and computing systems. In particular, the control system provides self-awareness and self-diagnosis capabilities to machines, enabling the detection and classification of failures for efficient management, effective utilization, and timely maintenance. The networking system enables communication among isolated industrial devices. The computing system provides the computational platform for time collection, storage, processing and data analysis (H. Xu et al. 2018).

Traditionally, the Industrial IoT requirements, i.e., transmission, storage, and computing, have been satisfied by cloud-based structures made of centralized servers that collect data from the connected devices and send back the results after data processing. Although its high storage and processing capacity, high reliability, scalability, and interoperability, cloud-based computing has two drawbacks from the data transmission point of view. First, the significant volume of data to transfer to the cloud may overload the network. Second, delays can occur in data transmission due to the distant location of the end devices from the cloud, which is especially critical for time-sensitive requirements and real-time applications. Hence, edge computing attracts significant attention since it moves data computing and storage from the cloud to the network edge located near the users. In this way, the peak traffic flows, the centralized network's bandwidth requirements, and the transmission latency during data computing and storage are notably reduced. Specifically, (1) due to the short transmission time of edge computing, the latency of the network is reduced, enabling real-time collection and analysis of the information; (2) due to its location, the edge-based storage provides short upload times of massive data; (3) Even though limited, edge devices have enough capacity to satisfy real-time IoT requirements; (4) Finally, edge nodes mitigate the power consumption of the IoT devices through the computation tasks offloading. However, because of the limited storage capacity of edge devices, several edge nodes will be used and coordinated for storing data, increasing the complexity of data management. Moreover, they are not able to perform complex data analytics. Hence, an IoT-based CPS can be realized by integrating cloud, and edge computing, which locally processes high-priority and delay-sensitive tasks while processing low-priority and delay-tolerant tasks in the cloud.

2.5 Industrial issues and expectations

Predictive Maintenance represents a great opportunity for manufacturing companies to improve productivity and profitability. In addition, the possibility to continuously monitor physical assets, enabled by the Industrial IoT and edge and cloud computing, opens a business opportunity for Original Equipment Manufacturing (OEM), which can offer to their client after-sales services through remote diagnostics and prognostics. (Brax and Jonsson 2009). This after-sale service has benefits for both OEMs and clients. For the former, it can represent a differentiation key from competitors. Indeed, machine producers usually adopt traditional preventive maintenance models to schedule maintenance interventions for their clients. This strategy often implies inspections that result in either the intervention or its rescheduling, depending on the actual detected condition. In both cases, maintenance inspections represent a waste of resources for machine producers and production losses for clients. Also, spare parts management is compromised. Second, thanks to the advances in technologies like Augmented Reality, corrective interventions are often conducted through remote maintenance, including the time needed for recognizing the problem before guiding the operator to the machine maintenance. The interest of OEMs in Predictive Maintenance as an after-sale service has also another objective. Machine producers are interested in the fault behaviors of machines operating in different environments. Indeed, they can use collected information to improve the machine design phase. Thus, a properly designed Predictive Maintenance service could (1) avoid inspections and support the optimal spare parts management, thanks to the continuous monitoring and prognostics; (2) speed up the process of remote maintenance, thanks to the streaming anomaly/fault detection; and (3) allow a structured data collection for the continuous improvement of existing diagnostics and prognostics models. On the other hand, clients could (1) always know the health condition of their machines; (2) recognize anomaly behaviors as they occur and, if possible, intervene without the involvement of the machine producer so to reduce the time to repair; (3) benefit from a most robust analysis conducted by the machine producer.

Despite the advances in ICT technologies and predictive analytics, companies at the beginning of the development of Predictive Maintenance, when actual data of normal and abnormal equipment behaviors are lacking or scarce, and companies producing new systems, when there is no experience on their operation, are intended to distrust the investment in Predictive Maintenance solutions (Compare, Baraldi, and Zio 2020). In the case of OEMs, they constantly design new types of machinery, for which historical data for models training is not available; moreover, clients are not always willing to share their data, as they feel their market share threatened by those producers that could decide to enter the same market with their own business. In particular, they tend to hide confidential data, such as the number of pieces produced every day, machinery settings, and the production process or materials. However, while the vendor knows its machine from the design phase, he does not know its behavior under the user's operating conditions (Sala et al. 2019), which affect the degradation of monitored component and often represent the labels (target value) needed for training the diagnostic and prognostic models. As a result, the data collected from clients are in most cases unlabelled, meaning that they do not record information about environmental conditions, operating conditions, fault modes (if any), maintenance interventions, or regulations carried on the machinery. Second, the amount of data generated by machinery is difficult to manage. The weight of high-frequency data is in the order of gigabytes per hour. It is impossible to acquire and store raw signals from all machinery in a continuous way. Therefore, data are usually collected intermittently, often during short tests conducted by machine producers in very controlled environments. Thus, data in fault conditions is trickier than data in healthy conditions to get, resulting in very unbalanced datasets; also, the dataset may not include data in different operating conditions, resulting in the impossibility to train accurate diagnostic and prognostic models. According to a study conducted by Madhikermi et al. (2018) in collaboration with two globally operating Original Equipment Manufacturing (OEM) companies, the following pitfalls that hinder companies from optimizing the maintenance service have been identified: wrong or incomplete data reported, lack of data validation technique, sensor data not logged for Condition-Based Maintenance, difficulty to integrate machine data and service reports.

Given these issues, the main objective of OEMs becomes to find a Predictive Maintenance architecture that: (1) can start from scratch, with very little prior knowledge about machinery behaviors; (2) supports the exchange of the data between producers and client, paying attention to privacy issues; (3) helps to collect the data in a structured way, with labels associated with each observation, i.e., low-frequency data associated with high-frequency data; (4) provides an early indication of anomalous and novel behaviors; (5) can learn from experience in different contexts; (6) performs different functions depending on the specific component and the whole machinery.

According to the findings of several interviews conducted on several machinery producers operating in the North of Italy, stakeholders requirements and benefits for a Predictive Maintenance system that allows remote monitoring, diagnosis, and prognosis, are the following (F. Calabrese et al. 2021):

1. A PdM system should allow the simultaneous monitoring of machines spread worldwide. Producers want to gather relevant data from installed equipment to expand their knowledge about the machinery's behavior during the actual functioning. Also, the remote monitoring of machinery health conditions would avoid inspections; hence, maintenance interventions and the spare parts management are improved.
2. A PdM system should be able to detect novel behaviors and learn from new incoming data. Machine producers have no sufficient data for training ML models because of the scarce possibility to conduct tests in their plants. The main issue is the lack of knowledge of all possible working conditions. Besides, many components fail after years of functioning, making it hard for the real-time application of pre-build prognostic models.
3. A PdM system should allow the collection of the data in a more structured way. Even if machine producers can get historical data from their clients, this data is not easy to process since the label (e.g., health status, implemented setting, and so on) is hardly available, and signals often present uncomprehensive trends. Low-frequency data, e.g., the setting of the machinery, and event data, e.g., the tuning made by operators or the anomalies that occurred during the machine functioning, should be collected together with high-frequency data. Hence, data would be automatically labeled, and every event affecting the trend of collected data would be recorded, speeding up the data pre-processing step.
4. A PdM system should accomplish various functions based on a specific component, which means that functions should be performed separately depending on the analysis goal. A complex machine consists of many elements, e.g., suckers, or sub-systems, e.g., extruder, for which various analyses may be helpful. For example, an extruder's screw degrades over time, and the goal is to predict its RUL. Instead, the detachment of a sucker happens suddenly. Thus, the goal is to detect the anomaly. Hence, the real-time analysis should consider these two components separately to understand the problem's cause and, if possible, let the machine operator intervene as soon as possible. However, as long as these two components work simultaneously under the same operating conditions, machine producers are also interested in how the anomalies or failures affect each other, considering a given working condition.
5. A PdM system should allow the CM data to integrate historical data usually recorded in an external database (e.g., Computerized Maintenance Management System – CMMS),

collecting work orders, spare parts management, and other information to develop a whole maintenance program. For machine producers that want to provide their clients with full-service maintenance, this is extremely important to get an internal optimization.

The above-described requirements represent the basis for the formulation of the research questions pointed out in section 1.1. Links between research questions, strategic lever, and requirements are shown in Table 2. In particular, the strategic lever acting on data collection and sharing derives from requirements 1, 3, and 5 since they imply building proper infrastructure for real-time data collection, sharing, and organization. The strategic lever acting on data reduction derives from requirement 4 since data reduction techniques allow performing both component-level and system-level analysis, depending on what data refer to. Similarly, the research lever acting on offline analysis addresses requirement 4 since it allows performing diagnostics and prognostics for each component and, eventually, at a system level. Research lever acting on streaming analysis derives from requirements 2, 3, and 4 since streaming algorithms allow discovering novel situations in real-time, automatically labeling current data, and performing component and system-level analysis. Finally, the last research lever acts on the design of a methodology that addresses all requirements.

Table 2 Research questions and industrial requirements

Research Question	Strategic Levers	Requirements
RQ 2	Acting on data collection and sharing	1, 3, 5
RQ 2	Acting on data reduction	4
RQ 3	Acting on offline analysis	4
RQ 3	Acting on streaming analysis	2, 3, 4
RQ 3	Acting on the logical architecture	1, 2, 3, 4, 5

2.6 References

Ahmad, Subutai, Alexander Lavin, Scott Purdy, and Zuha Agha. 2017. “Unsupervised Real-Time Anomaly Detection for Streaming Data.” *Neurocomputing* 262: 134–47. https://doi.org/10.1016/j.neucom.2017.04.070.

An, Dawn, Nam H. Kim, and Joo Ho Choi. 2015. “Practical Options for Selecting Data-Driven or Physics-Based Prognostics Algorithms with Reviews.” *Reliability Engineering and System Safety*. Elsevier Ltd. https://doi.org/10.1016/j.ress.2014.09.014.

Bagheri, Behrad, Shanhu Yang, Hung An Kao, and Jay Lee. 2015. “Cyber-Physical Systems Architecture for Self-Aware Machines in Industry 4.0 Environment.” *IFAC-PapersOnLine* 48 (3): 1622–27. https://doi.org/10.1016/j.ifacol.2015.06.318.

Balogh, Z., E. Gatial, J. Barbosa, P. Leitão, and T. Matejka. 2018. “Reference Architecture for a Collaborative Predictive Platform for Smart Maintenance in Manufacturing.” *INES 2018 - IEEE 22nd International Conference on Intelligent Engineering Systems, Proceedings*, 000299–000304. https://doi.org/10.1109/INES.2018.8523969.

Bona, Gianpaolo Di, Vittorio Cesarotti, Gabriella Arcese, and Tommaso Gallo. 2021. “Implementation of Industry 4.0 Technology: New Opportunities and Challenges for Maintenance Strategy.” *Procedia Computer Science* 180 (2019): 424–29. https://doi.org/10.1016/j.procs.2021.01.258.

Bonaccorso, Giuseppe. 2018. *Machine Learning Algorithms*. *O’Reilly*. Second. Birmingham: Packt Publishing Ltd. https://www.oreilly.com/library/view/machine-learning-algorithms/9781789347999/.

Brax, Saara A., and Katrin Jonsson. 2009. “Developing Integrated Solution Offerings for Remote Diagnostics: A Comparative Case Study of Two Manufacturers.” *International Journal of Operations and Production Management* 29 (5): 539–60. https://doi.org/10.1108/01443570910953621.

Cachada, Ana, Jose Barbosa, Paulo Leitao, Americo Alves, Leandro Alves, Joao Teixeira, and Carlos Teixeira. 2019. “Using Internet of Things Technologies for an Efficient Data Collection in Maintenance 4.0.” *Proceedings - 2019 IEEE International Conference on Industrial Cyber Physical Systems, ICPS 2019*, 113–18. https://doi.org/10.1109/ICPHYS.2019.8780217.

Calabrese, Francesca, Alberto Regattieri, Marco Bortolini, and Francesco Gabriele Galizia. 2022. “Fault Diagnosis in Industries: How to Improve the Health Assessment of Rotating Machinery.” In *Sustainable Design and Manufacturing. KES-SDM 2021. Smart Innovation, Systems and Technologies*, edited by Scholz S.G., Howlett R.J., and Setchi R., 257–66. Singapore. https://doi.org/10.1007/978-981-16-6128-0_25.

Calabrese, Francesca, Alberto Regattieri, Marco Bortolini, Mauro Gamberi, and Francesco Pilati. 2021. “Predictive Maintenance : A Novel Framework for a Data-Driven , Semi-Supervised , and Partially Online Prognostic Health Management Application in Industries.” *Applied Sciences (Switzerland)* 11: 1–28.

Compare, Michele, Piero Baraldi, and Enrico Zio. 2020. “Challenges to IoT-Enabled Predictive Maintenance for Industry 4.0.” *IEEE Internet of Things Journal* 7 (5): 4585–97. https://doi.org/10.1109/JIOT.2019.2957029.

Deisenroth, Marc Peter, A. Aldo Faisal, and Cheng Soon Ong. 2020. *Mathematics for Machine Learning*. *Quantitative Literacy: Why Numeracy Matters for Schools*. Cambridge University Press (2020). https://doi.org/10.1007/b97511.

Dougherty, Geoff. 2013. “Estimating and Comparing Classifiers.” In *Pattern Recognition and Classification: An Introduction*, 157–76. New York, NY: Springer. https://doi.org/10.1007/978-1-4614-5323-9.

Eck, Nees Jan van, and Ludo Waltman. 2010. “Software Survey: VOSviewer, a Computer Program for Bibliometric Mapping.” *Scientometrics* 84 (2): 523–38. https://doi.org/10.1007/s11192-009-0146-3.

Fei, Xiang, Nazaraf Shah, Nandor Verba, and Kuo-ming Chao. 2019. “CPS Data Streams Analytics Based on Machine Learning for Cloud and Fog Computing : A Survey.” *Future Generation Computer Systems* 90 (July): 435–50. https://doi.org/10.1016/j.future.2018.06.042.

ISO. 2003. “ISO 13374-1:2003(E) Condition Monitoring and Diagnostics of Machines — Data Processing, Communication and Presentation” 3: 24.

Jardine, Andrew K.S., Daming Lin, and Dragan Banjevic. 2006. “A Review on Machinery Diagnostics and Prognostics Implementing Condition-Based Maintenance.” *Mechanical Systems and Signal Processing*. https://doi.org/10.1016/j.ymssp.2005.09.012.

Karim, Ramin, Jesper Westerberg, Diego Galar, and Uday Kumar. 2016. “Maintenance Analytics – The New Know in Maintenance.” *IFAC-PapersOnLine* 49 (28): 214–19. https://doi.org/10.1016/j.ifacol.2016.11.037.

Khan, W. Z., M. H. Rehman, H. M. Zangoti, M. K. Afzal, N. Armi, and K. Salah. 2020. “Industrial Internet of Things: Recent Advances, Enabling Technologies and Open Challenges.” *Computers and Electrical Engineering* 81: 106522. https://doi.org/10.1016/j.compeleceng.2019.106522.

Kim Dawn An Joo-Ho Choi, Nam-Ho. 2017. *Prognostics and Health Management of Engineering Systems*. Springer International Publishing Switzerland.

Lee, Gil Yong, Mincheol Kim, Ying Jun Quan, Min Sik Kim, Thomas Joon Young Kim, Hae Sung Yoon, Sangkee Min, et al. 2018.

"Machine Health Management in Smart Factory: A Review." *Journal of Mechanical Science and Technology* 32 (3): 987–1009. https://doi.org/10.1007/s12206-018-0201-1.

Lee, Jay, Hung An Kao, and Shanhu Yang. 2014. "Service Innovation and Smart Analytics for Industry 4.0 and Big Data Environment." In *Procedia CIRP*, 16:3–8. Elsevier. https://doi.org/10.1016/j.procir.2014.02.001.

Lei, Yaguo. 2016. "Introduction and Background." In *Intelligent Fault Diagnosis and Remaining Useful Life Prediction of Rotating Machinery*, 1–16. Elsevier Inc. https://doi.org/10.1016/B978-0-12-811534-3/00001-9.

Lei, Yaguo, Naipeng Li, Liang Guo, Ningbo Li, Tao Yan, and Jing Lin. 2018. "Machinery Health Prognostics: A Systematic Review from Data Acquisition to RUL Prediction." *Mechanical Systems and Signal Processing* 104: 799–834. https://doi.org/10.1016/j.ymssp.2017.11.016.

Li, Rui, Wim J.C. Verhagen, and Richard Curran. 2020. "A Systematic Methodology for Prognostic and Health Management System Architecture Definition." *Reliability Engineering and System Safety* 193 (August 2019): 106598. https://doi.org/10.1016/j.ress.2019.106598.

Luo, Gang. 2016. "A Review of Automatic Selection Methods for Machine Learning Algorithms and Hyper-Parameter Values." *Network Modeling Analysis in Health Informatics and Bioinformatics* 5 (1): 1–16. https://doi.org/10.1007/s13721-016-0125-6.

Madden, Sam. 2012. "From Databases to Big Data." *IEEE Internet Computing* 16 (3): 4–6. https://doi.org/10.1109/MIC.2012.50.

Madhikermi, Manik, Andrea Buda, Bhargav Dave, and Kary Främling. 2018. "Key Data Quality Pitfalls for Condition Based Maintenance." *2017 2nd International Conference on System Reliability and Safety, ICSRS 2017* 2018-Janua: 474–80. https://doi.org/10.1109/ICSRS.2017.8272868.

Oztemel, Ercan, and Samet Gursev. 2020. "Literature Review of Industry 4.0 and Related Technologies." *Journal of Intelligent Manufacturing* 31 (1): 127–82. https://doi.org/10.1007/s10845-018-1433-8.

Ran, Yongyi, Xin Zhou, Pengfeng Lin, Yonggang Wen, and Ruilong Deng. 2019. "A Survey of Predictive Maintenance: Systems, Purposes and Approaches." *IEEE Communications Surveys & Tutorials*, 1–36.

Rauber, Thomas Walter, Antonio Luiz da Silva Loca, Francisco de Assis Boldt, Alexandre Loureiros Rodrigues, and Flávio Miguel Varejão. 2021. "An Experimental Methodology to Evaluate Machine Learning Methods for Fault Diagnosis Based on Vibration Signals." *Expert Systems with Applications* 167 (September 2020). https://doi.org/10.1016/j.eswa.2020.114022.

Sala, R., F. Pirola, E. Dovere, and S. Cavalieri. 2019. "A Dual Perspective Workflow to Improve Data Collection for Maintenance Delivery: An Industrial Case Study." In *IFIP WG 5.7 International Conference, APMS 2019*, 566:485–492. Springer Nature Switzerland. https://doi.org/10.1007/978-3-030-30000-5_60.

Shin, Insun, Junmin Lee, Jun Young Lee, Kyusung Jung, Daeil Kwon, Byeng D. Youn, Hyun Soo Jang, and Joo-Ho Choi. 2018. "A Framework for Prognostics and Health Management Applications toward Smart Manufacturing Systems." *International Journal of Precision Engineering and Manufacturing-Green Technology* 5 (4): 535–54. https://doi.org/10.1007/s40684-018-0055-0.

Teoh, Yyi Kai, Sukhpal Singh Gill, and Ajith Kumar Parlikad. 2021. "IoT and Fog Computing Based Predictive Maintenance Model for Effective Asset Management in Industry 4.0 Using Machine Learning." *IEEE Internet of Things Journal* 0 (c): 4–11. https://doi.org/10.1109/JIOT.2021.3050441.

Vlasov, Andrey I, Pavel V Grigoriev, Aleksey I Krivoshein, Vadim A Shakhnov, Sergey S Filin, and Vladimir S Migalin. 2018. "Smart Management of Technologies: Predictive Maintenance of Industrial Equipment Using Wireless Sensor Networks." *The International Journal Enterpreneurship and Sustainability Issues* 6 (2): 489–502. https://doi.org/http://doi.org/10.9770/jesi.2018.6.2(2).

Xu, Hansong, W E I Yu, David Griffith, and Nada Golmie. 2018. "A Survey on Industrial Internet of Things : A Cyber-Physical Systems Perspective." *IEEE Access* 6: 78238–59. https://doi.org/10.1109/ACCESS.2018.2884906.

Xu, Yan, Yanming Sun, Jiafu Wan, Xiaolong Liu, and Zhiting Song. 2017. "Industrial Big Data for Fault Diagnosis: Taxonomy, Review, and Applications." *IEEE Access* 5 (July): 17368–80. https://doi.org/10.1109/ACCESS.2017.2731945.

Yan, Jihong, Yue Meng, Lei Lu, and Lin Li. 2017. "Industrial Big Data in an Industry 4.0 Environment: Challenges, Schemes, and Applications for Predictive Maintenance." *IEEE Access* 5: 23484–91. https://doi.org/10.1109/ACCESS.2017.2765544.

Yang, Li, and Abdallah Shami. 2020. "On Hyperparameter Optimization of Machine Learning Algorithms: Theory and Practice." *Neurocomputing* 415: 295–316. https://doi.org/10.1016/j.neucom.2020.07.061.

Yu, Wenjin, Yuehua Liu, Tharam Dillon, Wenny Rahayu, and Fahed Mostafa. 2021. "An Integrated Framework for Health State Monitoring in a Smart Factory Employing IoT and Big Data Techniques." *IEEE Internet of Things Journal* X (X): 1–12. https://doi.org/10.1109/JIOT.2021.3096637.

3. DATA COLLECTION AND PROCESSING FOR SYSTEM HEALTH MANAGEMENT IN SMART FACTORIES

Since the first generation of maintenance strategies, for which data were collected manually from the operator experience and their storage, analysis, and transfer only relied on human abilities and verbal communication, data have always been crucial for optimizing the system health management (Jasiulewicz-Kaczmarek, Legutko, and Kluk 2020). Models and algorithms to extract knowledge from the data, and technologies for their collection and storage, are critical choices in this process. However, the effectiveness of the maintenance strategy also depends on specific components chosen for the analysis and the kind of data collected for selected components. In other words, a maintenance strategy may be suitable for a component and not suitable for another one, and the choice also depends on the efforts needed to get the right data for implementing a specific strategy. In particular, given the high investments required by Predictive Maintenance, e.g., sensor technologies and resources for data analysis, the first step towards its implementation is the selection of critical components and signals to collect.

Although the methods for identifying critical components have been studied and proposed primarily for reliability-centered maintenance, they are also adequate to select critical components for Predictive Maintenance. The most common approaches proposed in the literature are the Failure Mode, Effects and Criticality Analysis (FMECA), the Fault Tree Analysis (FTA), the Event Tree Analysis (ETA), the experience feedbacks, and others, which can rely on qualitative and quantitative information and adopt inductive or deductive kind of reasoning (Sarih et al. 2018). Therefore, the criticality of a component is often measured through the effect of components' failure on the productivity and environment and can be evaluated according to multiple criteria, i.e., fault severity, fault occurrence, and fault detectability, which consider both the technical and the economic aspects (Ji et al. 2019). According to (J. Lee et al. 2014), the critical components for which Predictive Maintenance represents the best solution for their health management are those with the most downtime, although their failure occurs infrequently. For instance, Lee et al. (2018) conducted a qualitative evaluation of critical components based on an extensive review on papers addressing fault diagnosis and prognostics in smart factories. They identified six main components suitable for Predictive Maintenance, i.e., the rotary component, machining and micro machining tool or process, electric component, components in non-conventional manufacturing machines, and other components. In particular, gears, bearings, and shafts represent the most frequently monitored component for the machine health assessment in the rotary components group. Among various machining tools, milling and turning tools' failures have the most significant impact on the machinery availability. In the electric components group, the electric motor is considered the most critical component.

Once critical components have been selected for Predictive Maintenance, choosing or identifying the signals providing helpful information on their health condition is necessary.

According to (Sajid et al. 2021), the types of data used for Predictive Maintenance are maintenance history (73%), asset usage (72%), asset condition (71%), condition data and maintenance history of other assets within the company (42%), environmental data (29%), condition data and maintenance history of other assets from other companies (9%), others (7%). Tiddens et al. (2020) grouped the data required for Predictive Maintenance into four categories: (1) asset history data, which are gathered from technical knowledge, inspections, and historical records of failures or costs; (2) usage and process data, which entail operational data, e.g., running hours, mileage or tons produced; (3) stressor data, which describe the exerted loads (stressors) on the system, including environmental data; (4) Condition Monitoring data, which provide signs of imminent failure of the equipment. Although all these data may contribute to implementing a Predictive Maintenance strategy, intelligent fault diagnosis and prognostics only rely on Condition Monitoring data in practice. In particular, vibration signals have been used most frequently for machinery health management, followed by acoustic emissions, force/torque, temperature, and electric signals (G. Y. Lee et al. 2018). In few cases, Big Data information provided by machine log systems, without installing sensors into the machines to collect specific Condition Monitoring data, are also considered in Predictive Maintenance (M. Calabrese et al. 2020). However, the number of studies exploring other types of data for Predictive Maintenance and the industrial applications that exploit the Industry 4.0 technologies for intelligent maintenance are limited.

For this reason, in this dissertation, several industrial case studies are reported, which focus on data collection and data processing in order to provide a screenshot of the state-of-the-art of Predictive Maintenance in industries of Northern Italy and highlight the constraints to overcome to make data exploitable by companies and make possible the transition towards Smart factories. The first case study concerns a system operating in a controlled environment, meaning that proper tests and simulation in a laboratory have been performed considering the goal of the analysis and recording all related data. The other case studies consider data provided by companies that collaborate with the Department of Industrial Engineering of the University of Bologna. Industrial environments are considered non-controlled environments since the data may be noisy and unreliable, and much information may be hidden because of privacy issues or is unavailable. Based on the collected datasets, the data that will be considered in this dissertation are classified as follows:

1. Condition Monitoring data: they include signals like vibrations, currents, pressures, temperatures, collected continuously in a certain period through proper sensors during machinery functioning;
2. Environmental data: they include the parameters that determine a specific setting, the type of product or material processed during data collection, external temperature or humidity;
3. Event data: they include all the actions to which the component/machinery was subject to, like maintenance interventions and sudden or planned shutdowns;
4. Domain knowledge information: they include all information related to the machine functioning provided by experts in the technology.

Therefore, after a literature review on data collection infrastructures and data processing methods, section 3.2 will provide, for each case study, the information on the data collection process, including the description of the available data, their format, and how they have been collected, transferred, and pre-processed for the subsequent analysis. Then, section 0 will provide the analysis results, grouped for the task to accomplish, highlighting the advantages of each task and data processing methods. Finally, in section 3.4, the outcomes of the data collection and processing steps will be presented.

3.1 Literature review and theoretical background

Condition Monitoring data, like force, vibration, temperature, voltage, and others, which indicate the health condition/state of the equipment under a given load profile, are fundamental to build a fault diagnosis and prognosis model, estimate its parameters and verify/validate its maturity (Javed, Gouriveau, and Zerhouni 2017). Since raw data are redundant and noisy, and the relevant information linked to the degradation process is usually hidden in raw data, they cannot be directly used for fault diagnosis and prognosis (Javed, Gouriveau, and Zerhouni 2017). Therefore, when using a data-driven approach for diagnostics and prognostics, processing and transforming raw signals is a fundamental step (Sarih et al. 2019). In particular, it is crucial to extract some characteristics from raw signals, named features, which should have two main properties. First, they have to be relevant, meaning that they reveal the health condition of the system under analysis; second, they have to be non-redundant, in order to reduce the computational complexity of Machine Learning (ML) algorithms used for the subsequent activities of diagnostics and prognostics.

In (Lei et al. 2020), the authors divide the studies on data processing into two categories, according to the evolution stage of Machine Learning theory:

1. Artificial feature extraction, which consists of two steps: feature extraction and feature selection. First, time, frequency, and time-frequency-domain analysis are conducted to extract common-used features from collected data; then, supervised Machine Learning algorithms are trained to select the most sensitive features from the extracted features; this approach to data processing is considered belonging to past practices, when fault diagnosis was addressed with traditional Machine Learning algorithms;
2. Deep Learning-based fault diagnosis, in which features are automatically learned from collected data in an unsupervised way. Therefore, end-to-end diagnosis models are constructed to learn features directly from raw data, and recognize the health state of machines by using Deep Learning theories; this approach belongs to the current practice, in which Big Data assume a relevant role and domain knowledge is decreased due to the increased complexity of systems.

Indeed, the traditional approach, i.e., artificial feature extraction, depends on prior knowledge of signal processing techniques, and shallow Machine Learning models limit their ability to learn complex non-linear relationships and handle Big Data. On the contrary, Deep Learning can effectively overcome the two issues above due to its automated learning process. However, their performance depends substantially on the quality and quantity of the data. Moreover, they require a large number of calculations during the training process and a large number of hyperparameters settings (Chen et al. 2020). For this reason, data processing for fault diagnosis and prognostics in industries can still benefit from the traditional approach, which includes two main steps, i.e., signal processing and dimensionality reduction. Signal processing involves extracting the features in one of the signal analysis domains, i.e., time, frequency, or time-frequency domain. Dimensionality reduction reduces the number of variables of a dataset and may be directly applied to raw signals or the previous extracted feature set.

Feature extraction and dimensionality reduction notably reduce the amount of data, which may have an important implication from the data sharing and storage perspective. Through the IIoT architecture, it is possible to collect the data from all machines on a shop floor and query streaming data used for Machine Learning models (Ayvaz and Alpay 2021). They can be trained by resorting to the Cloud Computing paradigm, which represents a hosting platform that provides diagnosis and prognosis solutions as a service (Fila, El Khaili, and Mestari 2020). However, the cloud may have high response times due to the long-distance transmission of massive data volumes, which is not suitable for fault detection systems for which real-time responses are essential (Fawwaz and Chung 2020). Performing data processing at the edge may solve this issue since only features extracted at the edge may be sent to the cloud (C. Cheng, Zhang, and Gao 2019). In addition, the reduced amount of data resulting from the transmission into the cloud of relevant features only may allow an easier aggregation of data collected from similar machines installed in different plants, obtaining larger datasets for models training. Therefore, a proper infrastructure including sensor networks, edge devices and cloud data centres should be built in order to distribute the computing among different "devices" and reduce the latency of both transferring and inference as well as the required storage memory.

3.1.1 IIoT architectures for data collection

According to (S.-W. Lin et al. 2017), a general IIoT system can be decomposed into five functional domains, among which data flows and control flows take place (Figure 16). As the information moves up in the functional domains, the scope of the information becomes broader and more general, new information can be derived, and new intelligence may emerge in the larger contexts. The control domain includes the functions performed by industrial control and automation systems, like sensors and actuators. Components or systems implementing these functions are usually deployed in proximity to the physical systems they control and may be geographically distributed. The operation domain is responsible for the management and operation of the control domain. Functions in this domain include prognostics, optimizations, monitoring and diagnostics, previsioning & deployment, and asset management. At this level, the set of functions should support major automation and analytics features, including (a) automated data collection, processing, and validation, (b) the ability to capture and identify significant events, such as downtime, delay, and (c) the ability to analyze and assign causes for known problems. Furthermore, many of the core functions in the operations domain may require performing advanced analytics on potentially large volumes of historical asset operational and performance data. The information domain is a functional domain for managing and processing data. It represents the collection of functions for gathering data from various domains, most significantly from the control domain, and transforming, modeling, and analyzing those data to acquire high-level intelligence about the overall system. These functions can be used in online streaming mode or offline batch mode. The application domain represents the collection of functions implementing application logic that realizes specific business functionalities. Functions in this domain apply application logic, rules, and models at a high level for optimization in a global scope. Finally, the business domain represents business functions supporting business processes and procedural activities business functions that an IIoT system must integrate to enable end-to-end operations of IIoT systems. Examples of these business functions include Enterprise Resource

Planning (ERP), Customer Relationship Management (CRM), Product Lifecycle Management (PLM), Manufacturing Execution System (MES), Human Resource Management (HRM), asset management, service lifecycle management, billing, and payment, work planning, and scheduling systems.

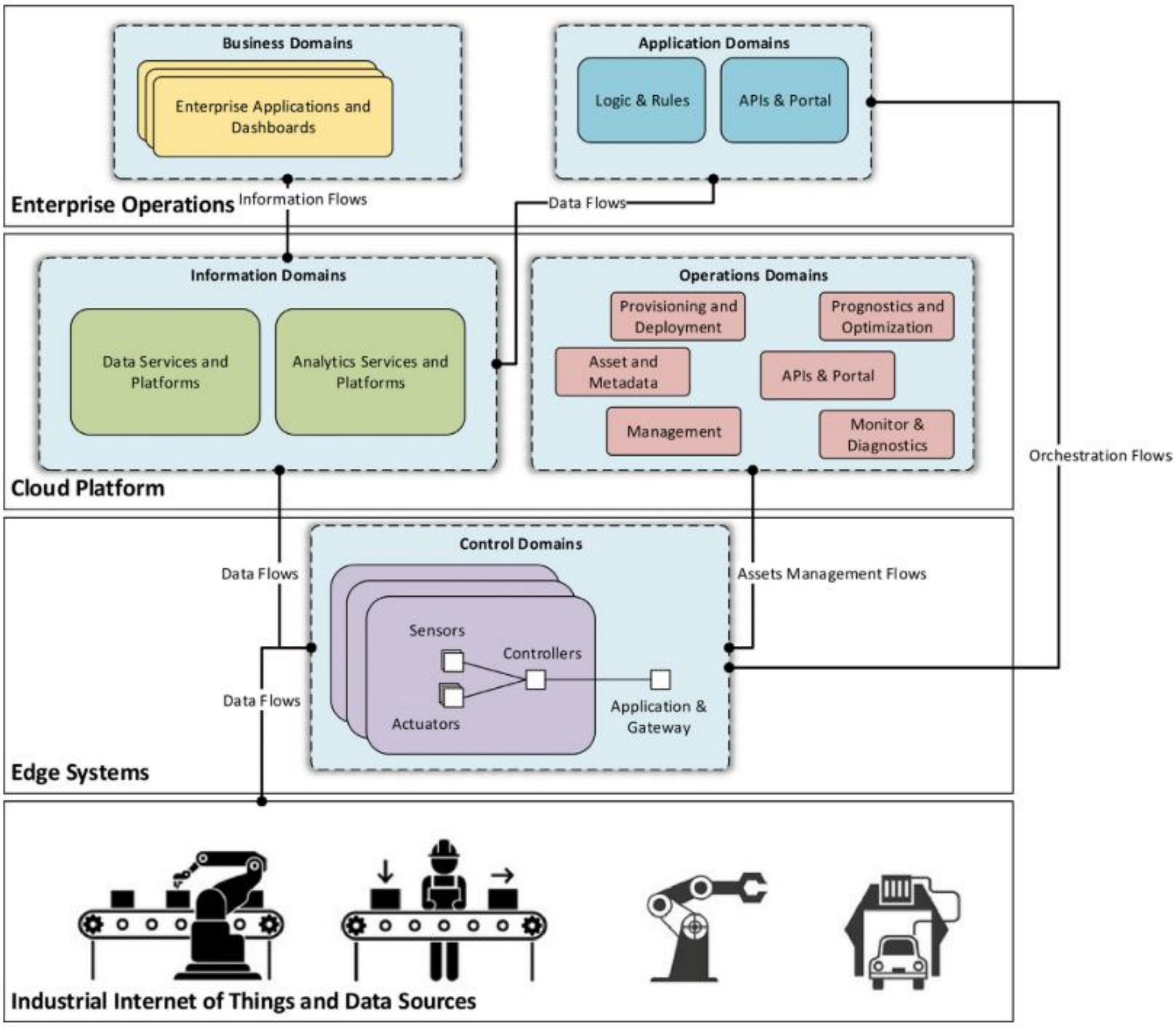


Figure 16 A general architecture for IIoT systems (W. Z. Khan et al. 2020)

An IIoT platform for Condition Monitoring and Predictive Maintenance follows the above-described architecture. In (Siddhartha et al. 2020), an IIoT architecture for real-time Condition Monitoring of a CNC machine is proposed, in which the five layers are named as machine layer, IoT layer, Internet Layer, Data Processing Layer, and Management Layer. In addition, a hierarchy made of four kinds of users of the designed system is also introduced. The plant head has complete access to all the machines' data analytic reports, oversees all the daily operations, division of work between managers, and makes sure all the procedures are followed. The Manager tracks the work progress and the data analysis. The supervisor oversees the daily operations on the shop floor, creates work schedules, monitors and evaluates the performance. Finally, operators set up and adjust machines, monitor them for unusual noises or movements, and execute the scheduled tasks to ensure the factory's equipment functions efficiently and that all procedures work correctly.

Xiaoli et al. (2011) proposed an Intelligent Internet of Things for Equipment Maintenance (IITEM), which guarantees the safe operation of critical equipment, performs specialized remote diagnosis, realizes online monitoring and transmission of dynamic operating parameters, conducts real-time analysis and assessment of the operating status, and makes timely and automatic alarm on the status when exceeding the limited value. In addition, the system makes early predictions of the potential equipment failure, predicts when the work status reaches an unacceptable level, and the equipment should be shut down for maintenance. Finally, the proposed IITEM offers, through the information network, the technical service for remote equipment fault diagnosis and scientific maintenance, provides the feedback control signals for optimal control of the operating status of equipment, implement the optimal control of equipment running status, and make the equipment work in the security zone or under the energy-saving and environmental protection state.

In (Balogh et al. 2018), a cluster of tools and services have been conglomerated in a distributed cloud, which encompasses cyber-physical systems, IIoT, M2M, big data analytics, data mining, predictive models, machine learning, and cloud technology for the establishment of a successful collaborative predictive maintenance framework, focusing on three main goals: data collection frameworks and interfaces, data analytics and methodologies for health monitoring and predictive maintenance, maintenance planning and scheduling, collaborative cloud analysis platform.

Bellavista et al. (2019) introduced a layered architectural model for collecting and analyzing the data collected from manufacturing machines to bring Small and Medium Enterprises (SMEs) closer to the Industry 4.0 transition. In particular, the proposed architecture includes three layers: (1) the machine layer, the Information Technology Layer, and the Operation Technology Layer. The proposed architecture is particularly suitable for data sharing between the vendor and his customers. Indeed, separating the machine layer from the higher layers allows achieving the desired performance levels and security and safety. Information security also embraces proprietary data extracted from the machines that generally vendors want to keep private. Strict access mechanisms in this layer prevent malicious intrusions and theft or alteration of application logic that could lead in their turn to dangerous situations or information leaks. Finally, the division of information in the OT and IT levels reflects the different needs of the technical and managerial departments. The managerial departments are often not interested in viewing and storing all the information coming from the manufacturing machines. On the contrary, technical departments are typically interested in the complete history that allows evaluating complex inferences helping identify critical parts of physical machines and improve production processes based on the actual operating values and not on those statically foreseen in the development phase.

The described IIoT architectures present all the characteristics and functionalities needed to implement intelligent health management in a smart factory. In particular, while the first and the last architectures are general and refer to an IIoT system that serves several objectives, e.g., process monitoring and predictive maintenance, the other two architectures are more maintenance-oriented. In addition, the last architecture also considers the possibility to distribute the system among different actors. Therefore, there is no generalizable architecture suitable for all contexts. On the contrary, the company's specific requirements and expectations guide the selection of the elements and actors to include in an IIoT system.

3.1.2 Signal analysis

Manual feature extraction is typically conducted in the time, frequency, and time-frequency domain (Goyal and Pabla 2015). One of the most common approaches in the time domain is the extraction of statistical features. The idea is that if a change occurs in the signal because of a variation in the operating condition, the Probability Density Function (PDF) of signal samples and its statistical parameters change accordingly (Mehrjou et al., 2017). Typical statistical features are peak, peak-to-peak, mean, variance, skewness, kurtosis, Root Mean Square (RMS), crest factor, root amplitude, shape factor, and impulse factor. Mean, variance, skewness, and kurtosis correspond to the first, second, third, and fourth statistical moment of the PDF of signal samples, respectively; in particular, skewness measures the asymmetry of the probability distribution about its mean, while kurtosis provides information about tails and peaks of the probability distribution (Gomes Teixeira de Almeida, Alexandra da Silva Vicente, and Rodrigues Padovese 2002). The RMS is a measure of the signal energy intensity, while the crest factor is a measure of the number and sharpness of peaks in the signal. In Table 3, the mathematical formulations of the most common statistical features in the time domain are provided.

Signal processing in the time domain is easy and requires low computational complexity, making it more suitable for real-time calculations. However, many critical components show characteristic frequencies for some fault modes. In these cases, frequency-domain analysis is preferred, as it easily allows identifying and isolating frequency components of interest. Frequency-domain analysis is often based on Fourier Transform of the signal, from which the power spectrum, higher-order spectra, and cepstrum, can be obtained (Liang, Iwnicki, and Zhao 2013). Instead, the envelope analysis is based on the Hilbert transform and is a reliable demodulation technique for detecting and isolating many faults in bearings, gear boxes, diesel engines, hitting machine parts, and even the imbalance and the misalignment when they are extreme (Geropp 1997).

Table 3 Statistical time-domain features

Feature	Formula	Feature	Formula
Peak	$x_{max} = max\|x_i\|$	Kurtosis	$\frac{1}{N}\sum_{i=1}^{N}\frac{(x_i - \bar{x})^4}{\sigma^4}$
Peak-to-peak	$\|max(x_i) - min(x_i)\|$	Skewness	$\frac{1}{N}\sum_{i=1}^{N}\frac{(x_i - \bar{x})^3}{\sigma^3}$
Mean value	$\frac{1}{N}\sum_{i=1}^{N} x_i$	Shape Factor	$RMS/Mean$
Root Mean Square (RMS)	$\sqrt{\frac{1}{N}\sum_{i=1}^{N} {x_i}^2}$	Impulse Factor	$x_{max}/Mean$
Crest Factor	x_{max}/RMS		

Although both time and frequency analysis often provide pretty good results, signals generated by a machine are non-stationery in most cases, and the Fourier Transform cannot be calculated. The time-frequency-domain analysis is mandatory in these cases. In addition, non-stationary components of signals contain rich fault-related information, and the time-frequency-domain analysis provides a more interpretable representation of the signal. Typical time-frequency analysis techniques are the Wigner-Ville Distribution (WVD), the Short-Time Fourier Transform (STFT), and the Wavelet Transform (WT). WVD and STFT are time-frequency distributions representing the energy or the power of signals into a two-dimensional function of both time and frequency. WT can be used for multi-scale analysis of a signal since its time-frequency resolution depends on the signal's frequency (Z. K. Peng and Chu 2004). The WT is a time-scale transform expressing the signal in a series of oscillatory functions (sons) with different frequencies at different times by dilations and translations from the wavelet prototype (mother wavelet). Continuous wavelet transforms (CWT) can be performed based on different wavelet bases (L. Li, Qu, and Liao 2007), (J. Lin and Qu 2000). Because of its good energy concentration properties, WT can present the signal with a minimal number of coefficients, which measure the similarity between the signal and each of its son wavelets; these coefficients can be effectively used as fault features. A continuous wavelet transform is defined by Eq. (1)

$$W(a,b) = \frac{1}{\sqrt{a}}\int_{-\infty}^{+\infty} x(t)\psi^*\left(\frac{t-b}{a}\right)dt \tag{1}$$

where $x(t)$ is a waveform signal, a is the scale parameter, b is the time parameter, ψ is a zero-average oscillatory function centered around zero with finite energy, and ψ^* is its complex conjugate. The series of wavelets with different frequencies at different times are obtained by dilations of the scale parameter and translation of the time parameter. In order to interpret the signal, the scalogram, defined as $|W(a,b)|^2$, and the wavelet phase spectrum, defined as the phase angle of the complex variable $W(a,b)$, are used. Continuous WT can be performed based on different wavelet bases, which may lead to different results. One of the most commonly used wavelets is the Haar wavelet, which has good low-pass filter characteristics and is mainly used for the fault diagnosis of rotors, gearboxes, and rolling bearings. Eq. (2) expresses the standard Haar wavelet

$$\psi(t) = \begin{cases} 1 & 0 \le t \le \frac{1}{2} \\ -1 & \frac{1}{2} \le t \le 1 \\ 0 & otherwise \end{cases} \tag{2}$$

Since the Haar wavelet is discontinuous in amplitude, it is a poor feature detector and is rarely used in machine diagnosis.

Sometimes, the Morlet wavelet is used. It has a band-pass characteristic, and it is a good detector of high-frequencies The Morlet wavelet is given by Eq. (3)

$$\psi(t) = e^{-\beta(t^2/2)} e^{j\omega_0 t} \tag{3}$$

where the parameters $\beta = \omega_0^2$ and ω_0 are defined in the particular application.

Besides feature extraction, WT, and in particular the discrete wavelet transform (DWT) and the wavelet packet transform (WPT), are also used as signal pre-processing techniques due to their powerful signal denoising ability (Gokhale and Khanduja 2010).

Recently, many researchers have used a new time-frequency analysis technique named Empirical Mode Decomposition (EMD). It consists of a time adaptive decomposition of the signal into a set of almost complete, almost orthogonal, and almost mono-component components, named Intrinsic Mode Functions (IMFs), from which energy features can be extracted (Ricci and Pennacchi, 2011). From the obtained IMFs, it is possible to extract the energy of the signals in different frequency bands that change when a fault condition occurs (Bin et al., 2012). The decomposition starts with the research of the maxima and the minima along with the signal $x(t)$. Then, the maxima and minima are interpolated by means of two splines, $s_{max}(t)$ and, $s_{max}(t)$, that represent the boundaries of the signal. Finally, the mean function $m(t)$ is extracted and removed from the original signal, obtaining the new signal $x_1(t) = x(t) - m(t)$. This signal is an IMF if satisfies the following conditions (Rai and Upadhyay 2017): (1) There must be equality or a difference of at most one between the number of extrema and the number of zero crossings, and (2) the envelope defined by the local maxima and the envelope defined by the local minima must have zero average. The procedure is repeated until the obtained signal satisfies those conditions. When the conditions are satisfied, then the first IMF, $C_1(t)$, is obtained and subtracted from the original signal, obtaining the residual signal $r_1(t) = x(t) - C_1(t)$. The residual signal represents the input for the second IMF computation. The process stops when the residual signal is a constant or a monotonic function (Ricci and Pennacchi 2011). From the obtained IMF, the energy features can be extracted by using Eq. (4):

$$E_j = \sum_{k=1}^{n} C_i|(k)|^2 \tag{4}$$

where E_j is the energy of the jth subband for the IMF C_i, and k represent the data samples.

Although domain-specific features are often used, the choice of the analysis domain requires knowledge and expertise about vibration signals. For each component, a deep study has to be conducted to identify the most representative features, which is an expensive and time-consuming activity that makes it difficult the online monitoring and its application in real industrial contexts. However, when no other options are available, for instance, because the physical meaning of relevant features wants to be kept or training data are few, relevant features may be selected through feature selection techniques. In other cases, dimensionality reduction techniques can be applied to avoid hand-crafted feature extraction. The next section will present the methods of dimensionality reduction for fault diagnosis.

3.1.3 Dimensionality reduction

In the context of data science, reducing the number of variables in a dataset is referred to as Dimensionality Reduction (DR). It can be performed by feature selection or feature learning methods. In the first case, the dimension of the dataset is reduced by selecting the subset of features that most contribute to the classification accuracy and eliminating those that contain redundant information. In the second case, original features are subject to different transformations to produce features with more informative content (Tang, Alelyani, and Liu 2014).

The basic idea of feature learning is to learn discriminative and effective features from massive data automatically. These features may not have a direct physical meaning as time, frequency, and time-frequency features, but can capture explanatory information hidden in the raw data, augment the essential information for discrimination, and suppress irrelevant variations (Lei et al. 2016). Traditional feature learning methods are Principal Component Analysis (PCA) (F. Wang et al., 2015) and Linear Discriminant Analysis (C. Zhang et al., 2018), that map all features into a low-dimensional space and remove the noise and outliers inhere in data, obtaining in this way robust learning models and stable performance. However, these methods have limitations in nonlinear transformations. Recently, manifold learning methods, such as Locality Sensitive Discriminant Analysis (LSDA) and Locality Preserving Projection (LPP) (X. Zhang et al. 2017), are emerging. They identify the structure of low-dimensional spaces (manifolds) embedded in the high-dimensional input space, near which this data is likely to concentrate, thus preserving the local neighborhood structure (Gan, Wang, and Zhu 2015). Feature learning can also be seen as a representation tool, which learns representations and transformations of the data, making it easier to extract useful information (Bengio, Courville, and Vincent 2013). As representation learning tools, sparse representation theory and deep learning are attracting more and more attention in machinery fault diagnosis. Sparse representation is a signal reconstruction method representing the input signal by a sparse linear combination of atoms chosen from a collection called a dictionary. In this way, only a small proportion of atoms will contribute to approximate the input signal (Han et al. 2017). This process is an NP-hard optimization problem, often solved by efficient greedy methods, such as matching pursuit (MP) and its variations. Unlike conventional basis representation models, the dictionary provides a broader array of basis functions in sparse representation, which offers more flexibility in signal representation. Traditional dictionaries included a manually set of basis functions, such as STFT or WT. However, they could not sufficiently match the characteristics of decomposed signals. Therefore, a dictionary learning process could be employed to capture statistical features in the signal as the basis function, in such a way that the dictionary is directly learned from data themselves (H. Liu, Liu, and Huang 2011). An easy and hyper-parameter-free method for unsupervised feature learning, i.e., sparse filtering, is proposed in (J. Wang et al. 2017), which only focuses on optimizing the sparsity of the learned representations and ignores the problem of learning the data distribution. Deep learning represents a breakthrough in feature learning. Deep architectures perform a series of nonlinear transformations to automatically learn a hierarchy of features representing data in a more and more abstract way. These architectures can be seen as a way to decompose the problem of feature extraction into sub-problems with different levels of abstraction. Indeed, if one-layer is able to extract relevant features, it is intuitive that a second layer taking as input those features could extract slightly higher-level features (Lillicrap et al. 2009).

Feature selection methods are applied to a set of features previously extracted in order to obtain the smallest subset of features in which only relevant and non-redundant features are selected. A so-build subset has a strong positive influence on several aspects of the following machine learning model: the prediction accuracy, the generalization ability, the over-fitting issue (that will be explained in the next chapter), the computational time, and the required storage. Features selection methods can be grouped into supervised, unsupervised, and semi-supervised methods. Supervised learning requires a set of observations associated with both descriptive attributes (features) and a pre-specified target variable (class or label) that, in this context, corresponds to the health/fault class to which features belong. Therefore, supervised methods determine the importance of a feature by evaluating its correlation with the fault classes, while unsupervised methods select features with the maximum representative and discriminant power, only depending on their values. Semi-supervised methods use both a limited number of labeled samples and a large number of unlabelled samples. Feature selection methods can also be divided into filter, wrapper, and embedded methods. Filter methods evaluate features by assigning a score to each of them depending on the general characteristics of the data. Then, they select the features with higher scores. The computational cost of filter methods is relatively low, and they are pretty robust to the problem of over-fitting. However, as they do not involve any classification algorithm, they ignore the effects of the selected features on the classification performance. Wrapper methods make use of classification algorithms for assessing the quality of the extracted features. First, a subset of features is extracted and evaluated based on the accuracy of a predefined classifier; then, the process is repeated with different subsets, and the feature subset with higher accuracy is selected. This process is an NP-hard problem, and different search strategies have been proposed in the literature. Genetic algorithms are one of the most adopted approaches in wrapper models (Cerrada et al. 2015). Although selected features usually have good performance, wrapper methods are computationally expensive and prone to over-fitting. Finally, embedded methods perform feature selection and classification tasks into a single optimization problem, considering feature selection as part of the training process. Therefore, they combine the advantages of both filter and wrapper methods, as they are less computationally expensive and consider the classification performance. Finally, embedded methods are also used to construct one or more optimal features in a supervised way. Among them, Genetic Programming (GP) is receiving significant attention. The difference with more traditional methods lies in the output, which is a combination of original features rather than one or more original features. Finally, unsupervised filter, wrapper, and embedded feature selection methods have also been proposed (Xie et al., 2018). Unsupervised filter methods, e.g., Laplacian Score, assign to each feature a score indicating its capacity to preserve the structure of data. Unsupervised versions of wrapper methods use clustering techniques instead of classification models (D. Wang, Nie, and Huang 2014). In embedding methods, feature selection is often combined with subspace learning models, such as Principal Component Analysis (Yonghua Zhu et al. 2018a), sparse learning (R. Hu et al. 2017), or manifold learning (X. Zhu et al. 2017), in order to make feature selection more stable and to enable objective functions to achieve global optimum.

At this point, basic concepts and mathematical formulations of some common methods are provided. They are the Pearson Correlation analysis, the Recursive Feature Elimination, The Principal Component Analysis and its incremental version, and Genetic Programming. The reason why of describing in detail these methods lies in the application of these methods to the case studies presented in this dissertation.

Pearson Correlation analysis is a filter-based method used for redundancy elimination. It assumes that if two variables have a high linear relationship, they contain redundant information (X. Cheng et al. 2019). A fixed threshold usually determines if features are highly correlated or not. In the present paper, first, the Pearson correlation is conducted, and a threshold equal to the absolute value of 0.9 is chosen. Then, the correlation matrix is first built using the Pearson correlation coefficient computed among all features and expressed by Eq. (5)

$$r_{XY} = \frac{Cov(X,Y)}{\sqrt{Var(X)Var(Y)}} \tag{5}$$

where, X and Y are two random variables, $Cov(X,Y)$ is the covariance of the two variables, and $Var(X)$ and $Var(Y)$ are the variance of the variable X and Y, respectively. The Pearson correlation coefficient, r_{XY}, is always included in the range $[-1,1]$, where -1 indicates that the two variables are negatively correlated, 1 indicates that the variables are positively correlated, and 0 that the variables do not correlate. After the correlation matrix construction, an iterative process for feature elimination is conducted. Hence, for each column, and for each row, if $r_{XY} \leq -0{,}9$ or $r_{XY} \geq 0{,}9$, the column is eliminated.

The Recursive Feature Elimination is a supervised and wrapper feature selection method, which assigns a weight to the features based on a particular classification model and recursively removes the features with the most negligible weight (Su, Liu, and Tao 2020). When the Support Vector Machine is used as the classification model, the RFE consists of the following steps:

1. Given a set of training samples $\{x_i, y_i\}$ $i = 1, \dots, N$, where $x_i$is the feature vector sample with S features, y_i is the label associated with the sample x_i. A linear SVM model is trained using these training samples, and its decision function is given by Eq. (6):

$$f(X) = \omega \cdot X + b \tag{6}$$

2. According to the trained SVM model, the ranking criterion of features can be calculated as follows:

$$\omega = \sum_{i=1}^{N} \alpha_i x_i y_i \tag{7}$$

$$J(s) = \omega_s^2 \tag{8}$$

Where α_i is the Lagrange multiplier, ω is the weight vector of the trained SVM model with S elements ω_s, $J(s)$ is the ranking criterion for the feature $s(q)_s$.

3. Eliminate the feature $s(q)_k$ with the lowest criterion from the feature set

$$s(q)_k = argmin(J) \tag{9}$$

$$SF^1 = \{s(q)_1, s(q)_2, \dots, s(q)_{k-1}, s(q)_{k+1}, \dots, s(q)_S\} \tag{10}$$

Where SF^1 is the feature subset with $S - 1$ features in the first iteration of feature selection

4. Repeating the above operations until all the features are eliminated from the feature set, obtaining the S feature subsets:

$$SF^1 = \{s(q)_1, s(q)_2, \dots, s(q)_s, \dots, s(q)_S\} \tag{11}$$

using the cross-validation technique, the feature subset SF^* with the best accuracy will be selected as the optimal feature subset for the subsequent fault diagnosis.

The Principal Component Analysis (PCA) (Yonghua Zhu et al. 2018) is a feature learning method. It is a statistical analysis approach to mapping multiple characteristic parameters to a few comprehensive features. These PCA-based comprehensive features are not related to each other and can represent original fault features effectively (F. Wang et al. 2015). Given a dataset X, of dimension m, PCA aims to find a set of orthonormal basis vectors of dimension $p < m$, which are called Principal Components (PCs), that maximize the variance over the dataset when it is projected onto the subspace spanned by these PCs. Basically, if we have data points in a two-dimensional space and we want to project them in one-dimensional space, what PCA does is to find the direction of the vector and the position of the points on that vector, which is expressed by coefficients, such that the reconstruction or projection error is minimized. To this aim, the covariance matrix of the dataset is first computed, and the eigenvalues and eigenvectors are extracted. The eigenvectors correspond to the PCs, while the corresponding eigenvalues represent the variance associated with that PC. The PCs are selected so that the cumulative variance described by them is higher than a certain percentage (usually, from 90 to 99%). Lippi & Ceccarelli (2019) presented an exact incremental implementation of PCA. As the authors state, exact means it provides the same results, i.e., the same PCs as in the batch version. In addition, it also contains an online data normalization, which is fundamental when variables assume very different values. Basically, the difference between the batch PCA and its incremental version formulated in that paper lies in the covariance matrix computation, which is recursive. The steps of the algorithm are the following:

1. The sample mean $\bar{x}_{n(j)}$and the standard deviation $\bar{\sigma}_{n(j)}$ are computed for each variable j ($j = 1, \dots, m$) over the first n available observations, in order to compute the standardized matrix Z_n as follows

$$Z_n = \begin{bmatrix} x_1 - \bar{x}_n \\ \dots \\ x_n - \bar{x}_n \end{bmatrix} \Sigma_n^{-1} \tag{12}$$

   Where $\Sigma_n \equiv diag(\sigma_n)$ is an $m \times m$ matrix

2. The covariance matrix Q_n of the data matrix $X_n$is computed as follows

$$Q_n = \frac{1}{n-1} Z_n^T Z_n \tag{13}$$

3. The standard diagonalization of Q_n is made by means of the eigenvector matrix C_n as follows

$$Q_n = C_n^{-1} \begin{bmatrix} \lambda_1 & & \\ & \dots & \\ & & \lambda_m \end{bmatrix} C_n \tag{14}$$

   Where the eigenvalues λ_i are put in descending order and express the variance associated with the *i*th Principal Component (PC), that is the *i*th eigenvector of C_n.

4. Finally, the time evolution of PC values until the time stamp n is computed as

$$PC_n = Z_n C_n \tag{15}$$

5. At the step $n + 1$, the mean and the standard deviation are updated and the standardized matrix Z_{n+1} is computed as follows

$$Z_{n+1} = \begin{bmatrix} Z_n \Sigma_n + \Delta \\ y \end{bmatrix} \Sigma_{n+1}^{-1} \tag{16}$$

   Where $y = x_{n+1} - \bar{x}_{n+1}$, Δ is a $n \times m$ matrix made of repeating n times the vector $\delta = \bar{x}_n - \bar{x}_{n+1}$

6. The covariance matrix nQ_{n+1} is computed ad follows

$$nQ_{n+1} = Z_{n+1}^T Z_{n+1} \tag{17}$$

   Which only depends on the covariance matrix computed at the point n and the new feature vector $\boldsymbol{x}_{n+1}$.

Finally, the updated Q_s are used to compute the *n*th values for the evolving PCs by means of Eq. (4).

Many AI optimization algorithms for the research of a maximum (or minimum) value of a function work in a finite domain, considering multiple constraints on the solution set and having issues if the objective function has multiple local maxima or non-linearity trends (Muni, Pal, and Das 2004). These algorithms require an unacceptable amount of time to reach the optimal solution. Hence, the attention moves towards heuristic algorithms, which can guarantee sub-optimum solutions to the problem in a reasonable time. One of the most effective categories of heuristic algorithms is represented by Evolutionary Algorithms (EA), which have their fundamentals in Darwin's evolutionary theory. As such, Genetic Programming (GP) is receiving great attention, especially in feature selection and construction for classification problems (Neshatian, Zhang, and Andreae 2012). GP emulates the evolution of the population's individuals through genetic operators (Guo et al. 2011). The individuals represent the possible solutions belonging to a population of a specific size. Their strength is evaluated by their ability to adapt to the environment. A fitness function measures this ability. Thus, only individuals with high values of the fitness function survive during the construction process. The main steps of GP can be summarized as follows (Vanneschi and Poli 2012):

1. First, an initial population of individuals is randomly generated
2. Then, the following steps are performed until a certain termination criterion is met
    a. A fitness value is assigned to each individual
    b. The individuals with the best fitness value are selected and reproduced for the next generation
    c. A new population is created through genetic operators
    d. The result of genetic operators represents a possible solution of the generation

Typical termination criteria are the fitness function threshold and the number of generations.

The main genetic operators used for individuals evolution are reproduction, crossover, and mutation. The reproduction operator copies an individual (chosen for his fitness score) into the new population without any transformation. The crossover operator introduces variation in the population by creating offspring that includes some parts of their parents, chosen by a selection method. The mutation operator is a stochastic alteration of one or more genes. It introduces the exploration of new spaces of the fitness surface to avoid the premature convergence of the program. Each operator is realized with a certain probability. In particular, the probability of the mutation operator P_m that does not go over 0,1. Between the operators' probabilities, the following relation holds:

$$P_r + P_c + P_m = 1 \tag{18}$$

Where P_r, P_c, P_m represent the probability of reproduction, crossover, and mutation, respectively.

At each iteration, the GP creates a program, which can have a tree-based representation. The nodes represent the function set, which contains all the operator types, e.g. mathematical, arithmetic, Boolean, conditionals, and looping. Typical elements of the function set are summarized in Table 4 (H. Wang, Dong, and Chen 2020). The tree's leaves represent the terminal set, which includes the variables that have to be combined with the operators and sometimes constant values. Other important parameters for the GP are the generation gap, that is the percentage of the population that survives from a generation to another, and the parents selection methods, often chosen between tournament selection and roulette-wheel selection (Folino 2003).

Table 4 Function set of Genetic Programming

Kind of primitive	**Example(s)**
Arithmetic	Add, Multiplication, Division
Mathematical	Sin, cos, exp
Boolean	AND, OR, NOT
Conditional	IF-THEN-ELSE
Looping	FOR, REPEAT

Summing up, this section has provided basic concepts of IIoT infrastructure for data collection and analysis and a brief description of the methods usually adopted for signal processing and dimensionality reduction, which are depicted in Figure 17.

In the following sections, the case studies to which these techniques will be applied are presented. First, the available datasets are described in section 3.2. Then, data processing, including signal processing and dimensionality reduction, is conducted in section 0 on all or some of the described datasets.

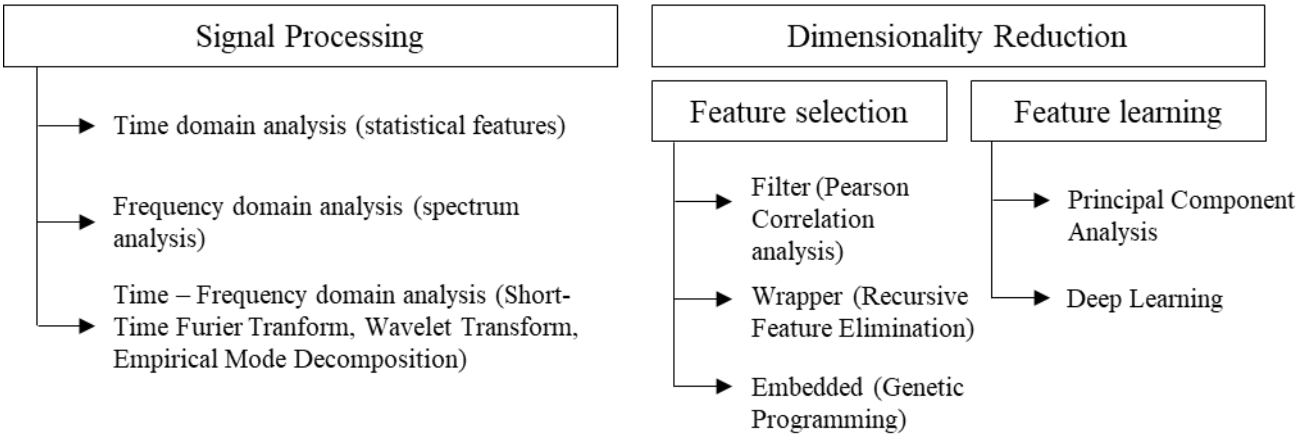

Signal Processing
Time domain analysis (statistical features)
Frequency domain analysis (spectrum analysis)
Time – Frequency domain analysis (Short-Time Furier Tranform, Wavelet Transform, Empirical Mode Decomposition)
Dimensionality Reduction
Feature selection
Filter (Pearson Correlation analysis)
Wrapper (Recursive Feature Elimination)
Embedded (Genetic Programming)
Feature learning
Principal Component Analysis
Deep Learning

Figure 17 Signal Processing and Dimensionality Reduction techniques

3.2 Data collection

This section describes the available datasets for fault diagnosis. In particular, it focuses on the format in which data are collected, i.e., number of different sources, number of files, weight (in terms of GigaBytes), and the type of available information. The first subsection considers a test rig built in the Department of Industrial Engineering of the University of Bologna. The system operates in a controlled environment, meaning that the operating and environmental conditions are known and kept under control in each experiment. The other subsections consider different case studies of industries that produce automatic machinery for the packaging sector. The case studies differentiate from each other for the collected data and the analyzed component or subsystem. Table 5 reports the available data (according to the classification presented at the beginning of this chapter) and the type of Condition Monitoring data for each case study.

Table 5 Case studies' available data

System/component	Available Data	Kind of CM signals
Test rig	Condition Monitoring, environmental, event, and domain knowledge data	Vibrations
Sealing group	Condition Monitoring and Environmental data	Displacement
Suction cups	Condition Monitoring data	Pressures
Case packer	Condition Monitoring data	Currents
Sealing group	Condition Monitoring and environmental data	Vibrations
Automatic Machinery	Condition Monitoring and domain knowledge data	Temperatures

3.2.1 Test rig (vibration signals)

The case study is conducted on vibration signals collected from an experimental platform built in the Department of Industrial Engineering of the University of Bologna (Figure 18).

The platform consists of an asynchronous motor, a gearbox made of two pulleys that exchange the rotation through a belt, two shafts that share the motion thanks to a couple of gears, an

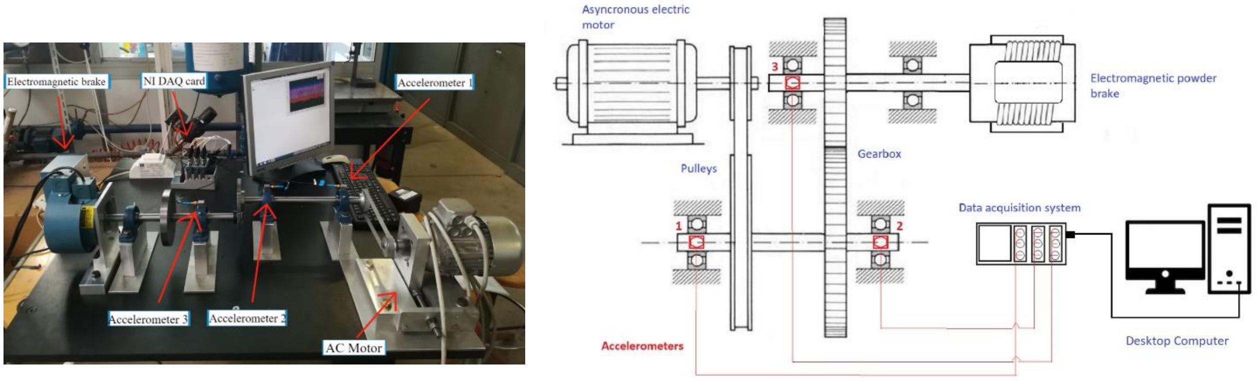
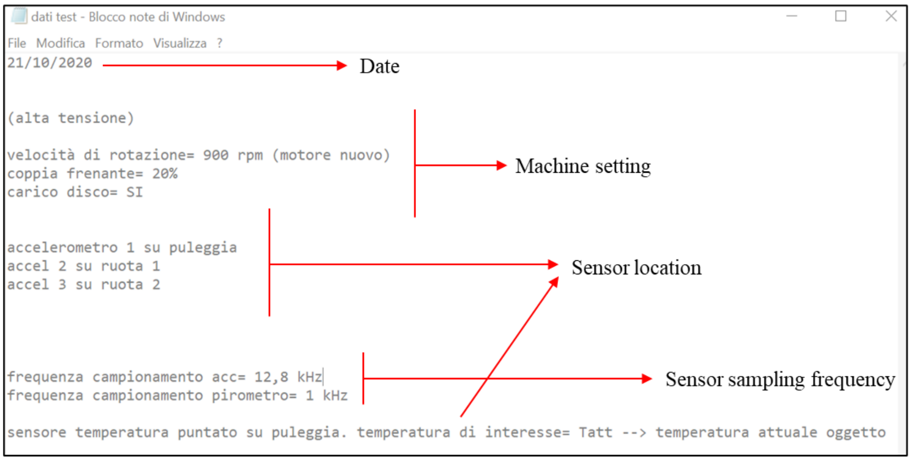


Figure 18 The test rig (left) and its mechanical scheme (right)

electromagnetic brake. The three-phase electric has eight poles with power equal to 0.13 kW and rotation speed equal to 660 rpm. The motion is transmitted to the first shaft through the belt running on the pulleys and put in tension thanks to a screw system positioned on the motor's support. A steel disk is attached to the second shaft to simulate a load on the centerline using a key and an o-ring. The braking system consists of an electromagnetic dust brake with adjustable braking torque in the range of 0-7.5 Nm and a 90 VDC command control fed by a transformer. The platform contains three accelerometers and a pyrometer. 3 Dytran 3093D3 triaxial accelerometers with IEPE technology are placed on the bearing's support, next to the second pulley and the two gearboxes. They have a sampling frequency of 12.8 kHz per axis and an acceleration range of 500 Gpeak. An OPTRIS CSmicro infrared sensor placed near the second pulley measures the pulley or the belt's temperature at a sampling frequency of 1 kHz. Regarding the data collection system, the acceleration sensors are connected to a computer through three-channel NI9230 I/O modules with a maximum sampling rate of 12.8 kS/s for each channel mounted on a four-slot NI 9274 chassis that collects all the data from the accelerometers before sending it to the computer through a USB connection. A data acquisition interface is placed in the pyrometer's cable, and temperature measurements are collected using the plug-n-play software CompactConnect supplied by the Optris company.

From the test rig, two kinds of data are collected, which are divided into three files

1. A file .txt, shown in Figure 19, in which the context data are recorded. In particular, the date of the specified setting implementation, the machine setting parameters (tension, rpm, braking force), the sensor location, and the sampling frequency of each sensor. This file is generated at the beginning of each test.

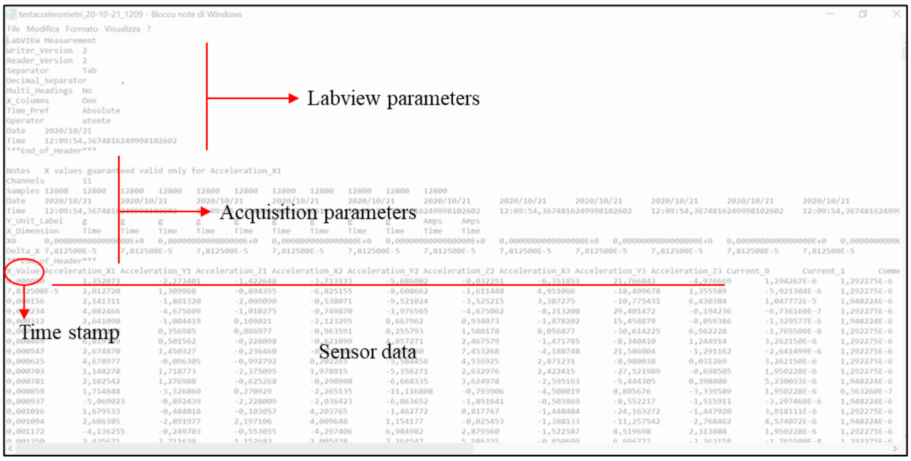

dati test - Blocco note di Windows
File Modifica Formato Visualizza ?
Date
Machine setting
Sensor location
Sensor sampling frequency

```
21/10/2020

(alta tensione)

velocità di rotazione= 900 rpm (motore nuovo)
coppia frenante= 20%
carico disco= SI

accelerometro 1 su puleggia
accel 2 su ruota 1
accel 3 su ruota 2

frequenza campionamento acc= 12,8 kHz
frequenza campionamento pirometro= 1 kHz

sensore temperatura puntato su puleggia. temperatura di interesse= Tatt --> temperatura attuale oggetto
```

Figure 19 Test rig: file .txt

2. A file .lvm, shown in Figure 20, in which the vibration signals collected by the three accelerometers are recorded. This file contains two headers in which Labview parameters and

acquisition parameters are recorded. In particular, acquisition parameters contain the date and time of recording, sampling frequency, unit of measure (g), the difference in time between two acquisitions. Finally, the accelerations for each ax of the accelerometers are recorded at each timestamp. For the accelerometers, these files are generated automatically every ten minutes during the machine functioning, and the resulting dimension of each file is equal to 1 GB. The length of the time interval between the two consecutive file creations has been set by the user, and corresponds to the best trade-off between the weight of each file and the number of resulting files from a complete test. Indeed, one single file including hours of data would be difficult to manage. On the other hand, too many files would require a larger pre-processing phase.


testaccelerometri_20-10-21_1209 - Blocco note di Windows
File Modifica Formato Visualizza ?
Labview parameters
Acquisition parameters
Time stamp
Sensor data

```
LabVIEW Measurement
Writer_Version  2
Reader_Version  2
Separator       Tab
Decimal_Separator       ,
Multi_Headings  No
X_Columns       One
Time_Pref       Absolute
Operator        utente
Date    2020/10/21
Time    12:09:54,3674816249998102602
***End_of_Header***

Notes   X values guaranteed valid only for Acceleration_X1
Channels        11
Samples 12800   12800   12800   12800   12800   12800   12800   12800   12800   12800   12800
Date    2020/10/21      2020/10/21      2020/10/21      2020/10/21      2020/10/21      2020/10/21      2020/10/21      2020/10/21      2020/10/21      2020/10/21      2020/10/21
Time    12:09:54,367481[illegible]      [illegible]     [illegible]6249998102602        12:09:54,3674816249998102602    12:09:54,3674816249998102602    12:09:54,367481624999
Y_Unit_Label    g       g       g       g       g       g       g       g       g       Amps    Amps
X_Dimension     Time    Time    Time    Time    Time    Time    Time    Time    Time    Time    Time
X0      0,0000000000000000E+0   0,0000000000000000E+0   0,0000000000000000E+0   0,0000000000000000E+0   0,0000000000000000E+0   0,0000000000000000E+0   0,0000000000000000E+0   0,0000000000
Delta_X 7,812500E-5     7,812500E-5     7,812500E-5     7,812500E-5     7,812500E-5     7,812500E-5     7,812500E-5     7,812500E-5     7,812500E-5     7,812500E-5     7,812500E-5
***End_of_Header***
X_Value Acceleration_X1 Acceleration_Y1 Acceleration_Z1 Acceleration_X2 Acceleration_Y2 Acceleration_Z2 Acceleration_X3 Acceleration_Y3 Acceleration_Z3 Current_0       Current_1       Comme
0,000000        1,352873        -2,273401       -1,422648       -3,713133       -5,686082       -0,032251       -6,351853       21,766843       -4,976669       1,294267E-6     1,292275E-6
7,812500E-5     2,012720        1,309968        -0,894355       -6,825155       -8,608662       -1,611448       4,951066        -18,409678      1,355589        -5,921304E-6    1,292275E-6
0,000156        2,141311        -1,801320       -2,009090       -0,538071       -9,521024       -3,525215       3,387275        -10,775431      6,430384        1,047772E-5     1,948224E-6
0,000234        4,082466        -4,675609       -1,010275       -0,749870       -1,976565       -4,675062       -8,213200       29,401472       -0,194236       -6,736160E-7    1,292275E-6
0,000312        2,641090        -1,004419       0,109021        -2,123295       0,667962        0,934073        -1,878202       15,458879       -0,059386       -1,329577E-6    1,948224E-6
0,0[illegible]  [illegible]     0,356985        0,086977        -0,963591       0,255793        1,580178        8,056877        -30,614225      6,962228        -1,765500E-8    1,292275E-6
0,000469        6,019[illegible]9       0,501562        -0,228098       -0,621099       4,857271        2,467579        -1,471785       -8,340410       1,244914        3,262150E-6     1,292275E-6
0,000547        2,674870        1,450327        -0,236460       -0,[illegible]  [illegible]80   7,453268        -4,188748       21,586004       -1,291162       -2,641499E-6    1,292275E-6
0,000625        4,678977        -0,096305       -0,992792       0,202265        -5,584456       4,536925        2,871231        -8,980038       0,031269        3,262150E-6     1,292275E-6
0,000703        1,148278        1,718773        -2,375095       1,978915        -5,356271       2,632976        2,422415        -27,521989      -0,698505       1,950228E-6     1,292275E-6
0,000781        2,102542        1,276988        -0,625268       -0,290908       -6,668335       3,624978        -2,595163       -5,444305       0,398800        5,230033E-6     1,948224E-6
0,000859        1,714848        -3,326860       0,270929        -2,265135       -11,116808      -0,793906       -4,500819       8,805676        -3,339589       1,950228E-6     6,363260E-7
0,000937        -5,069023       -0,892439       -2,228009       -2,036423       -6,863652       -1,891641       -0,503869       -8,552217       -1,515911       -3,297460E-6    1,948224E-6
0,001016        1,679533        -0,484018       -0,103057       4,203765        -1,462772       0,817767        -1,448484       -24,163272      -1,447920       3,918111E-6     1,292275E-6
0,001094        2,686385        -2,891977       2,197106        4,009648        1,154177        -0,025453       -1,388133       -11,257542      -2,768462       4,574072E-6     1,948224E-6
0,001172        -4,136255       -0,249703       -0,553055       -4,297406       6,984982        2,879560        -1,522587       8,519698        2,313888        1,950228E-6     1,292275E-6
0,001250        3,435671        2,711639        1,152683        2,005438        7,264547        5,586325        -0,850699       6,696777        -2,363158       -1,765500E-8    1,292275E-6
```

Figure 20 Test rig: file .lvm

3. A file .dat, which contains the data collected from the temperature sensor. The context data contained in the file is the same as the file .lvm. The difference lies in the CM data, which is the temperature signal, and in the fact that one only file covers all the test duration, because the lower sampling frequency. For this reason, the dimension of each file varies according to the duration of the test. For instance, for a test of 2 hours and 41 minutes, the dimension of the corresponding file is equal to 852 KB. Note that this file will not be used in this context since it does not provide helpful information on system settings and components' health conditions.

Two kinds experiments have been carried out on the test rig, according to two distinct goals:

1. The first experiment includes four datasets collected from the system under four distinct operating conditions, whose parameters are summarized in Table 6. All components of the test rig, i.e., the electric motor, the belt, the pulleys, and the gears, are in optimal conditions in three settings out of three. At the end of the last test, a sudden failure occurred to the electric motor. As it will be explained in the next sections, the goal of this experiment is the automatic

collection of contextual data during subsequent tests, that is to determine in which operating condition the system is working.

2. The second experiment has been conducted to collect run-to-failure data of the belt, which was put under an higher-than-nominal tension. These failure data have been collected under three different operating conditions, whose parameters are summarized in Table 7. In particular, two distinct trajectories have been obtained for setting 5 (F1 and F2) and setting 6 (F3 and F4); one trajectory has been obtained making vary the setting during the degradation of the belt (F5); finally, the last failure trajectory has been obtained under setting 7 (Table 8).

Figure 21 shows 1 second of data collected from each accelerometer.

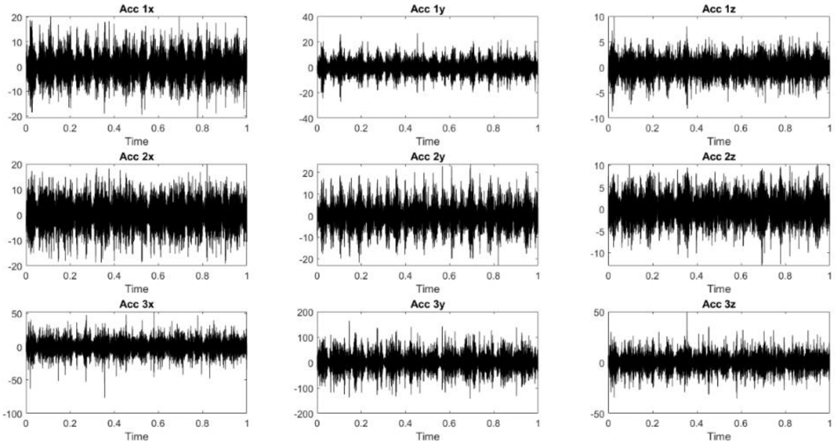


Figure 21 Test rig: raw signals collected from accelerometers (1 second)

Table 6 Operating conditions (first experiment on the test rig)

Operating Condition	Distance between pulleys (mm)	Breaking torque (Nm)	Rotation speed (rpm)	Duration (min)
Setting 1	27,33	0,1	660	82,8
Setting 2	27,33	0,5	660	184,2
Setting 3	27,54	0,1	660	77,4
Setting 4	27,54	0,328	660	25,8

Table 7 Operating conditions (second experiment on the test rig)

Operating condition	AC motor speed (rpm)	Braking force (Nm)
Setting 5	710	0.1
Setting 6	710	0.2
Setting 7	700	0,1

Table 8 Available Run-to-failure trajectories

Operating Condition	Run-to-Failure trajectories
Setting 5	F1 – F2
Setting 6	F3 – F4
Setting 5 – Setting 6	F5
Setting 7	F6

3.2.2 Sealing group (displacement signals)

The case of an automatic machinery of a manufacturer operating in the pharmaceutical sector is introduced in this subsection. The machine under analysis is a packaging line made of two parts connected by an automatic robot. The first section is a thermoforming machine that realizes the products in different materials; the second section is a cartooning machine, which packs the product into the carton. The machine can produce up to 320 products per minute and more than 260 cases per minute. Here, failures and malfunctions of the sealing group are considered, which is placed in the first section and is responsible for sealing the two parts of the product. The group includes two tables, in which the film flows horizontally and continuously, allowing the sealing at a low temperature. Each sealing cycle consists of five steps: (1) the top table goes up, (2) the tables close, (3) the cover is sealed to the base, (4) the tables open, and (5) the parts back to the initial position. The health condition of the sealing group is monitored through a displacement sensor that records the positions of the two tables. Figure 22 depicts the trend of the measured signal during a sealing cycle. Each cycle lasts 800 milliseconds, and the sampling frequency of the sensor is 1000 Hz.

A file .xls has been created for each test, containing the time stamp and the value recorded by the sensor. Table 9 summarizes the test performed on the system with the corresponding number of cycles included in each test. In total, four conditions are available, i.e., nominal, fault 1, fault 2, and fault 3. The condition "restored" is assumed to be equal to the nominal one. Hence, almost 50% of the available dataset corresponds to the nominal condition, while the remaining includes three different fault conditions.

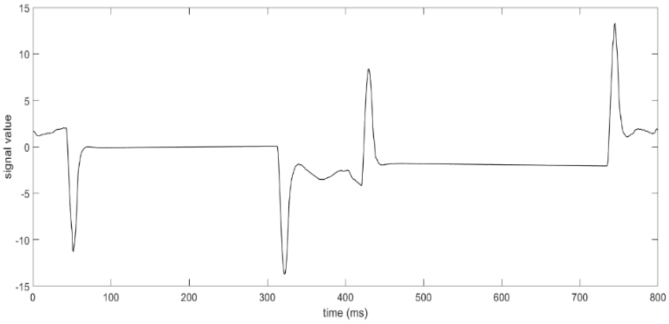


Figure 22 Sealing group: displacement signal of one cycle of the welding process in the nominal condition

Table 9 Sealing group (displacement signals): experiment type and duration

Dataset	Condition	Number of cycles	Dataset	Condition	Number of cycles
D1	Nominal 1	360	D6	Restored 2	110
D2	Fault 1	240	D7	Nominal 2	108
D3	Fault 2	200	D8	Nominal 3	48
D4	Restored 1	70	D9	Fault 1	93
D5	Fault 3	100	D10	Fault 3	90

3.2.3 Suction cups (pressure signals)

This case study considers a critical part of an automatic machine, whose malfunctioning strongly affects the quality of the products. The component is made of four suction cups, which rotate and push the product forward. The time length of a cycle is 0.134 s. The main problem that is related to this component is that, at each cycle, the pressure of the suction cups on the product decreases until one or more of the suction cups detach. It has been noted that if one only suction cup detaches, then the quality of the resulting product is still acceptable; however, if two suction cups detach, the component is not able to perform its function properly. Note that the detachment of the suction cup is almost instantaneous. The measure that best describes the component's functioning is the pressure, which suddenly drops when a suction cup detaches. Therefore, the pressure (bar) is collected at a sampling frequency of 10 kHz, under three conditions, named Nominal, Fault 1, and Fault 2, where Fault 1 represents the state in which only one suction cup is missing, and Fault 2 corresponds to the system with two suction cups missing. The pressure values in the different conditions are shown in Figure 23. Note that the test during which data was collected was not performed for maintenance purposes. Thus, the choice of the sampling frequency was derived from other considerations.

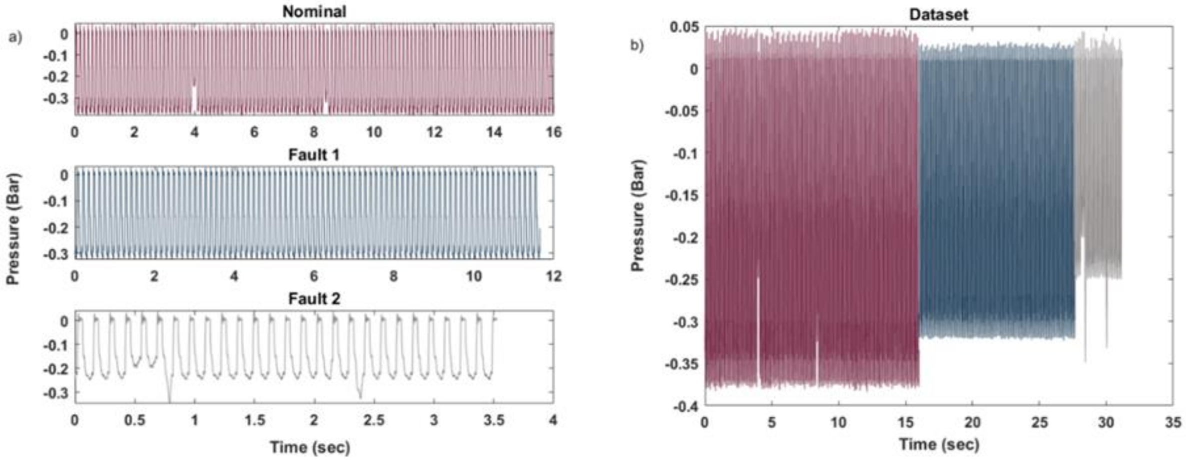


Figure 23 Suction cups: (a) Signals related to three different conditions. (b) Complete dataset containing all conditions.

3.2.4 Case packer group (current signals)

Like in the other cases, a subgroup of an automatic machinery is considered in this case study. In particular, the case packer group is responsible for putting the blisters into the case and sticking its edges. The incorrect execution of the sticking phase may generate unacceptable final products. For this reason, it is necessary to recognize the malfunctioning as soon as possible. To this aim, the current (mA) absorbed by the main actuator of the system is monitored. Indeed, different current trends corresponds to the different non-optimal working conditions. To demonstrate this assumption, the current signal during each machine cycle been collected with a sampling frequency of 500 Hz in different conditions. In total, one normal condition and four fault conditions were simulated. For each condition, 100 cycles are available, except for one condition, for which the test lasted only 47 cycles. Each cycle is made of 75 data samples and lasts 0,15 seconds. Therefore, 15 seconds of monitoring have been conducted for each condition, except for fault 4, for which the monitoring is 7,05 seconds. In Figure 24, all cycles of each condition are shown.

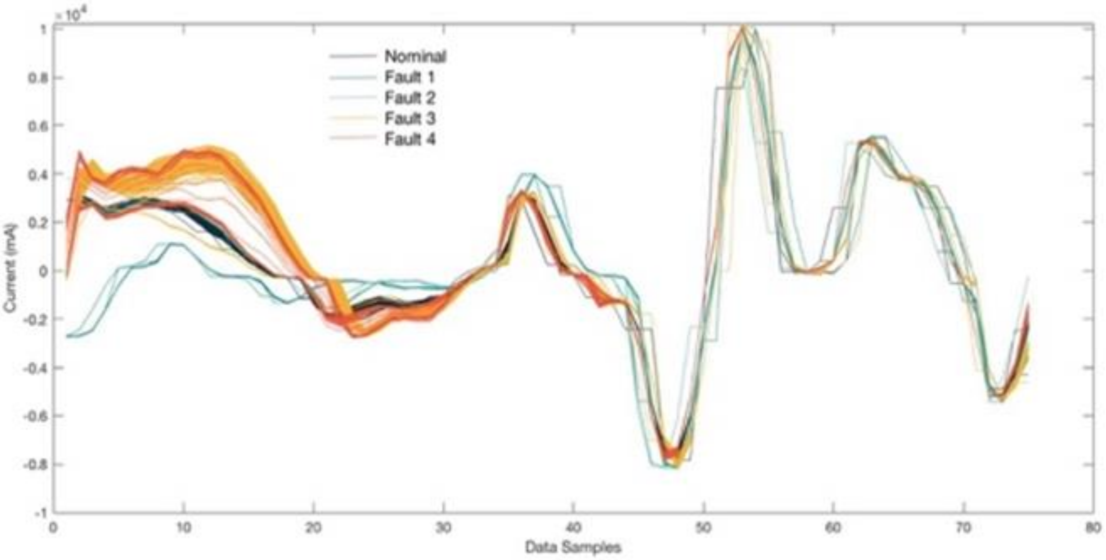


Figure 24 Case packer: raw signals corresponding to the five available conditions

3.2.5 Sealing group (vibration signals)

The fourth case study considers vibration signals collected from the sealing group of an automatic machinery for the packaging of tea sachets. The machinery is able to perform of 400 cpm (cycles per minutes), and realizes eight sachets at each cycle, resulting in a nominal production rate equal to 3200 sachets per minute. The sealing group is made of eight pockets paired up in twos. During each cycle, which lasts 0.15 seconds, two sachets are sealed and the component moves forward by two pockets. The monitoring objective is ensuring that the sachets sealed correctly. Vibration signals were collected through a triaxial accelerometer installed on the sealing group. Acceleration measurements were taken in three directions: radial, axial, and tangential to the circumference generated by the rotation of the component around its drum axis. The three signals collected by an accelerometer were connected to the three channels of a digitizer. For each channel, samples were taken every 3,9063e-5 seconds. The resulting sampling frequency is 25,600 Hz. This component can be used in four different configurations to make four distinct final products. In addition, two load conditions can be considered for each product: 50%, which is the most common, and 100%. In order to understand the component behavior in each possible configuration, eight different situations were simulated, during which vibration signals were collected. For the first condition, three tests were conducted, each including eight files, while for the second condition, only one test was conducted. Tests were carried out on the component in both nominal and damaged conditions. 412,000 samples are included in each file of each test, lasting 16.09 seconds, except for the last file of each test, which contains 140,750 samples for a time of 5,58 seconds. Figure 26 shows an example of a signal generated by one of the tests. It is worthy to note that some signals present anomalous peaks due to electric problems. By visualizing signals obtained from the three channels, it was evident that the z-axis was the most problematic. For this reason, analysis has been conducted by considering first the total value of acceleration Acc_{tot},

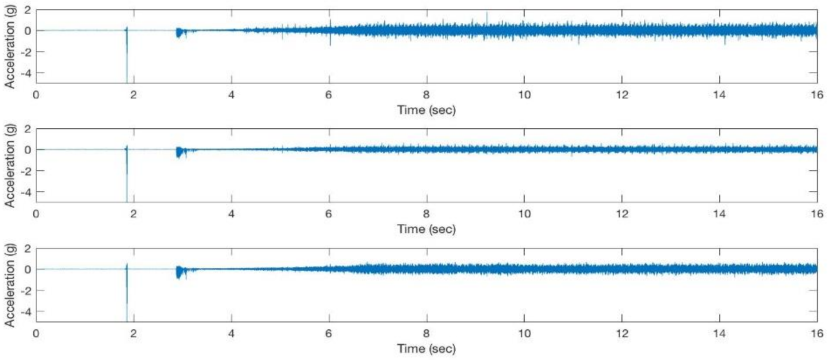


Figure 26 Vibration signals collected from the accelerometer

and then by considering the acceleration of only on x-y axis, Acc_{x-y}.

$$Acc_{tot} = \sqrt{a_{Ch1}^2 + a_{Ch2}^2 + a_{Ch3}^2} \tag{19}$$

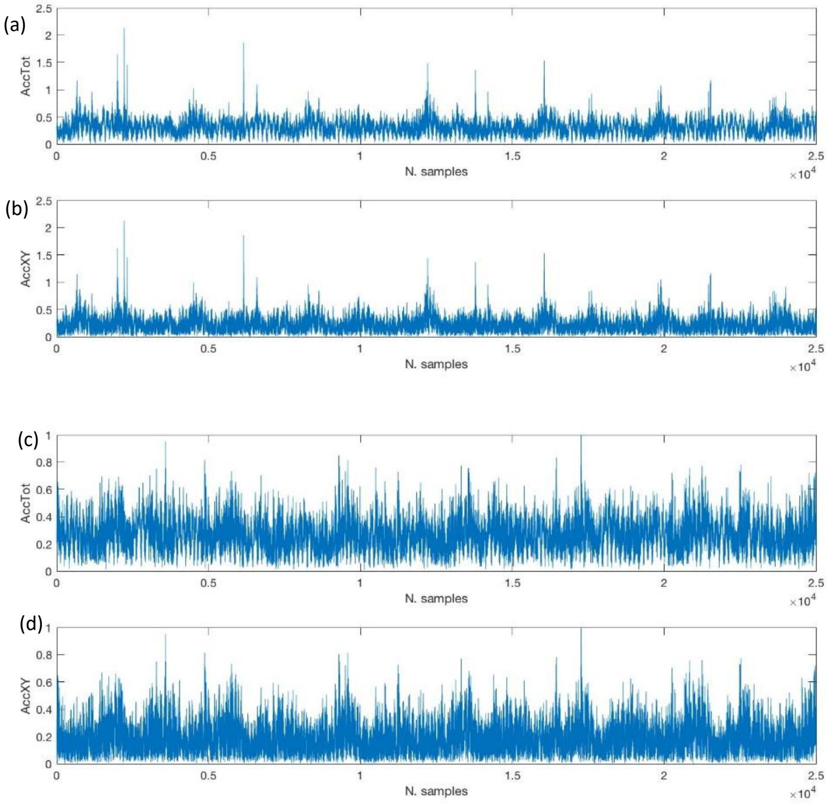


Figure 25 Sampled Signal. (a) New component- Case 1 (b) New component- Case 2 (c) Damaged component - Case 1 (d) Damaged component - Case2

$$Acc_{x-y} = \sqrt{a_{Ch1}^2 + a_{Ch2}^2} \tag{20}$$

Acc_{tot} and Acc_{x-y} will be referred to as Case 1 and Case 2, respectively. Figure 25 shows signals in each window of 1 second for both cases and conditions.

3.2.6 Automatic Machinery (temperature signals)

The last case study considers several subgroups of an automatic machinery. In particular, the data were collected from different users of different machinery belonging to the same product family. All the datasets were gathered in a unique folder organized by the machine producer, as depicted in Table 10. First, the dataset is divided for client and for machinery type. Then, for each client C and machinery type M, data are divided into five groups corresponding to five distinct subsystems (A, B, C, D, E). The data are recorded into three log files, each containing signals collected from different sources, at different sampling frequency, and form sensors placed on one or more subsystems. All files are provided in the .xls format. The log files belong to two categories: files extracted from the automation PC of the machinery and files extracted from the HMI interface, which were extracted continuously or on-demand. For this reason, datasets collected from distinct machinery cover different periods. In total, eight distinct data sources are available. Two of them are extracted from the HMI interface and can be considered low-frequency data, while the others are high-frequency data extracted from the PLC.

The first log file extracted from the HMI interface contains the data recorded one time per day after 30 minutes of the machinery functioning (low-frequency data). Variables in this file include:

1. The timestamp
2. Parameters collected from all, or some, sensors placed on the subsystems of the machinery (it depends on the clients)
3. Statistics (minimum, maximum, mean, and standard deviation) of the above-mentioned parameters, which were computed on a batch of 30 minutes
4. Setting values of the machinery.

The second log file extracted from the HMI interface contains the event “setting change” data (low-frequency data). Note that while the setting values of machinery recorded in the first log can be considered contextual data, the data recorded in the second log are event data. The difference lies in the “moment” in which the information is recorded. While in the first case the information is recorded for each observation (at each time stamp), in the second case, the information is only recorded when the setting changes. Thus, it can be considered an event.

The third and fourth log files contain the Condition Monitoring Data collected from sensors located in different positions of subsystem A, i.e., the extruder. In particular, the third log file contains temperature and pressure signals collected at a sampling frequency of 1 Hz. The fourth log file contains the speed and, in some cases, the production rate, collected at a sampling frequency of 10 Hz. The other logs contain the signals collected from the other subsystems at a sampling frequency corresponding to the production rate. These files will not be described in detail because no analysis has been conducted. The reason is that the datasets are extracted in different moments. Even for the same machinery, the different log files contain different timestamps. Hence, in some cases, no valuable data are available.

Table 10 Automatic machinery: available datasets

Machinery	Client	Machinery Group	Log File							
			1	2	3	4	5	6	7	8
M1	C1	A	X	X	X					
		B			X					
		C	X	X	X	X			X	
		E			X					
	C2	A	X	X	X	X				
		B			X					
M2	C3 (2)	A					X			
		B						X		
		C					X			
M3	C3	A					X			
		B						X		
		C					X			
	C4	A	X							
		B		X						
		C	X							
	C6 (2)	A					X			
M4	C5	A								
		B					X	X		
		C					X			
		D					X			X
		E						X		
M5	C3	A			X	X				
		B			X	X				
		C			X	X				
		E			X	X				

3.3 Data processing

This section describes the results obtained from applying several data processing methods to the datasets described in the previous section. Data processing is conducted to obtain two kinds of feature sets: the component-level feature set and the system-level feature set. In the first case, the feature set represents a nominal or fault condition in which a specific component operates. In the second case, the feature set represents the operating condition (i.e., the machinery setting) in which the system operates in a given instant. The methods used to obtain the feature sets can be divided into signal processing techniques and dimensionality reduction methods. The first set of methods consider time-domain and time-frequency-domain analysis. Then, in some cases, dimensionality reduction methods, i.e., filter and wrapper feature selection methods or feature construction methods, are also applied to select, or build, the most relevant feature sets. The goals and methods for each case study are summarized in Table 11.

Table 11 Case studies goals and methods for data processing

Case study	Goal	Method
Test rig (Case 1)	System-level feature extraction	1. Statistical features in the time-domain 2. A) Feature selection (Pearson Correlation analysis and manual feature selection) B) Feature Construction (Genetic Programming)
	Component-level feature extraction	1. Statistical features in the time-domain 2. A) Feature selection (Pearson Correlation analysis and manual feature selection)
Sealing group: Displacement (Case 2)	Component-level feature extraction	1. Statistical features in the time-domain 2. Pearson Correlation analysis 3. Recursive Feature Elimination
Suction Cups (Case 3)	Component-level feature extraction	1. Statistical features in the time-domain 2. A) Feature selection (Pearson Correlation analysis and Recursive Feature Elimination) B) Feature Construction (Principal Component Analysis)
Case packer group (Case 4)	Component-level feature extraction	1. Statistical features in the time-domain 2. A) Feature selection (Pearson Correlation analysis) B) Feature Construction (Principal Component Analysis)
Sealing group: Vibrations (Case 5)	Component-level feature extraction	1. Data pre-processing 2. A) Statistical features in the time-domain 3. B) Statistical features in the time-frequency domain (Morlet wavelet transform)
Automatic Machinery (Case 6)	System-level feature extraction	1. Feature construction (Principal Component Analysis)

The results are presented in the following subsections according to the specific task, i.e., signal processing, feature selection, and feature construction. In particular, signal processing techniques in the time-domain are applied to all datasets, except the extruder; the time-frequency analysis is only conducted for the sealing group (vibrations). Feature selection using filter and wrapper methods is applied to all datasets, except the extruder and the sealing group (vibration signals). Feature construction is applied to the second dataset of the test rig to extract both component-level and system-level features, the suction cups and the case packer to extract component-level features, and the extruder to extract system-level features.

3.3.1 Signal processing

The preliminary steps before signal processing are the following:

1. To determine the length of the window from which the statistics have to be extracted.
2. For the analysis in the time-frequency domain, to determine the wavelet for transforming the raw signal in the time-frequency domain.
3. To identify statistics to compute. In all cases, all or some of the typical statistical time-domain features described in Table 3 (section 3.1.2) are used. The statistical time-frequency features extracted from coefficients produced by the wavelet transform are described in Table 12, where, $C_i(t)$ are the coefficients generated by the Morlet transform, and C_m is the average value of $C_i(t)$.

The output of signal processing is represented by a matrix $[mxn]$, where m is the number of observations, which corresponds to the number of signal segments in which the raw signal has been divided, and n is the number of extracted features.

Note that signal processing does not consider the aim, i.e., component-level or system-level feature extraction. This aspect is considered in the feature selection or construction step. In addition, the goodness of the extracted features is generally assessed through classification models. For this reason, results from this analysis will be presented in the next chapter, after providing details on classification and clustering models for fault detection and diagnosis.

Table 12 Statistical features extracted in the time-frequency domain

Feature	Formula	Feature	Formula
f_1	$\min(C_i(t))$	f_3	$\sqrt{E[(C_i(t) - C_m)^2]}$
f_2	$\max(C_i(t))$	f_4	$f_2 - f_1$

Case 1. Test-rig: statistical features in the time domain. First, the dataset related to the first experiment conducted on the test rig is considered (Table 6). Each of the nine vibration signals (three for of the three accelerometers) is divided into segments of 1 second, including 12.800 samples, and

the most typical features in the time domain, i.e., peak, peak-to-peak, mean, RMS, Crest Factor, Skewness, Kurtosis, Shape Factor, and Impulse Factor, are extracted from each segment. In total, 81 time-features are extracted. As an example, the nine features extracted from the first signal (axes x of the first accelerometer) are shown in Figure 27. From the figure, it is possible to see that the trends of Crest Factor, Skewness, the Kurtosis, do not show any difference between the four settings, while the trend of the other features is more or less different among different settings, especially considering setting 4. Note that, in all the cases, several abnormal peaks are present, which may compromise the classification models and the anomaly detection models for setting recognition and fault detection.

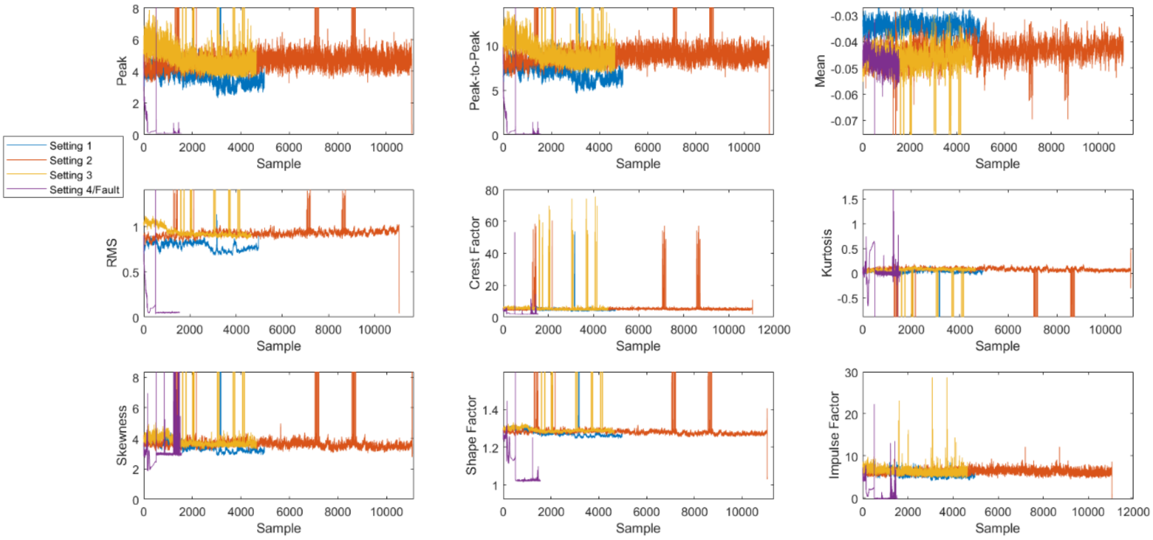


Figure 27 Test rig: Statistical time-domain features extracted from x-axes of accelerometer

<u>Case 2. Sealing group (displacement signals): statistical features in the time domain</u>. In Figure 28, the trend of the extracted features from displacement signals collected from the sealing group (Table 9) in each condition is shown. In this case, the length of the window is set to 800 ms, equal to the cycle duration. Ten typical time-domain features are extracted for each cycle, i.e., peak, peak-to-peak,

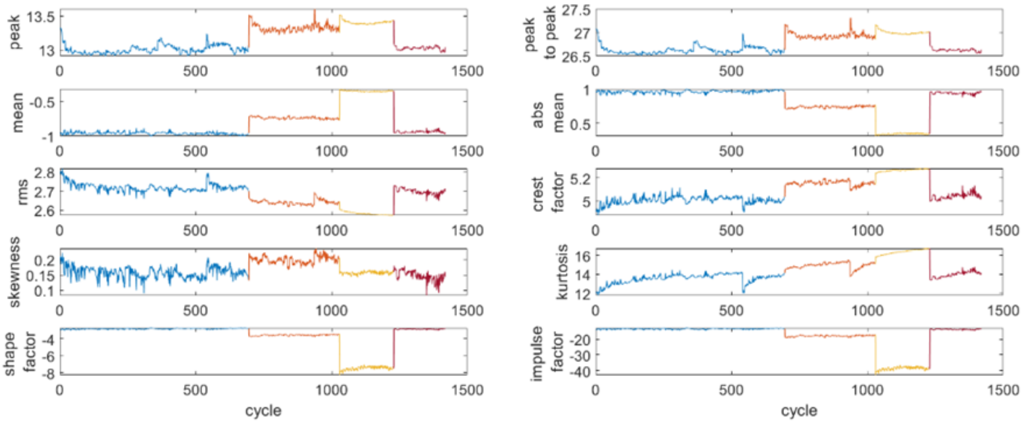


Figure 28 Sealing group (displacement signals): The trend of the extracted features during the nominal condition (blue), fault 1 (orange), fault 2 (yellow), fault 3 (violet).

mean, absolute mean, RMS, crest factor, skewness, kurtosis, shape factor, and impulse factor. During the nominal condition, an initial increasing trend can be observed for some features, e.g., kurtosis, which corresponds to the warm-up phase of the machine. Finally, fault 3 is very similar to the nominal condition.

Case 3. Suction cups: statistical features in the time domain. In Figure 29, the trends of the time-features extracted from the pressure signals collected from the suction cups in three conditions are shown. In this case, the time window from which features are extracted is set equal to 0.134 seconds (1340 observations), which corresponds to the duration of a cycle. It can be seen that the peak, the peak-to-peak, the mean, and the RMS assume different values in the three conditions, while the other features have similar trends.

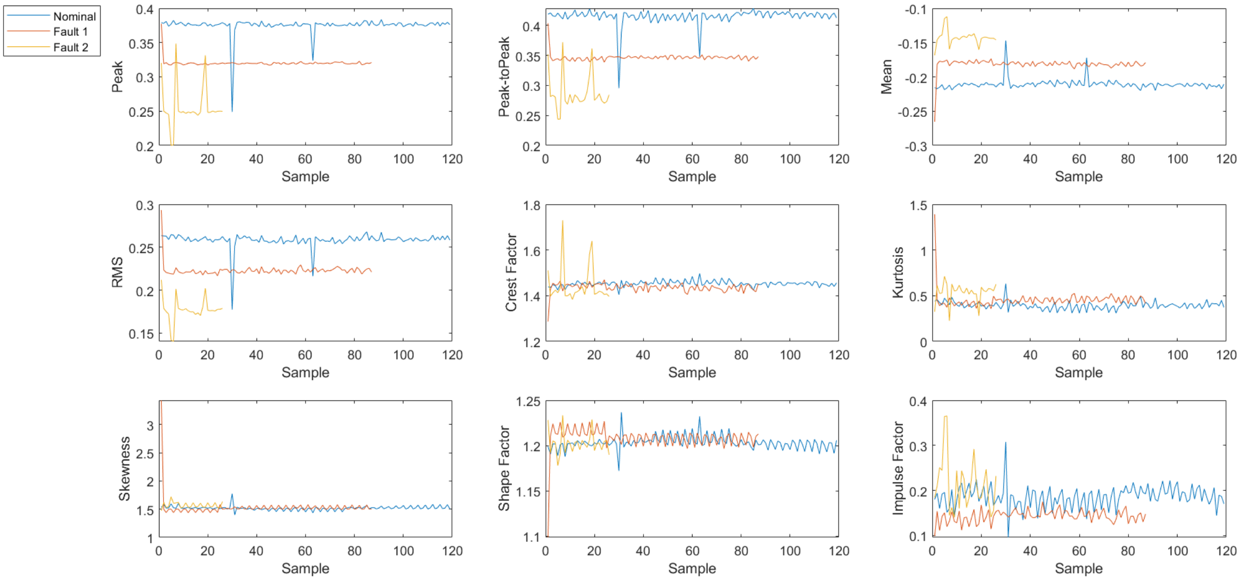


Figure 29 Suction cups: statistical time-domain features

Case 4. Case packer group: statistical features in the time domain. Figure 31 shows the extracted features at each cycle (75 samples), for each condition. It can be seen that, for each condition, all features assume similar trends during the cycles, except for the last condition. Indeed, all features extracted from data corresponding to Fault 4, except for the peak and the peak-to-peak, show a decreasing trend over the cycles.

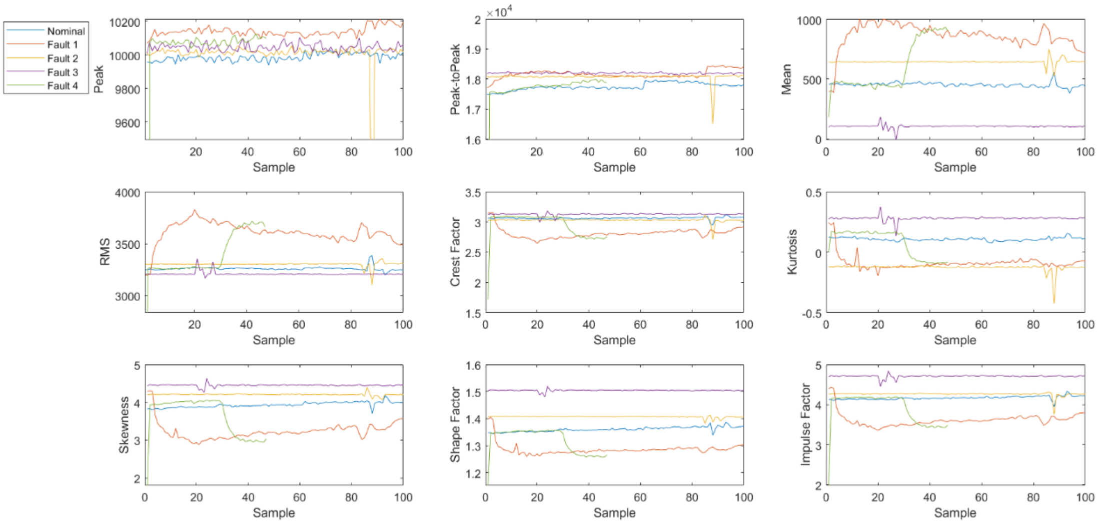


Figure 31 Case packer group: statistical time-domain features

Case 5. Sealing group (vibration signals): statistical features in the time domain and in the time-frequency domain. For this analysis, two different configurations (products) have been chosen, both with the 50% as load condition. Each test has been divided into 16 windows, containing 25,600 and lasting 1 second each, except for the last file of each test, which has been divided into five windows. Hence, 351 windows (signal segments) were obtained for each product for both nominal and failed

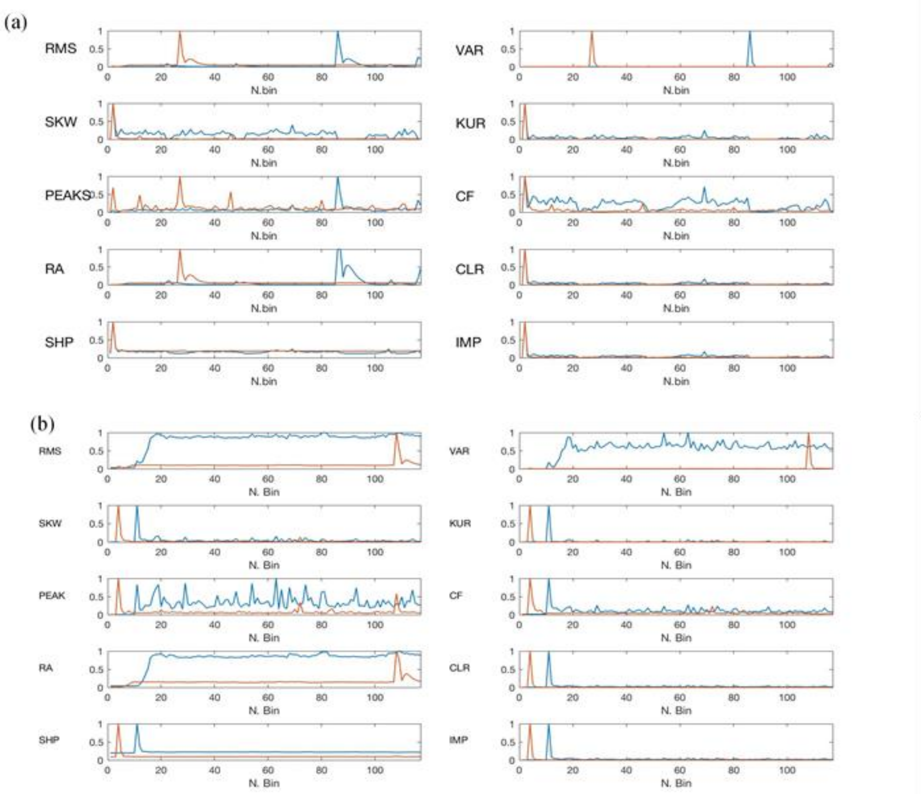


Figure 30 Sealing group (vibration signals): time-domain features for (a) product 1, and (b) product 2 in nominal condition (blue) and damaged condition (orange)

components. From each signal, the following statistical time-domain features have been extracted: RMS, variance, skewness, kurtosis, peak, crest factor, root amplitude, clearance, shape factor, and impulse factor, which are shown in Figure 30. It can be seen that the trend of the time features, in both nominal and damaged condition, are different for product 1 and product 2. For Time-Frequency-domain analysis, each signal segment has been transformed by using the Morlet wavelet transform. Figure 32 shows the signal in the time domain (red signal), the coefficients $C_i(t)$ and the reconstruction of the signal (green signal) for the component in nominal condition (on the left) and the damaged component (on the right). It is evident that damaged components present higher frequencies than the component in the nominal operating condition.

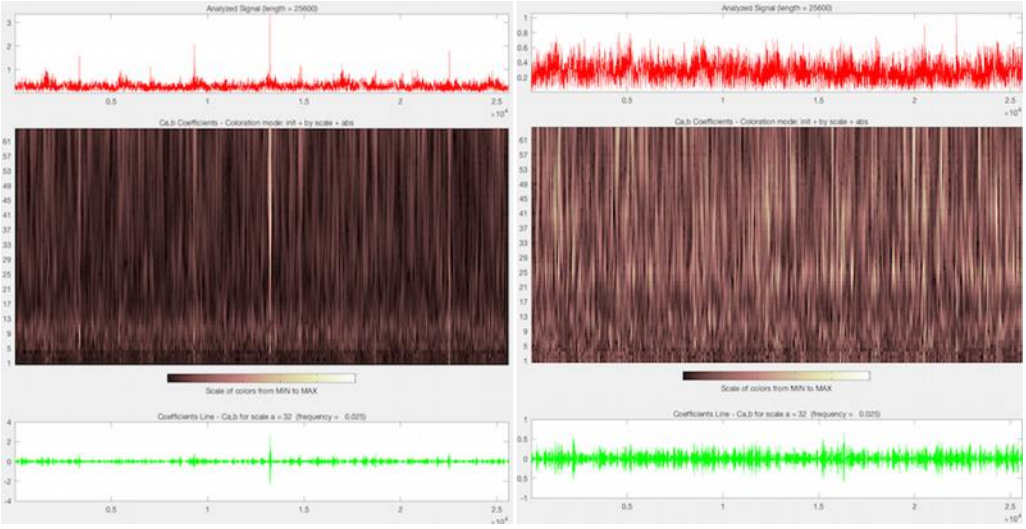


Figure 32 Sealing group (vibration signals): Morlet Wavelet Transform of nominal signals (left) and demaged signals (right)

3.3.2 *Feature selection*

Two methods are considered for feature selection: the Pearson correlation analysis and the Recursive Features Elimination. The Pearson Correlation analysis has been conducted in Matlab, through the function `corrcoef`, which gives a squared matrix containing the correlation coefficients among all the features. Then, features with a correlation coefficient greater than 0.9 or lower than -0.9 have been eliminated. Note that the Pearson Correlation analysis is a filter method, that does not require any classification models to evaluate the selected feature sets. Hence, the selected set of features is independent from the specific aim (setting recognition or fault diagnosis). On the contrary, the wrapper feature selection approach named Recursive Feature Elimination (RFE) is a wrapper method, and has been used to select the optimal feature set in terms of classification accuracy. The RFE algorithm has been applied using the function `rfe_cv` of the Python library Scikit Learn. In some cases, a manual feature selection is also performed, meaning that the relevant features have been selected based on subjective judgments derived from the graphic visualization.

Case 1. Test rig: Pearson Correlation analysis and manual feature selection. For the first experiment on the test rig, the most relevant features were selected using two methods, i.e., the Pearson Correlation Analysis and the manual selection. In the first case, 32 features have been selected, which are summarized in Table 13.

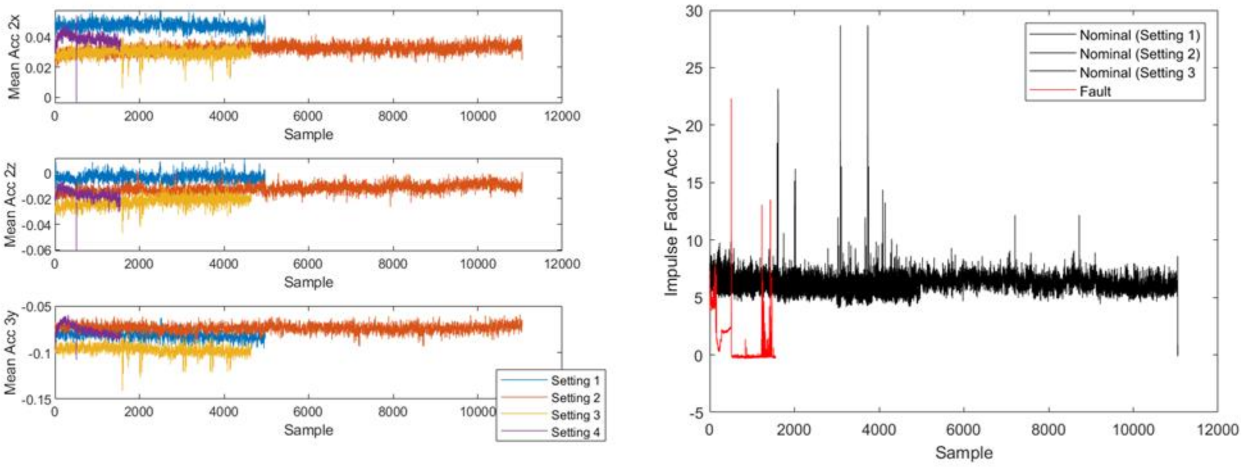
Table 13 Test Rig. Selected Feature through Pearson Correlation Analysis

Accelerometer	Ax	Feature
1	x	Peak, Mean, RMS, Shape Factor, Impulse Factor
	y	Mean, Impulse Factor
	z	Mean, RMS, Shape Factor, Impulse Factor
2	x	Peak, Mean, Crest Factor, Kurtosis, Shape Factor, Impulse Factor
	y	Peak, Mean, Impulse Factor
	z	Peak, Mean, Kurtosis, Impulse Factor
3	x	Mean
	y	Mean, Crest Factor, Kurtosis, Skewness
	z	Mean, Kurtosis, Impulse Factor

In the second case, all features have been plotted and relevant features have manually been selected according to two criteria, reflecting the purpose of the analysis.

1. For the operating condition recognition (OCR) goal, relevant features should be as much as constant (similar) when refer to the same condition. On the contrary, they should assume different values depending on the condition they refer. Therefore, the mean value of signals 4, 6, and 8 (corresponding to the x and z axes of the second accelerometer and the y axes of the third accelerometer, respectively) were selected. Indeed, they best distinguish the four settings, which are represented with different colors in Figure 33a.
2. For the fault detection goal, relevant features should be as much as constant (similar) in all the settings. On the contrary, they should assume anomalous values when the fault occurs. Therefore, the impulse factor of signal 2 (axes y of the first accelerometer) was selected, as it resulted to be independent of the implemented setting and shows anomalous peaks, which are depicted in red in Figure 33b, when the electric motor fails.

Thanks to the manual feature selection, only 4 of the 81 previously extracted features have been selected, which are divided into two sets: the system-level feature set, including the three features selected for the OCR task; the component-level feature set, including one only feature, selected for the fault detection task.



(a) (b)

Figure 33 Test rig: features selected for condition recognition; b) Feature selected for fault detection

<u>Case 2. Sealing group (displacement signals): Pearson Correlation analysis and Recursive Feature Elimination</u>. Although only ten features have been extracted from displacement signals (Figure 28), they show similar trends. Hence, some redundancy is present. To select the relevant and non-redundant features, first, the Pearson correlation analysis has been conducted. Then, the Recursive Features Elimination is also applied to take into account the ability of the features to classify the different classes. First, datasets corresponding to all available machinery conditions are considered in the feature selection process. In a second scenario, fault 2 has been removed from the training dataset to evaluate how the selected feature set changes when not all conditions are known in advance. In other words, the generalization ability of the feature sets is evaluated, where generalization, in this case, means the relevance in distinguishing not only known conditions, but also operating conditions that can be implemented in the future (this concept will be better explained in the next chapter). The ten statistical features extracted in the time domain are used to build the Pearson correlation matrices of both scenarios, shown in Figure 34. It can be seen that, in both cases, some of the extracted features have a high correlation, like the mean and the mean of the absolute values, which have a correlation coefficient equal to -0.98 (Figure 34a). In this case, the mean absolute on the column is eliminated, while the mean on the row is kept. This process is repeated until the whole matrix is analyzed. In the end, for scenario 1, only four features, i.e., peak, peak to peak, the mean, and the skewness, are selected, whose correlation coefficients are shown in Figure 34b. For scenario 2, the peak, peak to peak, and the skewness are selected (Figure 34d). To further evaluate the goodness of the selected features, the RFE algorithm is applied, which assigns a score equal to 1 to both feature subsets selected through the Pearson correlation analysis. This score confirms that the selected features have a low correlation and positively contribute to the classification accuracy.

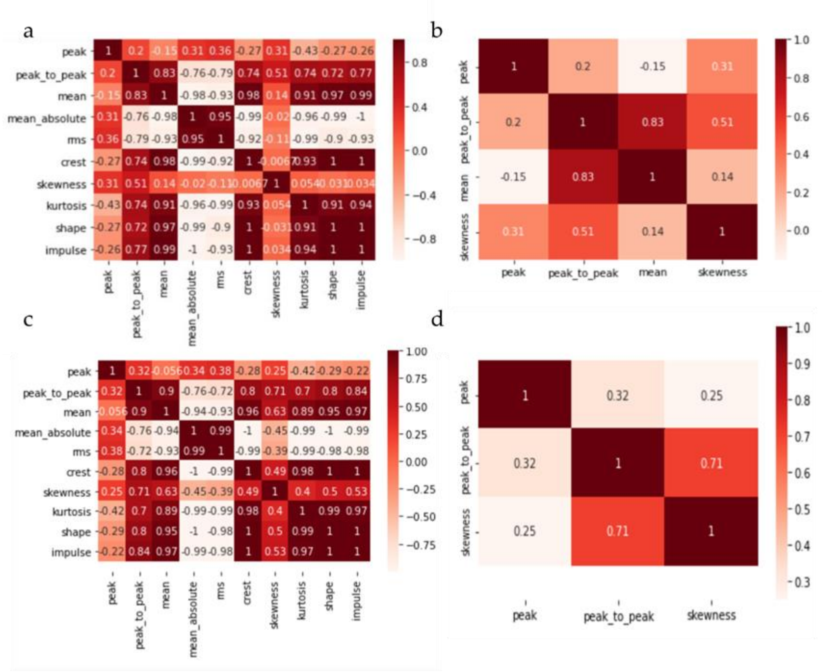


Figure 34 Pearson correlation matrix of the whole set of features (a) and Pearson correlation matrix of the selected features (b) – scenario 1. Pearson correlation matrix of the whole set of features (c) and Pearson correlation matrix of the selected features (d)

Case 3. Suction Cups: Pearson Correlation analysis and Recursive Feature Elimination. In this case, the ten statistical features extracted in the time domain are used as input of the Pearson Correlation analysis and the Recursive Feature Elimination. In both cases, the peak, the Crest Factor, the Kurtosis, the Skewness, the Shape Factor, and the Impulse Factor are selected since they are the less correlated among each other and provide the best classification accuracy results.

Case 4. Packer group: Pearson Correlation analysis. In this case, only 4 features, i.e., the peak, the mean, the RMS, and the Crest Factor have been selected through the Pearson Correlation analysis.

3.3.3 Feature construction

Feature construction is intended as a combination or transformation of extracted features that implies the loss of their original physical meaning. The Genetic Programming, and the Principal Component Analysis are considered feature construction methods. When applied to the case studies, some common decisions have been taken. In case of the PCA, the PCs retaining at least the 99% of the variance are selected. In the case of GP, two fitness functions are built, according to the kind of features (Health Indicator or system-level features) that need to be extracted. Both algorithms are implemented using MATLAB. In particular, the PCA has been applied using the function `pca`, while the GP has been applied using the Genetic Programming Toolbox developed by Janos Abonyi (2021).

Case 1. Test rig: Genetic Programming. In the GP design, both classification and clustering measures have been considered in the objective function to evaluate both the supervised and unsupervised learning ability of the GP. For the classification GP-based feature construction, the accuracy of K-NN is considered in the fitness function, as in (B. Peng et al. 2020). For the clustering GP-based feature construction, the mean of the silhouette coefficient of all observations, given by Eq. 3, is used as the fitness function (Schofield and Lensen 2020). The silhouette coefficient of a given point measures its similarity to points belonging to the same cluster, compared with points belonging to the other clusters. Here, it is computed on clusters generated by the k-Means algorithm.

$$Fit = \frac{1}{N}\sum_{i=1}^{N} s_i \tag{21}$$

where

- $s_i = \frac{a_i - b_i}{\max\{a_i, b_i\}}$ is the silhouette value of the point i,
- $a_i = \frac{1}{|C_i|-1}\sum_{j\in C_i, i\neq j} d_{i,j}$ is the average distance from the *i*th point to the other points in the same cluster as i,
- $b_i = \min\limits_{k\neq i} \frac{1}{|C_k|}$ is the minimum average distance from the *i*th point to points belonging to other clusters
- N is the total number of observations.

The silhouette value ranges from -1 to 1. A high silhouette value indicates that *i* is well matched to its cluster, and poorly matched to other clusters. If most points have a high silhouette value, then the clustering solution is appropriate.

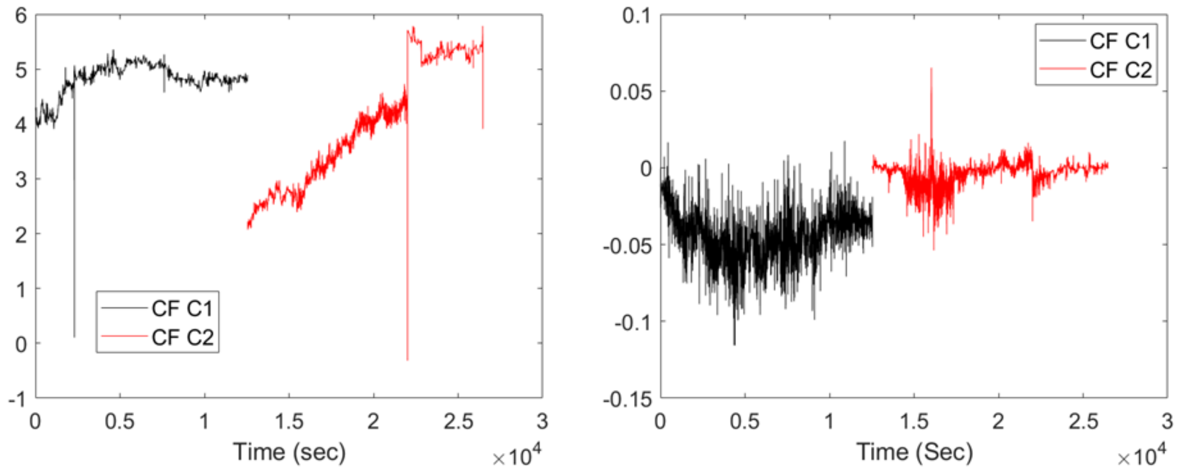
The so-built GP algorithms have been applied to the second experiment conducted on the test rig (see Table 7), in order to find the system-level feature set able to distinguish the two operating conditions despite the belts' degradation. In particular, the 81 statistical features (9 for each of the 9 signals) extracted in the time domain from run-to-failure trajectories F1, F2, F3, and F4 have been used for training the GP. After running the GP 20 times, the best fitness value of classification-based GP, equal to 0.9999, is provided by the feature given in Eq. 22 and shown in Figure 35a, where the black line corresponds to the constructed feature for the first operating condition, and the red line corresponds to the constructed feature for the second operating condition.

$$f_{42} + f_4 \tag{22}$$

where, f_{42} corresponds to the skewness extracted from the fifth signal (the y-axis of the second accelerometer) and f_4 corresponds to the RMS of the x-axis of the first accelerometer. For clustering-based GP, the best fitness value, equal to 0.8408, is provided by the feature given in Eq. 23 and shown in Figure 35b.

$$f_{67} * f_{42} * f_{48} \tag{23}$$

where f_{67} is the RMS of the x-axes of the third accelerometer, f_{42} is the skewness extracted from the y-axis of the second accelerometer, and f_{48} is the mean extracted from the z-ax of the second accelerometer.



(a) (b)

Figure 35 Constructed feature through (a) classification-based GP, and (b) clustering-based GP

Case 2. Suction cups: Principal Component Analysis. In this case, the PCA is applied to the set of time-domain features extracted in the previous section (see Figure 29), for each condition. For the nominal condition and the first fault condition, the first two PCs are selected, which explain the 99.9% and 99.6% of the variance, respectively. For the last condition, three PCs are selected, which explain

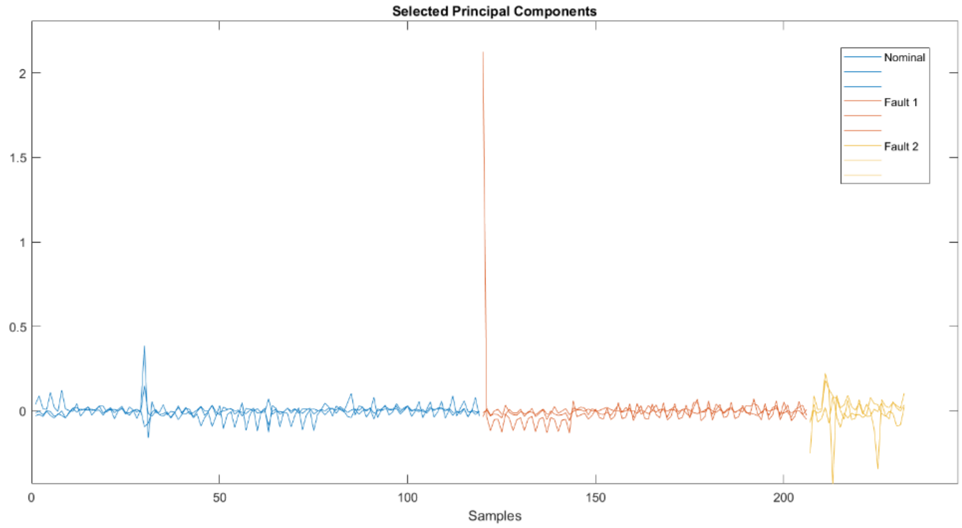


Figure 36 Suction Cups. Extracted Principal Components

the 97.38% of the variance. Since Machine Learning models require the same number of variables for each condition, three PCs are selected for each condition. As it can be seen in Figure 36, the PCs assume similar values in all conditions.

Case3. Case Packer group. For each condition, the statistical time-domain features are used as input for the PCA and results are depicted in Figure 37. For the nominal condition and the first fault condition, the first three PCs explain the 99.5% and the 99.6% of the variance, respectively. For the second and fourth fault conditions only the first PC is selected, which explains the 99.04% and the 99.16% of the variance, respectively. Finally, for the third fault condition, for PCs are selected, which explain the 99.99% of the variance. However, the number of features for classification through Machine Learning has to be the same for all conditions. Hence, the maximum number of PCs (four) has been selected for each condition.

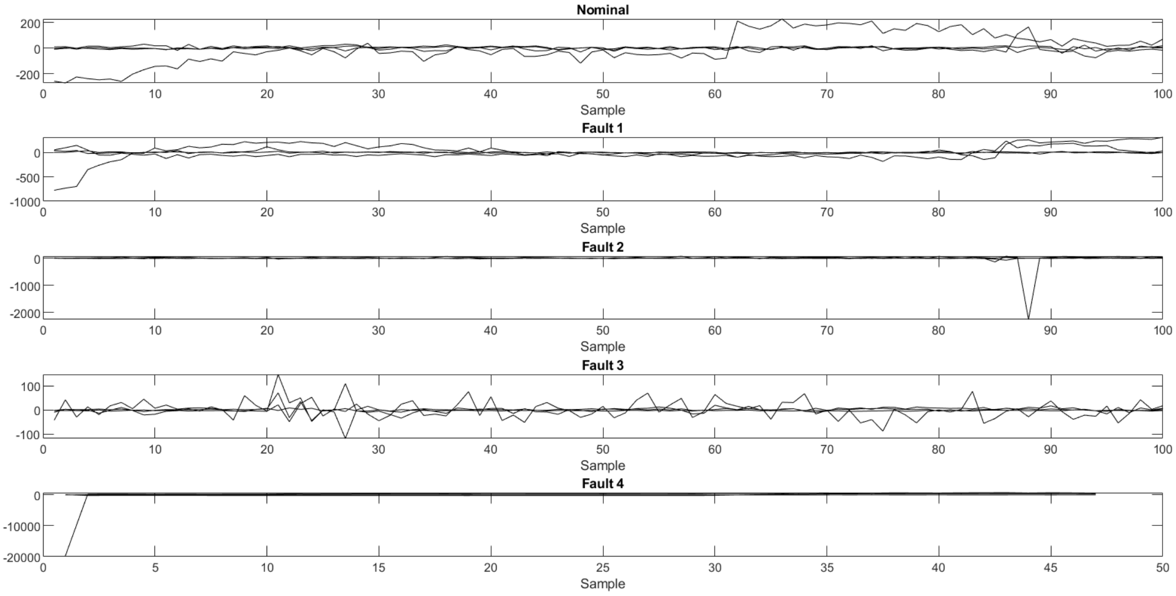


Figure 37 Case Packer. Extracted Principal Components

Case 4. Automatic machinery: Principal Component Analysis. Given the heterogeneity of the provided data (see Table 10), a selection of the most suitable datasets is made. In particular, the period of time in which CM data, context data, and event data are available is selected[5]. At the end, only the data listed in Table 14 have been selected, which include the data related to two components installed in two distinct machinery. As summarized in Table 14, the first machinery (M1 – C1) was monitored for 11 months, while the second machinery (M1 – C2) was monitored for 21 months. For each unit, two different settings were implemented. While for the first machinery, there is only a change from

[5] The reason why even if context data are available, CM data are considered for the analysis will be explained in the section dedicated to the outcomes and considerations of the next chapter.

setting 1 to setting 2, in the second unit, there are two changes: from setting 3 to setting 4, and from setting 4 to setting 3. The 11 variables (temperatures) in the third log (sampling frequency of 1 Hz) have been considered for feature extraction, since they provide valuable information on the operating condition of the machinery. Indeed, as shown in Figure 38, they correspond to the actual values (CM data) of the set values of the machinery (contextual data). Note that the actual temperatures are stored in the third log (1 Hz), while the setting temperatures are stored in the second log. Since the ground truth (the implemented setting, given by the machinery set values) is necessary to evaluate the performance of feature extraction and clustering, only a subset of the total observations, having both the actual and the set values of the temperature, were used in this case.

First, a sampling step has been conducted before extracting relevant features from Condition Monitoring data (Log 1 Hz). The mean value of each variable is computed over a batch of 1800 samples, which correspond to 30 minutes of data. This value has been arbitrarily chosen based on the following factors: the accuracy of prediction, the latency of the algorithm, the memory storage of the possible edge device, and its computational capacity. In this case, the batch of 1800 samples represents a reasonable compromise among all these factors, which demonstrates to smooth the signal with no loss of information.

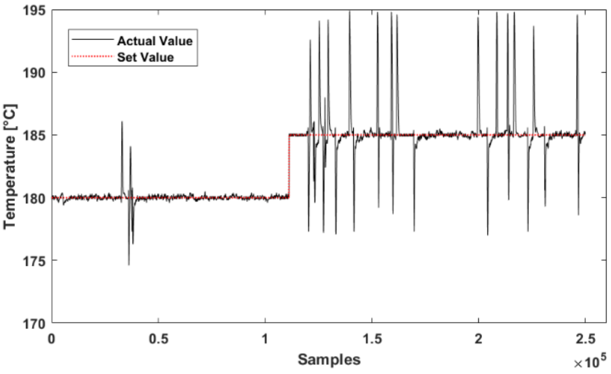


Figure 38 Extruder's settings: set value vs. actual value

Table 14 Extruder: Selected datasets

Unit	Period	Setting
1	From 2017-10-20 to 2017-11-03	1
	From 2017-12-04 to 2018-09-17	2
2	From 2017-01-12 to 2017-06-12	3
	From 2017-06-12 to 2017-06-26	4
	From 2017-06-26 to 2018-09-06	3

Therefore, both batch and incremental PCA are applied to the 11 sampled signals. Figure 39 shows the results of the Incremental PCA, compared with those obtained from a batch PCA, for unit 1 and unit 2, respectively.

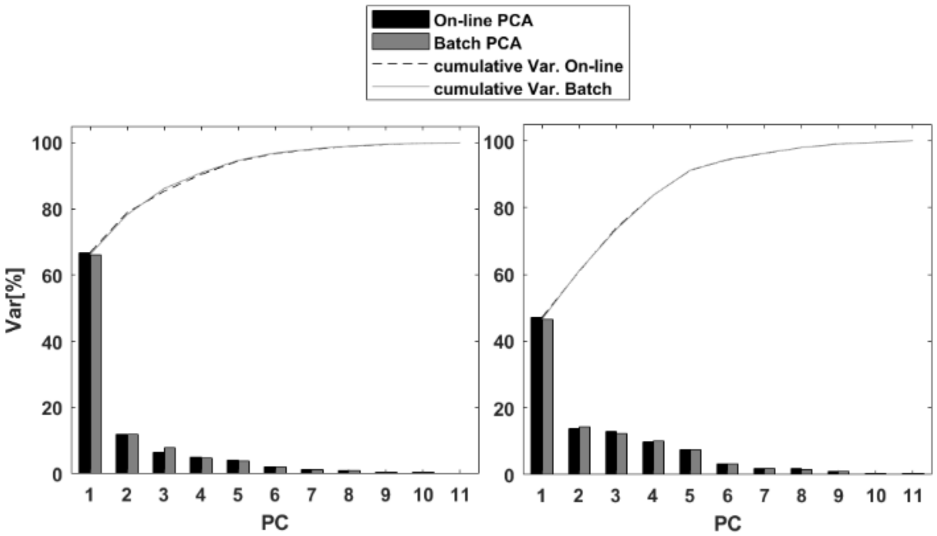


Figure 39 Extruder: Performance of Incremental PCA vs. batch PCA for unit 1 (left) and unit 2 (right)

As shown in Table 15, the first four PCs extracted from the data related to the first unit can retain 90% of the variance during all the analyses. The first five PCs retain the same variance for the second unit. The trend of these PCs for units 1 and 2 are shown in Figure 40, on the left and on the right, respectively.

Table 15 Extruder: Explained variance of the extracted PC's

Unit	PC	Variance	Cumulative Variance	Unit	PC	Variance	Cumulative Variance
1	1	66.45%	66.45%	2	1	46.61%	46.61%
	2	11.88%	78.33%		2	14.49%	61.1%
	3	7.81%	86.14%		3	12.46%	73.56
	4	4.77%	90.91%		4	10.16%	83.72%
					5	7.55%	91.27%

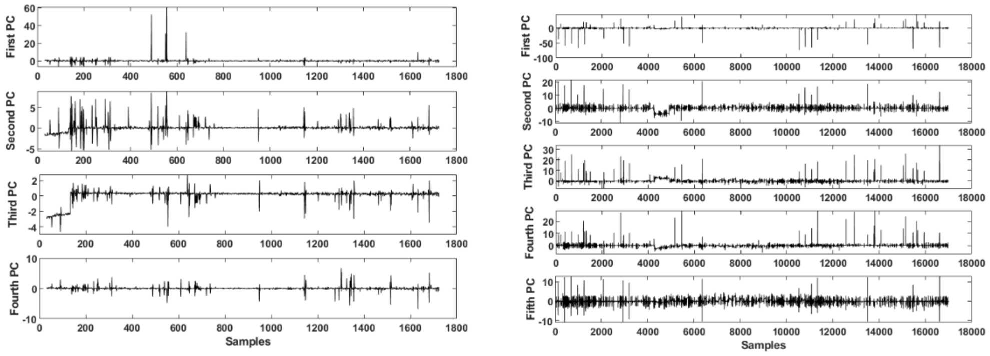


Figure 40 Extruder: Trend of selected PC's of unit 1 (left) and unit 2 (right)

As in the case of the sealing group (vibrations), the extracted PCs for the two units are pretty different among each other.

3.4 Outcomes

The main considerations and outcomes of the data collection and data processing are summarized in this section.

The first outcome concerns the data availability, completeness, and quantity issues. Table 16 shows the characteristics of all available datasets, i.e., the kind of collected signals, the other data (that include contextual, event, and domain knowledge data), the number of nominal conditions and fault conditions, and the duration of corresponding datasets.

Table 16 Characteristics of the available datasets

Case study	CM data	GB (raw signals)	Other data	N unit	N nominal conditions	Duration (s)	N fault conditions	Duration (s)
Test rig	Vibrations	18.19	⚙ 🛠 🧠	1	4	22,212	1 (sudden) 4 (run-to-failure)	Unknown
Sealing group	Displacement	0.05	⚙	1	5	556.8	3	F1: 266.4 F2: 160 F3: 152
Suction Cups	Pressures	0.009	-	1	1	15.99	2	F1: 11.65 F2: 3.52
Case packer group	Currents	0.0004	-	1	1	15	4	F1, F2, F3: 15 F4: 7.05
Sealing group*	Vibrations	2.60	⚙	4	2	1,891.36	1	945.68
Extruder*	Temperature	48.8	🧠	5	4	~2 hours per day for 2 years	0	-

* All the data, including those that were not used for the analysis

As it can be seen, the first evidence is the difference between the quantity of data collected from the test rig and the quantity of data collected by industries. Except for the last two cases, the other datasets contain only a few seconds of data for each condition. In addition, the data associated with a fault condition are very limited in respect to nominal ones. The last two cases are two exceptions. For the sealing group, the tests were conducted exactly for the purpose of fault diagnosis, within an European project. Hence, they are similar to controlled environments. In the last case, the datasets include the signals collected from all sensors installed on five different machines. However, only 10% of them can be used for effective fault diagnosis, which happens because of the different datasets are collected in different periods even of the same machinery. The second consideration that can be drawn from the comparison of available data is that contextual data are missing in most cases. Except for the test rig and the second sealing group, considered controlled environments, the operating condition under which the data have been collected are provided in only one case (sealing group – displacement), in which one only operating condition is available. Therefore, the effect of the operating condition on failures and machinery behaviors cannot be evaluated. Finally, even when a large amount of data is provided, for instance, in the extruder case study, any fault conditions are available. The goal of the analysis was to evaluate the degradation of the screw. However, the mean life of a screw is three years, and therefore, during the data collection period, no degradation occurred. Similarly, for other case studies, it is not possible to build prognostics models since the test cannot cover the whole life of the component.

In other words, concerning the data collection phase, two situations may occur:

1. Companies collect too little data. In this situation, the data are collected during controlled simulations of the different nominal or faulty conditions for a few seconds or minutes. In these cases, almost all methods work well. The goal is to find the best method for the case under analysis. However, the identified methods are not generalizable, and the many doubts about their application into the actual industrial environments are justified. In other words, the built models are hardly used for a real-time PdM strategy.
2. Companies collect too much data without contextual information. Although "more is better" is usually true, when speaking of data collected from industrial machinery, the quality of the data should be the primary goal. As demonstrated in the last case, in which data coming from several machinery for a long period of time are available, if data are collected in different moments, or if data are not enclosed with contextual data, they do not provide valuable information or cannot be even used.

Therefore, a proper data collection methodology should be selected as the first step towards the implementation of a PdM system.

The second outcome concerns the data processing methods. In particular, the focus is on signal segments' length from which to extract the features, the advantages of data reduction, and the qualitative evaluation of the extracted and selected feature sets. The effectiveness of the methods will be evaluated in the next chapter since this process involves classification models. Table 17 summarizes the results regarding the number of selected features and the corresponding reduction of datasets' weight. Note that this table adopted the notation PC analysis for Pearson Correlation analysis.

Table 17 The impact of feature extraction and selection

Case study	GB (raw signals)	Time window (samples/seconds)	Extracted Features	Selected/Constructed Features	KB (Feature sets)
Test rig (system-level)	18.19	12.800 / 1	81	PC analysis = 32 Manual = 3	PC analysis = 9 Manual = 1.15
Test rig (component-level)		12.800 / 1	81	PC analysis = 32 Manual = 1	
Sealing group (displacement)	0.05	800 / 0.008	10	4	34 KB
Suction Cups	0.009	1,340 / 0.134	9	PC analysis = 5 PCA = 9	PC analysis = 21 PCA = 13
Case packer	0.0004	75 / 0.15	9	PC analysis = 4 PCA = 20	PC analysis = 21 PCA = 29
Sealing group (vibrations)*	0.84	25,600/1	14	14	$1.9*10^3$
Extruder*	4.41	18,000/1,800	11	4	$4.5*10^3$

* Only the data used for the analysis

Concerning the length of the signal segments, it can be seen that for data collected at high frequencies, like vibrations, currents, displacements, and pressures, the time window is chosen according to the cycle duration or the number of samples collected in 1 second. In the case of temperature signals, the time window is set equal to 30 minutes. Hence, the first aspect that affects the choice is the kind of signal. The aim of feature extraction and selection is to reduce the amount of data to analyze while keeping as much informative content as possible. Hence, in the case of pretty stable temperature values, as in the last case study, the informative content is retained even if a large time window is chosen. In addition, when the goal is to recognize the operating condition (OCR), the change detection may also be seen after several minutes since it does not affect the machinery's functioning. Hence, the length of the time window also depends on the goal of the analysis. Indeed, in the case of fault detection (for instance, the detachment of a suction cup), a high frequency should also be kept in the feature set since the anomalous condition should be recognized as soon as possible in order to restore the optimal condition. Given that, the choice of the time window length mainly impacts streaming applications, in which the feature vector has to be extracted before the samples included in the next time window arrive in the system. In other words, the latency of the feature extraction and selection algorithm has to be lower than the time window.

Concerning the second point, i.e., the reduction of data quantity, it is evident how signal processing and data reduction notably reduce the amount of data to use as input of diagnostics models. This aspect represents an advantage for two reasons: first, as will be described in the next chapter, a well-reduced feature set improves the performance of classification models and reduces the execution times of the algorithms. Second, it facilitates data transferring and storage. Since machinery and sensors may be in large numbers, the idea is to gather into the cloud only relevant features, at least when a high quantity of raw signals from which features are extracted is already available, so to reduce the amount of data. For instance, it is unnecessary to collect all raw signals of a system in a nominal condition. However, features that represent in a more synthetic way that signals may be helpful to evaluate modifications in the environment. Hence, in some cases, it would be better to transfer only the relevant information into the cloud, i.e., the features, reducing the problems related to the latency of both transfer and analysis and the storage memory required.

Finally, considering the several signal processing and dimensionality reduction techniques, we can conclude that no technique outperforms the other in all cases. In particular, the fact that each case is different represents a constraint for building general solutions applicable in all situations. In addition, the relevance of this consideration increases if considering the difference existing between sets of features extracted from datasets collected from the same machinery. Indeed, in the sealing group (vibration signals), two different products processed by the same machinery generate very different signals. Therefore, we can conclude that the operating condition under which a component operates and external factors strongly affect the trend of the extracted features and, ultimately, the fault diagnosis ability of models. However, time-domain features can represent different conditions in all cases and are fast to compute, making them suitable for streaming feature extraction. On the contrary, the PCA applied directly to raw signals requires long execution times and, eventually, it is applied to a time-domain feature, i.e., the mean, extracted from raw signal segments to decrease the training time (see the case study of the extruder). Finally, it can be seen that the GP provides very different results if trained with the classification-based or the clustering-based objective function. Looking at the second experiment conducted on the test rig, the feature constructed using the classification-based objective function retains the increasing trend corresponding to the degradation of the belt, which may represent an issue for the streaming application of fault detection. On the contrary, clustering-based approaches are more suitable in the case of operating condition recognition because of their nature to create groups of similar (near) data.

Given the primary importance of the data collection and processing phases for the system health management, the design of a proper infrastructure that facilitates the collection of structured data and the reduction of their quantity is a crucial aspect for implementing Predictive Maintenance in Smart Factories. For this reason, a framework for data collection, processing, and transferring is proposed, which represents the third outcome of the activity research presented in this chapter. The framework, depicted in Figure 41, includes two layers.

The edge storage and analytics layer consists of distributed computing devices close to the machines in one or more plants. They should be provided with a storage memory, whose capacity depends on the number of signals collected from each machinery and the sampling frequency. The idea is that raw signals are stored for a given time. Then, based on the output of fault detection and diagnosis (that will be explained in the next chapter), the raw signals may be sent to the cloud or not in certain moments of the day. Then, once the signals have been sent, the data in the temporary storage memory may be overwritten. The analytics part aims to extract relevant features based on the analysis performed in the cloud. Hence, a permanent memory is needed to store the parameters needed for the extraction and selection: the length of the time window and the script for feature extraction and selection. In addition, the analytics part generates the features matrices. These have to be sent into the cloud. Note that the data flow between edge and cloud, in this case, is not necessarily in real-time.

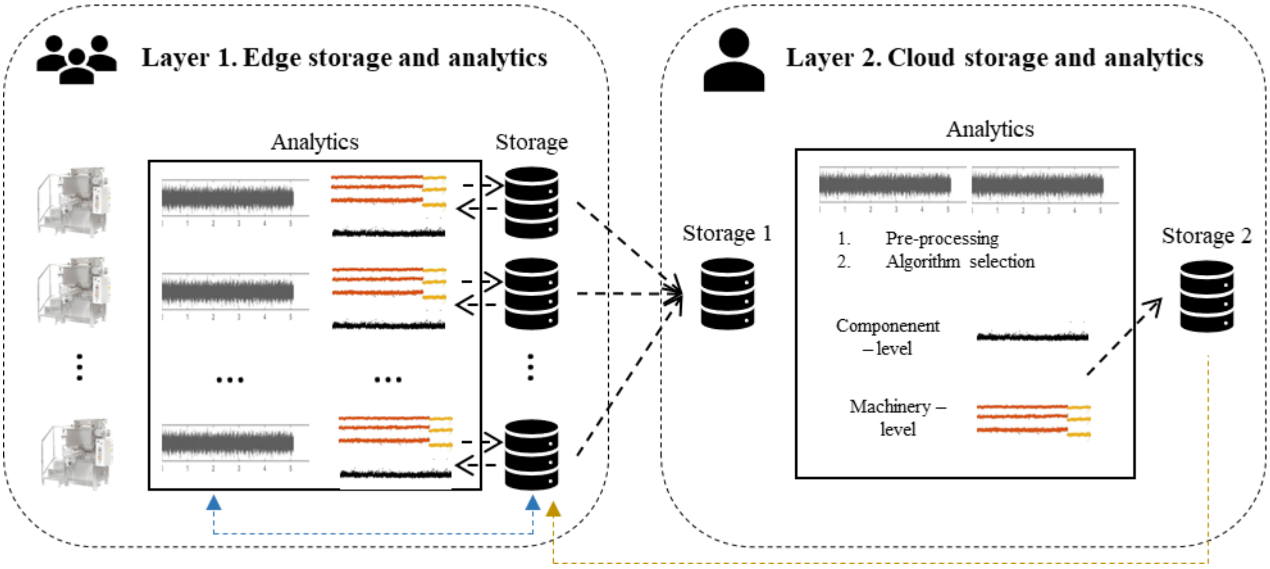


Figure 41 Framework for data collection

In the cloud storage and analytics layer, the analytics part of the cloud consists of data processing, as described in the previous sections. Hence, different algorithms and models are trained to identify the optimal feature sets for each component and system. To facilitate the feature extraction and selection, the data in the storage memory of the cloud can be organized as depicted in Figure 42. In particular, two databases can be obtained. The first one collects the data from the edge storage memory of each machinery. In this database, the following information should be included:

1. The timestamp, to know precisely when each record is collected
2. The client, to identify the location of machinery to which the data refer. In case one only company is involved in the system, the client could correspond to the plant or the manufacturing cell in which the machinery is installed.
3. The machinery, to identify the type of machinery to which the data refer.
4. The component to which the data refer.
5. The setting. It is the operating condition and environmental information at the moment of data collection. It corresponds to the label of the system condition.
6. The condition. It expresses the health condition of the component. It corresponds to the label of the component condition.
7. The CM data (if present). They include all relevant signals referred to the component in that health condition at that time.
8. The features sets. They include all the relevant variables extracted at the edge.
9. The event data. They may be a change in the operating condition of the system or one of the components, a maintenance intervention (e.g., lubrification), or a sudden shutdown of the system.

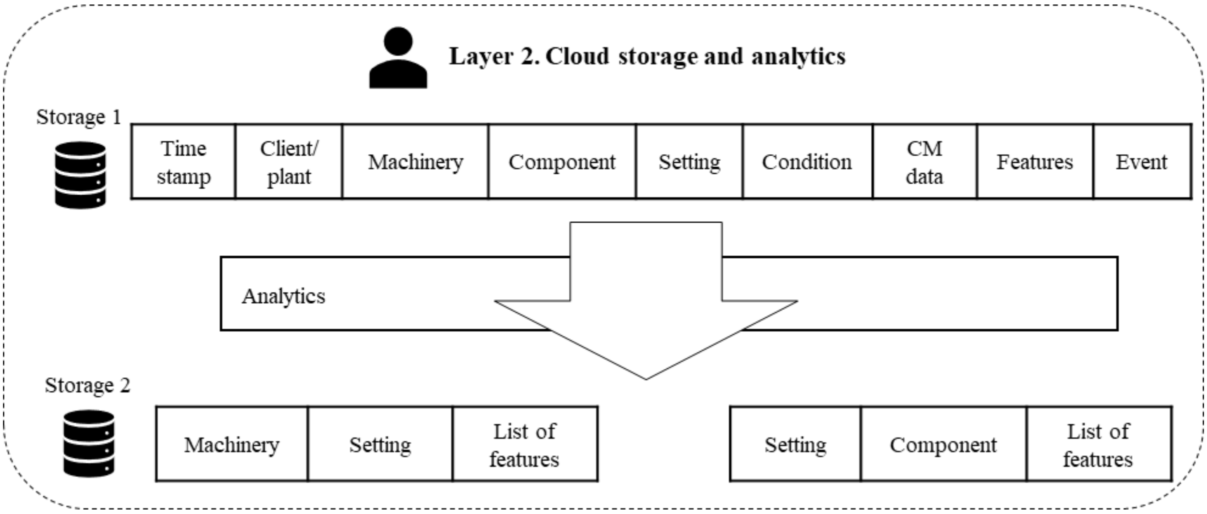


Figure 42 Database organization for data integration

The event data, the CM data, and the feature sets have to be recorded at different times. Records may be inserted according to the highest sampling frequency, and the other fields are filled with default values. In the alternative, the DB may be duplicated for each sampling frequency. In this case, more storage memory is needed, and the analysis is more challenging because of the disaggregation of the information.

3.5 References

Ayvaz, Serkan, and Koray Alpay. 2021. “Predictive Maintenance System for Production Lines in Manufacturing: A Machine Learning Approach Using IoT Data in Real-Time.” *Expert Systems with Applications* 173 (September 2020): 114598. https://doi.org/10.1016/j.eswa.2021.114598.

Balogh, Z., E. Gatial, J. Barbosa, P. Leitão, and T. Matejka. 2018. “Reference Architecture for a Collaborative Predictive Platform for Smart Maintenance in Manufacturing.” *INES 2018 - IEEE 22nd International Conference on Intelligent Engineering Systems, Proceedings*, 000299–000304. https://doi.org/10.1109/INES.2018.8523969.

Bellavista, Paolo, Filippo Bosi, Antonio Corradi, Luca Foschini, Stefano Monti, Lorenzo Patera, Luca Poli, Domenico Scotece, and Michele Solimando. 2019. “Design Guidelines for Big Data Gathering in Industry 4.0 Environments.” *20th IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks, WoWMoM 2019*. https://doi.org/10.1109/WoWMoM.2019.8793033.

Bengio, Yoshua, Aaron Courville, and Pascal Vincent. 2013. “Representation Learning: A Review and New Perspectives.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35 (8): 1798–1828. https://doi.org/10.1109/TPAMI.2013.50.

Bin, G. F., J. J. Gao, X. J. Li, and B. S. Dhillon. 2012. “Early Fault Diagnosis of Rotating Machinery Based on Wavelet Packets - Empirical Mode Decomposition Feature Extraction and Neural Network.” *Mechanical Systems and Signal Processing* 27 (1): 696–711. https://doi.org/10.1016/j.ymssp.2011.08.002.

Calabrese, Matteo, Martin Cimmino, Francesca Fiume, Martina Manfrin, Luca Romeo, Silvia Ceccacci, Marina Paolanti, et al. 2020. “SOPHIA: An Event-Based IoT and Machine Learning Architecture for Predictive Maintenance in Industry 4.0.” *Information (Switzerland)* 11 (4): 1–17. https://doi.org/10.3390/INFO11040202.

Cerrada, Mariela, René Vinicio Sánchez, Diego Cabrera, Grover Zurita, and Chuan Li. 2015. “Multi-Stage Feature Selection by Using Genetic Algorithms for Fault Diagnosis in Gearboxes Based on Vibration Signal.” *Sensors (Switzerland)* 15 (9): 23903–26. https://doi.org/10.3390/s150923903.

Chen, Jiayu, Cuiying Lin, DI Peng, and Hongjuan Ge. 2020. “Fault Diagnosis of Rotating Machinery: A Review and Bibliometric Analysis.” *IEEE Access* 8: 224985–3. https://doi.org/10.1109/ACCESS.2020.3043743.

Cheng, Cheng, Bei Ke Zhang, and Dong Gao. 2019. “A Predictive Maintenance Solution for Bearing Production Line Based on Edge-Cloud Cooperation.” *Proceedings - 2019 Chinese Automation Congress, CAC 2019*, 5885–89. https://doi.org/10.1109/CAC48633.2019.8996482.

Cheng, Xu, Andre Listou Ellefsen, Guoyuan Li, Finn Tore Holmeset, Houxiang Zhang, and Shengyong Chen. 2019. “A Step-Wise Feature Selection Scheme for a Prognostics and Health Management System in Autonomous Ferry Crossing Operation.” In *Proceedings of 2019 IEEE International Conference on Mechatronics and Automation, ICMA 2019*, 1877–82. IEEE. https://doi.org/10.1109/ICMA.2019.8816219.

Fawwaz, Dzaky Zakiyal, and Sang-Hwa Chung. 2020. “Real-Time and Robust Hydraulic System Fault Detection via Edge Computing.” *Applied Sciences* 10 (5933). https://doi.org/10.3390/app10175933.

Fila, Redouane, Mohamed El Khaili, and Mohamed Mestari. 2020. “Cloud Computing for Industrial Predictive Maintenance Based on Prognostics and Health Management.” *Procedia Computer Science* 177: 631–38. https://doi.org/10.1016/j.procs.2020.10.090.

Folino, Gianluigi. 2003. “Algoritmi Evolutivi e Programmazione Genetica : Strategie Di Progettazione e Parallelizzazione Algoritmi Evolutivi e Programmazione Genetica : Strategie Di Progettazione e Parallelizzazione.”

Gan, Meng, Cong Wang, and Chang’an Zhu. 2015. “Multiple-Domain Manifold for Feature Extraction in Machinery Fault Diagnosis.” *Measurement: Journal of the International Measurement Confederation* 75 (November): 76–91. https://doi.org/10.1016/j.measurement.2015.07.042.

Geropp, Berod. 1997. “Envelope Analysis - A Signal Analysis Technique for Early Detection and Isolation of Machine Faults.” *IFAC Proceedings Volumes* 30 (18): 977–81.

Gokhale, M. Y., and Daljeet Kaur Khanduja. 2010. “Time Domain Signal Analysis Using Wavelet Packet Decomposition Approach.” *International Journal of Communications, Network and System Sciences* 03 (03): 321–29. https://doi.org/10.4236/ijcns.2010.33041.

Gomes Teixeira de Almeida, Rui, Silmara Alexandra da Silva Vicente, and Linilson Rodrigues Padovese. 2002. “New Technique for Evaluation of Global Vibration Levels in Rolling Bearings.” *Shock and Vibration*. Vol. 9. IOS Press.

Goyal, Deepam, and B S Pabla. 2015. “Condition Based Maintenance of Machine Tools — A Review.” *CIRP Journal of Manufacturing Science and Technology* 10: 24–35. https://doi.org/10.1016/j.cirpj.2015.05.004.

Guo, Ling, Daniel Rivero, Julián Dorado, Cristian R. Munteanu, and Alejandro Pazos. 2011. “Automatic Feature Extraction Using Genetic Programming: An Application to Epileptic EEG Classification.” *Expert Systems with Applications* 38 (8): 10425–36. https://doi.org/10.1016/j.eswa.2011.02.118.

Han, Te, Dongxiang Jiang, Xiaochen Zhang, and Yankui Sun. 2017. “Intelligent Diagnosis Method for Rotating Machinery Using Dictionary Learning and Singular Value Decomposition.” *Sensors (Switzerland)* 17 (4). https://doi.org/10.3390/s17040689.

Hu, Rongyao, Xiaofeng Zhu, Debo Cheng, Wei He, Yan Yan, Jingkuan Song, and Shichao Zhang. 2017. “Graph Self-Representation Method for Unsupervised Feature Selection.” *Neurocomputing* 220: 130–37. https://doi.org/10.1016/j.neucom.2016.05.081.

Janos Abonyi (2021). Genetic Programming MATLAB Toolbox (https://www.mathworks.com/matlabcentral/fileexchange/47197-genetic-programming-matlab-toolbox), MATLAB Central File Exchange. Retrieved October 26, 2021.

Javed, Kamran, Rafael Gouriveau, and Noureddine Zerhouni. 2017. “State of the Art and Taxonomy of Prognostics Approaches, Trends of Prognostics Applications and Open Issues towards Maturity at Different Technology Readiness Levels.” *Mechanical Systems and Signal Processing* 94: 214–36. https://doi.org/10.1016/j.ymssp.2017.01.050.

Ji, Bongjun, Seunghwan Bang, Hyunseop Park, Hyunbo Cho, and Kiwook Jung. 2019. “Multi-Criteria Decision-Making-Based Critical Component Identification and Prioritization for Predictive Maintenance.” *Industrial Engineering and Management Systems* 18 (3): 305–14. https://doi.org/10.7232/iems.2019.18.3.305.

Khan, W. Z., M. H. Rehman, H. M. Zangoti, M. K. Afzal, N. Armi, and K. Salah. 2020. “Industrial Internet of Things: Recent Advances, Enabling Technologies and Open Challenges.” *Computers and Electrical Engineering* 81: 106522. https://doi.org/10.1016/j.compeleceng.2019.106522.

Lei, Yaguo, Feng Jia, Jing Lin, Saibo Xing, and Steven X. Ding. 2016. “An Intelligent Fault Diagnosis Method Using Unsupervised Feature Learning Towards Mechanical Big Data” 63 (5): 3137–47. https://doi.org/10.1109/TIE.2016.2519325.

Lei, Yaguo, Bin Yang, Xinwei Jiang, Feng Jia, Naipeng Li, and Asoke K. Nandi. 2020. “Applications of Machine Learning to Machine Fault Diagnosis: A Review and Roadmap.” *Mechanical Systems and Signal Processing* 138: 106587. https://doi.org/10.1016/j.ymssp.2019.106587.

Li, Li, Liangsheng Qu, and Xianghui Liao. 2007. “Haar Wavelet for Machine Fault Diagnosis.” *Mechanical Systems and Signal Processing* 21 (4): 1773–86. https://doi.org/10.1016/j.ymssp.2006.07.006.

Liang, B., S. D. Iwnicki, and Y. Zhao. 2013. “Application of Power Spectrum, Cepstrum, Higher Order Spectrum and Neural Network Analyses for Induction Motor Fault Diagnosis.” *Mechanical Systems and Signal Processing* 39 (1–2): 342–60. https://doi.org/10.1016/j.ymssp.2013.02.016.

Lillicrap, Timothy P., Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2009. “Continuous Control with Deep Reinforcement Learning.” *Foundations and Trends in Machine Learning* 2 (1): 1–127. https://doi.org/10.1561/2200000006.

Lin, Jing, and Liangsheng Qu. 2000. “Feature Extraction Based on Morlet Wavelet and Its Application for Mechanical Fault Diagnosis.” *Journal of Sound and Vibration* 234 (1): 135–48. https://doi.org/10.1006/jsvi.2000.2864.

Lin, Shi-Wan, Bradford Miller, Jacques Durand, Graham Bleakley, Chigani Amine, Martin Robert, Murphy Brett, and Mark Crawford. 2017. “The Industrial Internet of Things Volume G1 : Reference Architecture.” *Industrial Internet Consortium White Paper* Version 1.: 58 Seiten.

Lippi, Vittorio, and Giacomo Ceccarelli. 2019. “Incremental Principal Component Analysis: Exact Implementation and Continuity Corrections.” *ICINCO 2019 - Proceedings of the 16th International Conference on Informatics in Control, Automation and Robotics* 1: 473–80. https://doi.org/10.5220/0007743604730480.

Liu, Haining, Chengliang Liu, and Yixiang Huang. 2011. “Adaptive Feature Extraction Using Sparse Coding for Machinery Fault Diagnosis.” *Mechanical Systems and Signal Processing* 25 (2): 558–74. https://doi.org/10.1016/j.ymssp.2010.07.019.

Mehrjou, Mohammad Rezazadeh, Norman Mariun, Norhisam Misron, and Mohd Amran Mohd Radzi. 2017. “Analysis of Statistical Features Based on Start-up Current Envelope for Broken Rotor Bar Fault Detection in Line Start Permanent Magnet Synchronous Motor.” *Electrical Engineering* 99 (1): 187–201. https://doi.org/10.1007/s00202-016-0404-3.

Muni, Durga Prasad, Nikhil R. Pal, and Jyotirmoy Das. 2004. “A Novel Approach to Design Classifiers Using Genetic Programming.” *IEEE Transactions on Evolutionary Computation* 8 (2): 183–96. https://doi.org/10.1109/TEVC.2004.825567.

Neshatian, Kourosh, Mengjie Zhang, and Peter Andreae. 2012. “A Filter Approach to Multiple Feature Construction for Symbolic Learning Classifiers Using Genetic Programming.” *IEEE Transactions on Evolutionary Computation* 16 (5): 645–61. https://doi.org/10.1109/TEVC.2011.2166158.

Peng, Bo, Shuting Wan, Ying Bi, Bing Xue, and Mengjie Zhang. 2020. “Automatic Feature Extraction and Construction Using Genetic Programming for Rotating Machinery Fault Diagnosis.” *IEEE Transactions on Cybernetics*, 1–15. https://doi.org/10.1109/TCYB.2020.3032945.

Peng, Z. K., and F. L. Chu. 2004. “Application of the Wavelet Transform in Machine Condition Monitoring and Fault Diagnostics: A Review with Bibliography.” *Mechanical Systems and Signal Processing*. Academic Press. https://doi.org/10.1016/S0888-3270(03)00075-X.

Rai, Akhand, and S. H. Upadhyay. 2017. “Bearing Performance Degradation Assessment Based on a Combination of Empirical Mode

Decomposition and K-Medoids Clustering." *Mechanical Systems and Signal Processing* 93: 16–29. https://doi.org/10.1016/j.ymssp.2017.02.003.

Ricci, Roberto, and Paolo Pennacchi. 2011a. "Diagnostics of Gear Faults Based on EMD and Automatic Selection of Intrinsic Mode Functions." *Mechanical Systems and Signal Processing* 25 (3): 821–38. https://doi.org/10.1016/j.ymssp.2010.10.002.

———. 2011b. "Diagnostics of Gear Faults Based on EMD and Automatic Selection of Intrinsic Mode Functions." *Mechanical Systems and Signal Processing* 25 (3): 821–38. https://doi.org/10.1016/j.ymssp.2010.10.002.

Sajid, Sufiyan, Abid Haleem, Shashi Bahl, Mohd Javaid, Tarun Goyal, and Manoj Mittal. 2021. "Data Science Applications for Predictive Maintenance and Materials Science in Context to Industry 4.0." *Materials Today: Proceedings* 45: 4898–4905. https://doi.org/10.1016/j.matpr.2021.01.357.

Sarih, Houda, Ayeley P. Tchangani, Kamal Medjaher, and Eric Pere. 2019. "Data Preparation and Preprocessing for Broadcast Systems Monitoring in PHM Framework." *2019 6th International Conference on Control, Decision and Information Technologies, CoDIT 2019*, no. Cm: 1444–49. https://doi.org/10.1109/CoDIT.2019.8820370.

Schofield, Finn, and Andrew Lensen. 2020. "Evolving Simpler Constructed Features for Clustering Problems with Genetic Programming." In *2020 IEEE Congress on Evolutionary Computation, CEC 2020 - Conference Proceedings*. https://doi.org/10.1109/CEC48606.2020.9185575.

Siddhartha, B., Arunkumar P Chavan, G. Krishna, and K.N. Subramanya. 2020. "IoT Enabled Real-Time Availability and Condition Monitoring of CNC Machines." In *The 2020 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS) IoT*, 78–84.

Su, Xuanyuan, Hongmei Liu, and Laifa Tao. 2020. "TF Entropy and RFE Based Diagnosis for Centrifugal Pumps Subject to the Limitation of Failure Samples." *Applied Sciences (Switzerland)* 10 (8). https://doi.org/10.3390/APP10082932.

Tang, Jiliang, Salem Alelyani, and Huan Liu. 2014. "Feature Selection for Classification: A Review." In *Data Classification Algorithms and Applications*, edited by Charu C. Aggarwal, 1st Editio, 2–28. New York.

Tiddens, Wieger, Jan Braaksma, and Tiedo Tinga. 2020. "Exploring Predictive Maintenance Applications in Industry." *Journal of Quality in Maintenance Engineering*. https://doi.org/10.1108/JQME-05-2020-0029.

Vanneschi, Leonardo, and Riccardo Poli. 2012. "Genetic Programming — Introduction, Applications, Theory and Open Issues." In *Handbook of Natural Computing*, 709–39. Berlin: Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-92910-9.

Wang, De, Feiping Nie, and Heng Huang. 2014. "Unsupervised Feature Selection via Unified Trace Ratio Formulation and K-Means Clustering (TRACK)." *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 8726 LNAI (PART 3): 306–21. https://doi.org/10.1007/978-3-662-44845-8_20.

Wang, Fengtao, Jian Sun, Dawen Yan, Shenghua Zhang, Liming Cui, and Yong Xu. 2015a. "A Feature Extraction Method for Fault Classification of Rolling Bearing Based on PCA." *Journal of Physics: Conference Series* 628 (1). https://doi.org/10.1088/1742-6596/628/1/012079.

———. 2015b. "A Feature Extraction Method for Fault Classification of Rolling Bearing Based on PCA." *Journal of Physics: Conference Series* 628 (1). https://doi.org/10.1088/1742-6596/628/1/012079.

Wang, Hao, Guangming Dong, and Jin Chen. 2020. "Application of Improved Genetic Programming for Feature Extraction in the Evaluation of Bearing Performance Degradation." *IEEE Access* 8: 167721–30. https://doi.org/10.1109/access.2020.3019439.

Wang, Jinrui, Shunming Li, Xingxing Jiang, and Chun Cheng. 2017. "An Automatic Feature Extraction Method and Its Application in Fault Diagnosis." *Journal of Vibroengineering* 19 (4): 2521–33. https://doi.org/10.21595/jve.2017.17906.

Xiaoli, Xu, Zuo Yunbo, and Wu Guoxin. 2011. "Design of Intelligent Internet of Things for Equipment Maintenance." *Proceedings - 4th International Conference on Intelligent Computation Technology and Automation, ICICTA 2011* 2: 509–11. https://doi.org/10.1109/ICICTA.2011.412.

Xie, Ting, Pengfei Ren, Taiping Zhang, and Yuan Yan Tang. 2018. "Distribution Preserving Learning for Unsupervised Feature Selection." *Neurocomputing* 289: 231–40. https://doi.org/10.1016/j.neucom.2018.02.032.

Zhang, Chao, Erwei Li, Liang Liu, and Yong Zhou. 2018. "Dimension Reduction Method For Data-Driven PHM Sysyem Based on ULDA Algorithm." *ICIC Express Letters* 9 (8): 765–71. https://doi.org/10.24507/icicelb.09.08.765.

Zhang, X., Q. Zhang, H. Li, Y. Sun, and X. Qin. 2017. "Fault Diagnosis Using Locality Sensitive Discriminant Analysis for Feature Extraction." *2017 Prognostics and System Health Management Conference (PHM-Harbin)*, 1–6. https://doi.org/10.1109/PHM.2017.8079198.

Zhu, Xiaofeng, Xuelong Li, Shichao Zhang, Chunhua Ju, and Xindong Wu. 2017. "Robust Joint Graph Sparse Coding for Unsupervised Spectral Feature Selection." *IEEE Transactions on Neural Networks and Learning Systems* 28 (6): 1263–75. https://doi.org/10.1109/TNNLS.2016.2521602.

Zhu, Yonghua, Xuejun Zhang, Ruili Wang, Wei Zheng, and Yingying Zhu. 2018a. "Self-Representation and PCA Embedding for

Unsupervised Feature Selection.” *World Wide Web* 21 (6): 1675–88. https://doi.org/10.1007/s11280-017-0497-2.

———. 2018b. “Self-Representation and PCA Embedding for Unsupervised Feature Selection.” *World Wide Web* 21 (6): 1675–88. https://doi.org/10.1007/s11280-017-0497-2.

4. The Integration Of Learning Paradigms For System Health Management

The previous chapter addressed the problems of Condition Monitoring data, contextual data, and event data collection from machinery. Moreover, the methods for data processing that allow extracting valuable information for effective fault diagnosis were also described and applied to several case studies. As a result, a framework for data collection and processing is proposed based on the interaction of edge and cloud computing and on the concept of keeping only the relevant features in memory. In addition, a database structure for data integration and organization in the cloud is proposed, which facilitates the data analysis.

The next step in the system health management is fault diagnosis. As described in Chapter 2, fault diagnosis consists of 3 steps, i.e., fault detection, fault identification, and fault isolation. In particular, fault detection deals with the recognition of behaviors different from the nominal one and can be addressed through anomaly detection algorithms. Instead, fault identification deals with recognizing the specific fault, e.g., a crack in the raceway ring of a rolling bearing. To this aim, classification algorithms are adopted to build models that predict the health condition to which a new observation belongs. As pointed out in many parts of this dissertation, the necessary condition for fault detection and diagnosis is the availability of the correct data. In particular, in the first case, the data collected when the system operates in a nominal condition are sufficient. Then, everything considered different from the nominal learned behavior falls into one category, i.e., anomalous behavior. On the contrary, the data needed for fault identification includes all possible, i.e., nominal and fault, modes in which the system could operate to identify the system's condition and determine the kind of maintenance intervention to perform.

However, as demonstrated in the previous chapter, collecting a large amount of data during nominal and faulty conditions is critical for many industries. In particular, the following issues may be pointed out:

1. Data availability. It refers to the lack of data collected during all possible conditions in which a system may operate during its life (Y. Hu et al. 2017). In particular, while it is easy to obtain data in nominal conditions, data in fault conditions are hard to obtain given the loss of productivity and the quality, the environmental and safety issues arising from machinery working in fault conditions. In addition, failures may happen for several reasons, which might be unpredictable.

2. Data quantity. It refers to the amount of data collected for each condition and is strictly correlated to the data availability issue. However, in this case, the focus is on the number of observations and duration of each test. Either data are collected continuously from machinery or batches of data are downloaded from the PLC at certain moments, the quantity of data corresponding to the nominal behavior is always larger than data collected during fault conditions. The insufficient quantity of fault data implies that unbalanced datasets are available, making classification models less accurate.
3. Data incompleteness. Complete data should include condition monitoring data and the condition they refer to, i.e., the label. In addition, the event data, like the occurrence of a failure, the maintenance interventions, or the presence of transients, speed up the data pre-processing phase, which usually takes 70% of the time of the whole process of data analysis. Finally, since the system's settings, e.g., the load or the braking force, affect its behavior, contextual data are necessary to build models independent of, or specific for, a particular condition. While it is easy to obtain all this information in controlled environments, like laboratories, industries struggle with getting all required contextual information due to time, economic, and cultural constraints.

All these aspects are especially critical for Original Equipment Manufacturers, which only collect the data during the brief tests conducted for the quality assessment. In some cases, OEMs may get the data from their clients' machinery, partially solving the data availability and quantity issues. However, the problem of data incompleteness notably increases due to the limited clients' disposition to share sensitive information, like the production rate or the machine settings. In addition, clients are often spread worldwide, and the data transferring is not trivial. Firstly, because of the high volume of the data to transfer, which would require a long time and a large bandwidth. Secondly, for the privacy issue. Industries are still skeptical, and networks are still subject to cyberattacks.

Given these issues, the traditional offline and batch fault diagnosis, which relies on many labeled data, may be ineffective and impracticable. The framework for data collection and processing proposed in the previous chapter may solve this issue. However, the analysis of the data at the edge requires a combination of different approaches that consider the peculiarities of streaming data. In other words, the system health management has to handle unlabelled and unbalanced datasets and operate in an evolving environment. These two aspects are strictly correlated and can be addressed by integrating offline and batch fault diagnosis with online and streaming models to recognize both known conditions (fault diagnosis) and detect unknown behaviors (fault detection) (Y. Xu et al. 2017).

According to (Gouriveau, Ramasso, and Zerhouni 2013), two strategies exist to cope with unlabeled and unbalanced datasets. The first one considers evolving algorithms, which rely on a training scheme that enables repeating the learning phase as required to manage state discovery (as new data are available), notably when data are imbalanced. The second strategy deals with incompleteness and uncertainty of labels by taking advantage of semi-supervised learning approaches. A solution that integrates the two strategies may consider novelty detection as part of the incremental or evolving learning framework (Saucedo-Dorantes et al. 2020).

To this aim, a health management system should include

1. Classification algorithms that can classify known conditions (fault diagnosis)
2. Anomaly detection algorithms that can detect novel behaviors (fault detection)
3. Incremental algorithms able to deal with streaming data peculiarities (novelty detection)

Therefore, classification models for fault diagnosis, models for fault detection, and novelty detection approaches and frameworks will be revised in this chapter. The theoretical background of some models for each of the three tasks will also be provided. Finally, they will be applied to the case studies listed in the previous chapter.

4.1. Literature review and theoretical background

In evolving environments, observations are assumed to arrive in a data stream, and no ground truth for class labels is available in the data stream at all times, except for the initial training set (Yue Zhu, Ting, and Zhou 2018). When new unlabelled observations arrive in the data streams, new labels may emerge. The main goal of incremental learning, which also includes novelty detection, is to recognize abnormal conditions (anomalies) that have never occurred before while classifying machinery conditions (nominal and fault health condition) as one of the known classes if there is no novel abnormality (Z. Yang, Long, et al. 2021). In other words, incremental learning approaches allow detecting the so-called concept drift, which occurs in the data stream when a machinery condition changes. In addition, novelty detection algorithms establish whether the concept drift corresponds to a known machinery operating condition or a novel operating condition. In the first case, a classifier pre-trained on the available training data is applied to assign one known label to the observation. In the second case, a new label has to be created and the classification model updated. This problem is also known as novelty detection in a data stream, rarely addressed in the literature (Y. Wang et al., 2021). In other words, novelty detection deals with detecting new data samples that an ML model was not previously aware of (J. Zhang et al. 2006).

Novelty detection is traditionally considered a one-class classification problem, in which only one class, i.e., the nominal class, is known. Hence, it usually requires a dataset including many data samples belonging to the same class. Unlike anomaly detection and outlier detection, novelty detection aims to find a set of points not explained from the diagnosis model, instead of one single point differing from the nominal or known data (Pimentel et al., 2014). In addition, novelty detection is also included in multi-class systems, where two or more normal classes exist, and a condition is considered novel if it differs from all known classes (Chan et al., 2020). In other words, the problem becomes recognizing novelties and, at the same time, classifying the known instances into two or more diverse classes. The first relevant papers on this topic were published in early 2000. In (Y. Li, Pont, and Barrie Jones 2002), the authors highlighted how threshold-based classifiers might be appropriate for condition monitoring and fault diagnosis systems as they allow interpreting an unknown fault when none of the classifier output exceeds the defined threshold. Recent studies rely on novelty detection and diagnosis models to discover new scenarios, starting from data related to a healthy condition and known fault conditions. In (Cariño et al. 2018), an ensemble-based classifier for novelty detection and an evolving classifier for diagnosis are separately applied to a different set of features and then integrated into a unique methodology to discover new patterns in case only data related to a healthy condition is available. In that work, a measurement labeled as ‘unknown’ can represent either an outlier, a new fault, or a new operating condition. Thus, machine user intervention is required to verify which of the above cases the detected novelty refers to. In (Cariño et al. 2020), a hybrid approach for multi-modal signal analysis, novelty detection, and fault diagnosis is presented. It aims at detecting and incrementally including newly discovered scenarios based on time-frequency features. In (Dyer, Capo, and Polikar 2014), a new scenario named Initially Labelled Streaming Environment (ISLE) is defined. It is characterized by an infinite verification latency, meaning that no labeled data are received after initialization. In this context, the authors developed a new algorithm named COMPOSE, which learns drifting concepts from a streaming and non-stationary environment that provides only unlabelled data after initialization. An application to fault diagnosis of COMPOSE is introduced in (Y. Hu et al. 2016). The resulting framework detects gradual and abrupt changes; then, a classifier is updated to include a new class, and a feature set is selected to represent the new class.

Another approach to novelty detection sees novelty as a concept, an abstraction of cohesive and representative examples, that introduces characteristics different from known concepts. In this sense, several clustering algorithms for novelty detection in data streams have been proposed, such as OnLine Novelty Detection and Drift Detection Algorithm (OLINDDA) (Spinosaa, de Carvalhoa, and Gamab 2009), Higia (Garcia et al. 2019), MINAS (MultIclass learNing Algorithm for data Streams) (de Faria, Ponce de Leon Ferreira Carvalho, and Gama 2016). The idea behind these algorithms is to learn a decision model based on labeled data during the offline training phase; then, during the online phase, novelty patterns are recognized as unknown sets, or micro-clusters, made of examples that the model does not explain. Even in these cases, the offline decision model can represent different classes. Other clustering-based approaches for novelty detection do not require the training of models with the nominal class. Evolving Intelligent Systems (EIS) can handle streaming data and adapt their structure and parameters depending on the input data. EISs can be seen as an evolution of the Fuzzy Rule-Based models (FRB), i.e., sets of linguistic statements in the form IF-THEN, where conditions and consequences are associated with fuzzy concepts, i.e., linguist terms (L. Wang 1992). Evolving FRB classifiers can classify the data from scratch. Thus, they can start building the model sample by sample from the first input data. Evolving clustering algorithms can be used to identify the antecedent part of evolving FRB classifiers. Clustering algorithms rely on the idea that multiple clusters correspond to different system modes (Agelov and Zhou 2006). As a new input data arrives, the system integrates it into existing clusters if the data is compatible with the existing model structure and adapts the local parameters of the corresponding cluster (parameter adaptation); otherwise, the algorithm creates a new cluster (structure evolving) (Kasabov and Filev 2006). In (Inacio, Lemos, and Caminhas 2015), a recursive clustering algorithm incorporating a drift detection method is proposed, in which the clustering updating process depends not only on the similarity measure but also on the monitoring changes in the input data flow, which gives the algorithm greater robustness to the presence of outliers and noise. However, these methods require the membership function to be determined *a priori*. In (Angelov and Yager 2011), a new way of determining the antecedent part for FRB has been introduced. The antecedent part is called AnYa system and is non-parametric. The membership function is directly extracted from the data and represents their density and distribution. Hence, the cluster concept is replaced by the concept of cloud, which has no boundaries, could assume any shape, and is not represented by any centers or prototypes. Therefore, in contrast to traditional clustering methods, in AnYa there is no need to use a particular kind of distance; local and global density, which can be calculated recursively as new input arrives, are the only variables affecting the cloud shape and the point assignment process. Based on AnYa fuzzy systems, an evolving system is proposed in (Andonovski et al. 2018), which can protect from the addition of new clouds based on outliers and allows removing less active and informative clouds.

To summarize, novelty detection approaches may be classified according to two criteria: the learning paradigm, which can be supervised or unsupervised, and the necessity of a training phase before the streaming application. Hence, three different approaches can be distinguished, as shown in Figure 43:

1. Classification-based approaches with training on one or more classes (e.g., COMPOSE)
2. Clustering-based approaches with training on one or more classes (e.g., OLINDDA, MINAS)
3. Clustering-based approaches that can be applied from scratch (e.g., ADP)

In addition, classification-based approaches may adopt an incremental or non-incremental learning paradigm, depending on whether the classification model is re-trained when a novel class is detected. On the contrary, all clustering-based approaches adopt an incremental learning paradigm.

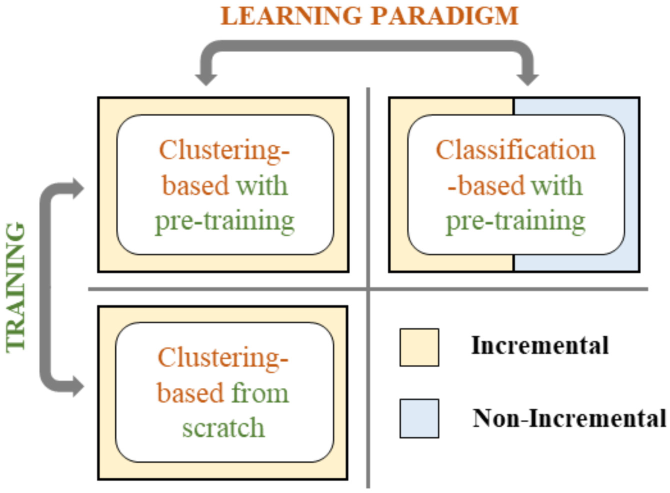

LEARNING PARADIGM
TRAINING
Clustering-based with pre-training
Classification-based with pre-training
Clustering-based from scratch
Incremental
Non-Incremental

Figure 43 Classification of novelty detection approaches in evolving environments

More recently, deep learning models, like autoencoders and Long Short-Term Memory (LSTM) networks (Z. Yang, Gjorgjevikj, et al. 2021), (Park et al. 2019) are used to detect novelties and simultaneously classify known normal and faulty conditions.

4.1.1. Supervised learning for fault diagnosis

In a linear classification problem with two classes, the goal is to find an optimal hyperplane that best separates the two classes. Given a dataset made up of n m-dimensional samples $X = \{\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n\}$, $\bar{x}_i \in \mathbb{R}^m$ associated with the target set $Y = \{y_1, y_2, \dots, y_n\}$, where $y_i \in \{0,1\}$, the weight vector made of m continuous components can be defined as in Eq. (24)

$$W = (w_1, w_2, \dots, w_m)^T \text{ where } w_i \in \mathbb{R} \qquad (24)$$

Then, the quantity z can be defined as in Eq. (25), which represents the value determined by the hyperplane equation.

$$z = \bar{x} \cdot \bar{w} = \sum_i w_i x_i \quad \forall \bar{x} \in \mathbb{R}^m \qquad (25)$$

Therefore, in a binary problem, if the coefficients' set has been correctly defined, the output is given by Eq. (26)

$$z = \begin{cases} 1, & z \geq 0.5 \\ 0, & z < 0.5 \end{cases} \qquad (26)$$

The goal of classifiers is to optimize the values of the vector W to minimize the classification error.

Among the most adopted classifiers for diagnostics, there are Decision Trees, k-Nearest Neighbour (k-NN), Artificial Neural Networks (ANNs), Support Vector Machines (SVMs) (R. Liu et al. 2018), and Deep Learning models (S. Khan and Yairi 2018), such as Deep Belief Networks (DBNs), Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).

Decision Trees are heuristic methods that classify an object into a certain number of classes, defined by the user, based on its attribute values. They recursively partition the instance space, starting from a root node containing all samples, and dividing samples in each node into two (binary trees) or more (multi-split trees) disjoint sets (nodes), based on a predefined splitting criterion until a defined termination criterion is met. The children nodes are characterized by a higher uniform degree of the target class than the parent nodes. Nodes with no outgoing edges are called leaves: observations in each leaf are labeled with the most representative class (Figure 44). Alternatively, the leaf may hold a probability vector indicating the probability that the target attribute has a specific value. The probability is estimated as the class frequency among training instances that belong to the leaf for each leaf. Each path from the root node to a leaf is called the decision rule and is used to classify new observations. When building a classification tree, two parameters have to be set:

- the splitting rule identifies the best target attribute among those available and selects the best splitting criterion for making descendent nodes;
- the stopping criterion is evaluated for each node to establish whether it is a leaf.

Splitting criteria can be univariate or multivariate, depending on whether the splitting is based on a single attribute value or on a function of different attribute values. Univariate criteria are often adopted: they are based on the function evaluation, called impurity, which expresses the degree of homogeneity of the observations in a node (Vercellis, 2009). Even though large decision trees are more accurate, they may lead to overfit the data. Furthermore, a large number of leaves reduces the interpretability of the decision rule. For these reasons, an appropriate stopping criterion has to be defined. The algorithm can terminate when: (1) all instances in the training set belong to a single value of the target attribute; (2) the maximum depth of the tree has been reached; (3) the number of cases in the terminal node is less than the minimum number of cases for parent nodes; (4) if the nodes are split, the number of cases in one or more child nodes would be less than the minimum number of cases for child nodes; (5) the best splitting criteria is not greater than a certain threshold (Rokach &

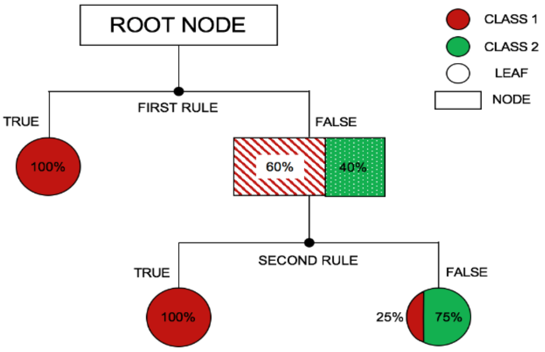


Figure 44 Example of a binary classification tree

Maimon, 2015). Classification trees are non-parametric models that require low computational speed and are easy to understand. However, their heuristic procedure can get stuck in local minima (Khan and Yairi, 2018). For this reason, the depth of the three is often considered during the hyper-parameters optimization process in order to find the best trade-off between prediction accuracy on the training set and the generalization error (Yadav and Jadhav 2021).

The Support Vector Machine (SVM) finds its theoretical foundations in statistical learning theory (STL). Given a set of input features with the associated labels, SVM constructs a hyper-plane that separates the hyper-space into two or more classes to achieve the maximum separation between the classes and minimize the expected generalization error. This process leads to the formation of two hyper-planes (bounding planes) parallel to the separating plane, which are located at a certain distance (margin) from each other (Figure 45). Points on the bounding planes at the minimum distance from the separating hyperplane are the support vectors: they define the position of the separating plane generated by the classifier in the attribute space and correspond to the most representative observations for each target class. In the case of binary classification and linearly separable data, the separating hyperplane is given by (Widodo and Yang 2007):

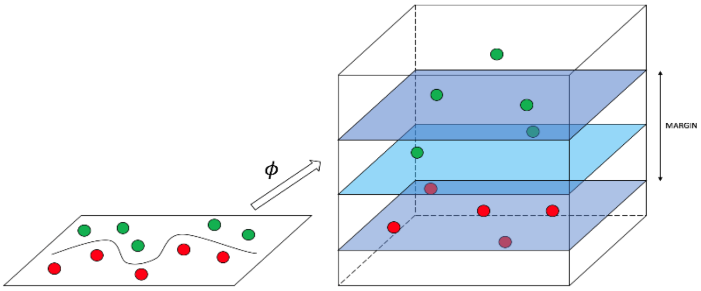


Figure 45 Separating plane and boundaries planes for a linearly separable dataset

$$f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b = 0 \tag{27}$$

where, $\boldsymbol{x}_i(i = 1 \dots M)$ are the input data, and M is the number of samples. $\boldsymbol{w}$ (M-dimensional vector) and b (scalar) define the position of the separating plane. Supporting hyperplanes are given by

$$\boldsymbol{w}^T\boldsymbol{x} - b = 1\,,\ \ \boldsymbol{w}^T\boldsymbol{x} - b = -1 \tag{28}$$

Denoting the margin with $\delta = \frac{2}{\|\boldsymbol{w}\|}$, the optimization problem can be formulated as follows

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{M} \xi_i$$
$$\text{s.to} \begin{cases} y_i(\boldsymbol{w}^T\boldsymbol{x} - b) \geq 1 - \xi_i \\ \xi_i \geq 0 \end{cases} \quad i \in M \tag{29}$$

where C is the penalty for misclassifications and ξ_i is the distance between the margin and sample x_i, laying on the wrong side of the margin. The constraints force each point x_i to lie in the half-space corresponding to the class value y_i. The problem can be solved via Lagrangian duality. It also allows identifying the support vectors associated with positive Lagrange multipliers in the optimal solution of the dual problem. Given the Lagrangian function of the model, the optimal solution can be found by setting the partial derivatives for the variables $\boldsymbol{w}$ and b of the primal problem equal to zero. The objective of the dual problem can be obtained by replacing the obtained equality constraints in the Lagrangian function. The decision function used by the SVM algorithm to classify a new observation **x** is given by the not linear function in Eq. (30)

$$f(x) = sgn\left(\sum_{i,j=1}^{m} \alpha_i\, y_i \boldsymbol{x}_i \boldsymbol{x}_j + b\right) \tag{30}$$

SVM can also be used in non-linear classification tasks. Non-linearly separable original space might be transformed into a linearly separable space through a kernel, i.e., a function that maps the *n*-dimensional input vector into *l*-dimensional feature space. Several Kernel functions, like linear, polynomial, and Gaussian Radial Basis Function (RBF), can be used. The choice is crucial since the kernel defines the feature space in which the training set observations will be classified (Dhiman et al., 2021). For this reason, the kernel and the penalty error are the primary hyperparameters to optimize (Prosvirin, Duong, and Kim 2019). SVM has good generalization capabilities with few training data, and a hyperplane can always separate data from two or more categories. It produces better accuracy and generalization capabilities than any other classification model, especially when facing large problems.

k-Nearest Neighbour (KNN) algorithm is an instance-based learning algorithm in which new observations are classified by comparing them with the most similar in the training set. The assumption is that input data of the same class should be closer in the features space. Therefore, the algorithm computes the distance between a new input data and all the training set observations: the new observation will be assigned to the class of its *k* nearest neighbors (Figure 46). Alternatively, the method can make discrete predictions by reporting the relative frequency of the classes in the neighborhood of the prediction point (Holmes and Adams 2002). Three decisions have to be taken when building a k-NN classifier: the distance metric, the number of neighbors, and the combination function for observations in the training set. The distance between two observations is a measure of their degree of similarity. Among the several distance metrics, the Euclidean distance is the most adopted. In any case, attribute values should be normalized before their computation. The number of neighbors, *k*, is a critical issue because it highly influences classification accuracy. Large values of *k* reduce the effect of outliers but reduce the boundaries evidence between two classes; on the other hand, small values of *k* make the algorithm particularly sensitive to the presence of anomalous values, causing the over-fitting of the data. Finally, similar samples in the training set can combine in different ways to classify a new observation. The primary combination function is the so-called unweighted voting: it calls for establishing how many observations have to be involved in the classification, *k*, and comparing the new observation with its *k* nearest neighbors. An advanced combination function is weighted voting, which considers the significant influence of the nearest neighbors by assigning a vote to each of them. The weight is inversely proportioned to the distance between neighbors and the observation to classify. k-NN shows an advantage of simple implementation. It requires less computation power during the training than other methods but more effort during the classification process.

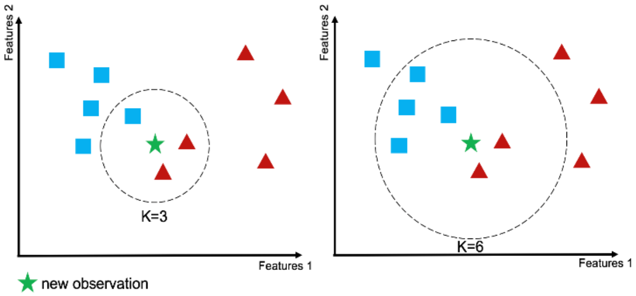


Figure 46 K-Nearest Neghbor

Artificial neural networks (ANNs) are learning algorithms constituted by sets of nodes connected by weighted edges. The adjustment of weights is based on a pre-established rule and represents the learning process. Different connection modes and learning rules generate different kinds of neural networks. The typical structure adopted in the fault diagnosis is the feedforward neural network (FFNN), a layered, acyclic network in which connections are allowed only between nodes belonging to two consecutive layers. It contains an input layer, one or more hidden layers, and an output layer, each having a certain number of nodes (Figure 47). The number of nodes in the input layer depends on the dataset's number and type of attributes. In the output layer, it depends on the specific classification task; the number of hidden layers and the number of nodes in each of them is established by the user, who should consider that more nodes produce higher accuracy and flexibility but make the network more sensitive to the overfitting; few nodes may lead to an unacceptable training accuracy (Larose 2005). Data inputs are collected in the input layer, which is only used for passing input values along with the first hidden layer. Then, they are linearly combined with weights through a combination function, which gives a scalar value, named net:

$$net_j = \sum_i W_{i,j} x_{i,j} \tag{31}$$

where $x_{i,j}$ is the i*th* input to node j and $W_{i,j}$ is the weight associated with the i*th* input to node j. Within each node, the net function is then passed as input to an activation function, that usually is the Sigmoid function:

$$y = \frac{1}{1 + e^{-net_j}} \tag{32}$$

Then, the Sigmoid function's output is passed to the nodes of the next level, and the procedure is repeated until the activation function for the output node is computed. The obtained value is the output of the learning process, i.e., the predicted value for the target variable of the first sample. It is compared to the actual value, and the difference between them, called error, is estimated. Neural networks are, in most cases, trained by using the back-propagation algorithm. It takes the prediction error for a particular observation and filters it back to the network, assigning a portion of responsibility for that error, δ_j, to each node. Weights are adjusted in order to minimize the sum of squared errors (SSE). The new value of the weight is given by adding to the current weight value the amplitude of its adjustment, equal to:

$$\Delta w_{i,j} = x_{i,j} \delta_j \eta \tag{33}$$

where

$$\delta_j = \begin{cases} output_j(1 - output_j)(actual_j - outpt_j) & if\ j\ is\ an\ output\ layer\ node \\ output_j(1 - output_j) \sum\limits_{downstream} W_{jk} \delta_j & if\ j\ is\ an\ hidden\ layers\ node \end{cases} \tag{34}$$

and $\sum_{downstream} W_{jk} \delta_j$ is the weighted sum of error responsibilities for nodes downstream to node j. η is a constant ranging between 0 and 1, called learning rate: if small, the adjustment tends to be small, and it would take more time before letting the new weight converge to the optimal value; on the other hand, if η is too large, an oscillation around the optimal value may occur. The trade-off between convergence time and accuracy has to be found. The back-propagation algorithm terminates when: (1) the number of iterations or the total amount of time consumed by the algorithm is reached;

or (2) a threshold level for the SSE on training data is set, considering that short training times may reduce the model efficacy and that ANNs are prone to overfitting. ANNs require a small number of parameters to optimize for training but may require more computational resources, and there is no standard to determine the network structure.

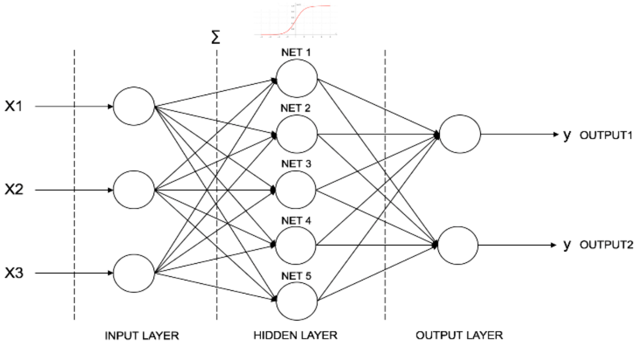


Figure 47 Feedforward neural network

Classification models can be evaluated by using different criteria. The most common criteria is accuracy, which is a measure of the ability of the model to predict the target class for future observations and is given by Eq. (35)

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \tag{35}$$

Where:

- TP (True Positive) is the number of positive observations correctly classified
- TN (True Negative) is the number of negative observations correctly classified
- FP (False Positive) is the number of positive observations classified as negative
- FN (False Negative) is the number of negative observations classified as positive

When the distribution of the class is not balanced, i.e., the dataset contains much more majority class than minority class instances, the accuracy is not a sufficient measure for the model evaluation, as long as one can always select the majority class and obtain good accuracy performance. Thus, other measures, like sensitivity, specificity, and precision, can be used. Sensitivity, also called recall or true positive rate, measures how well the classifier recognizes positive samples. It is equal to the ratio of the number of true positive samples (correct predictions for the positive examples) to the number of positive samples (Eq. 36). Precision measures how many samples classified as a positive class are indeed positive, and it is equal to the ratio of the number of true positive samples to the sum of the number of true positive samples and the number of false-positive samples (Eq. 37). Specificity, also named true negative rate and defined by Eq. (38), measures how well the classifier recognizes negative samples and is equal to the ratio between the number of true negative (correct predictions for the negative examples) to the number of negative samples.

$$Recall = \frac{TP}{FN+TP} \tag{36}$$

$$Precision = \frac{TP}{FP+TP} \tag{37}$$

$$Specificity = \frac{TN}{TN+FP} \tag{38}$$

Confusion matrices for binary classifications are 2×2 matrices whose rows correspond to the observed values and whose columns correspond to values predicted by the classifier. As we can see in Table 18, the main diagonal of the confusion matrix contains the number of elements that have been correctly classified for each class, while the off-diagonal contains the number of observations that have been incorrectly classified. The confusion matrix is built comparing the actual class of every instance in the test set to the class assigned by the trained classifier.

Table 18 Confusion matrix

	Predicted negative	**Predicted positive**
Negative examples	True negative (TN)	False positive (FP)
Positive examples	False negative (FN)	True positive (TP)

To visually evaluate the accuracy of a classier, *Receiver Operating Characteristic* (ROC) curve charts are available. Using a two-dimensional plot, they illustrate the trade-off between the number of correctly classified positive observations and the number of incorrectly classified negative observations. The x-axis represents a false positive rate, and the *y*-axis represents a true positive rate. To build the trajectory of the ROC curve for a specific classifier, it is necessary to have pairs of values (false positives, true positives) that have been obtained for different parameters of the learning function. In each ROC curve, we can individuate three characteristic points: the point (0,0), which represents a negative prediction for all the observations; the point (1,1), which represents a positive prediction for the observations; the point (0,1), that indicates that no prediction error has been made and therefore is the ideal condition. In addition, ROC curves can also be used to compare different classifiers and to find the optimal one. A classifier is preferred when its ROC curve has a greater area than others' ones (AUC); the classifier at a certain point of the ROC convex hull is optimal if a line passes through that point since no other line with the same slope passing through another point with a larger TP intercept exists. ROC curves are typically used in binary classification.

4.1.2. Unsupervised learning for fault detection

Unsupervised learning is mainly performed by clustering, in which a set of unlabelled data is grouped into clusters in such a way that objects belonging to the same clusters are similar between each other and dissimilar to objects belonging to other clusters. A typical clustering analysis involves the following activities (Flynn 1999): pattern representation, the definition of the pattern proximity measure, clustering, data abstraction, and assessment of the output.

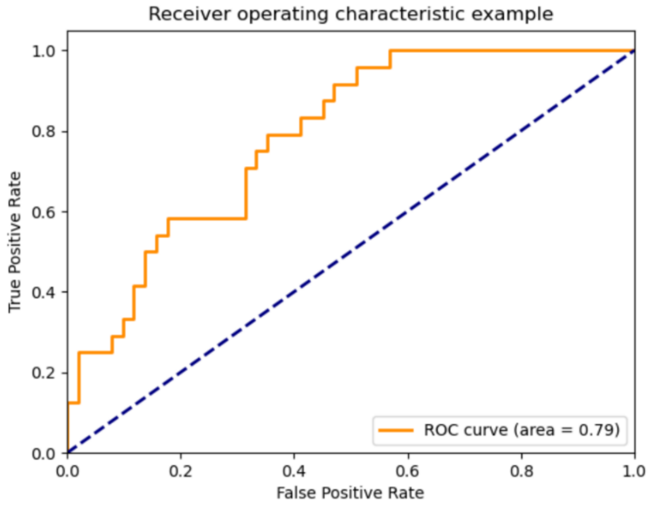


Figure 48 ROC curve

The first distinction of numerous clustering techniques can be made between hard and fuzzy clustering. In hard clustering, the membership of elements in a cluster is assessed in binary terms, meaning that an element either belongs to a cluster or not. On the contrary, the membership of elements in a fuzzy cluster is described by a membership function valued in [0,1], meaning that elements can belong to more clusters to a certain degree. According to (Warren Liao 2005), clustering algorithms can be grouped into five categories: hierarchical methods, partitioning methods, density-based methods, grid-based methods, and model-based methods. A hierarchical algorithm groups patterns into hierarchical structures, like dendrograms, showing how patterns are grouped and the similarity levels of each grouping. Hierarchical algorithms can adopt either a divisive or agglomerative strategy. In the first case, the whole dataset is considered a cluster and then divided into smaller clusters depending on the data structure; at the opposite, the agglomerative strategy considers a certain number of initial clusters and then merges clusters at each level according to their similarity. Hierarchical algorithms have several drawbacks: first, they are not able to adapt themselves once a merge/split decision have been made (Warren Liao 2005); second, they are computationally expensive, and there are no standards for the identification of the stopping criterion; finally, information about relationships between cluster is lost (Datta, Mavroidis, and Hosek 2007). Partitioning methods construct k partitions of the data by minimizing a specific objective function, normally chosen to be the squared error. The most popular partitioning algorithms perform an initial partition of the data set into k clusters, with randomly selected k centroids; then, they assign each feature vector to its closest centre and update the centroid of the clusters; the procedure is repeated until a convergence criterion is met. Partitioning methods are simple and rapidly converge when the dimensionality of the input data is small. However, they only work well for finding spherical-shaped clusters and small and balanced data sets; the number of the desired output clusters, k, have to be known in advance; the initial random partition can lead to stack in a local optimum. Density-based methods (Ester et al. 1996) and grid-based (Xiangli Li, Han, and Qiu 2017) techniques expand clusters from core points/dense grids to boundary points/grids. The main difference lies in the definition of clustering/grid boundaries, which are based on density in density-based methods, while they are determined by statistics of points within a single grid. These methods are both able to form clusters of arbitrary shape and do not require to specify the number of clusters.

Clustering methods are often used for detecting anomalies corresponding to the occurrence of a failure. An anomaly can be defined as a point or a set of points, which differ from the most common behavior. In general, anomaly detection methods learn, during the training, a representation of the normal operation of the machinery and are used at serving time to identify deviations from this representation. During the test phase, they are therefore required to assign novelty scores to each test data and then categorize them as belonging to a new operating condition depending on whether the score exceeds a certain threshold or not. Anomaly detection techniques can adopt a supervised, unsupervised and semi-supervised learning paradigm, according to the availability of labeled data. In particular, unsupervised anomaly detection algorithms can be categorized in nearest neighbor-based techniques, clustering-based methods, and statistical algorithms (Goldstein and Uchida 2016).

The most common ML methods adopted for novelty detection are summarized in this section. The Local Outlier Factor (LOF) algorithm is an unsupervised anomaly detection method that computes the local density deviation of a given data point to its neighbors. It considers as outliers the samples that have a substantially lower density than their neighbors (Breunig et al. 2000). The Isolation Forest ‘isolates’ observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature. Since a tree structure can represent recursive partitioning, the number of splits required to isolate a sample is equivalent to the path length from the root node to the terminating node. This path length averaged over a forest of such random trees measures normality and our decision function. Random partitioning produces noticeably shorter paths for anomalies. Hence, when a forest of random trees collectively produces shorter path lengths for particular samples, they are highly likely to be anomalies (W. Xu et al. 2010). In our case, the One-Class Support Vector Machine is an unsupervised learning algorithm trained only on the nominal data, the negative examples. It learns the boundaries of these points and can, therefore, classify any points that lie outside the boundary as outliers (Scholkopf et al., 2001). The Principal Component Analysis (PCA) is frequently used in exploratory data analysis because it reveals the inner structure of the data and explains the variance in the data. PCA looks for correlations among the variables and determines the combination of values that best captures differences in outcomes. For anomaly detection, each new input is analyzed, and the anomaly detection algorithm computes its projection on the eigenvectors, together with a normalized reconstruction error. The normalized error is used as the anomaly score. The higher the error, the more anomalous the instance is (F. T. Liu, Ting, and Zhou 2012). Online clustering can be considered a distance-based anomaly detection approach, in which the "normal" class is characterized by a small number of prototype points in the data space. During the prediction step, the distance between the "normal" points and new points is computed. A threshold is set to determine whether the current pattern belongs to the same cluster as the normal one, or creates a new cluster. Among DL approaches, the most widely adopted method for anomaly detection problems are autoencoders. They are neural architectures that compress the input data into a compact vector representation (encoding phase) and try to reconstruct the original data starting from this intermediate representation (decoding phase) (Xiang Li, Li, and Ma 2020). In the context of anomaly detection, these architectures identify new operating conditions when the reconstruction error obtained exceeds a certain threshold, which confirms that the processed input cannot refer to any normal condition encountered in the training phase. This generic architecture can be implemented using different types of neural networks: simple Feed-Forward neural networks, Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs). Feed-Forward AutoEncoder relies on Multilayer Perceptrons, or MLPs for short, to encode and decode the input data and intermediate representations, respectively. CNN AutoEncoder applies convolutive filters to an input organized in a grid to derive an intermediate representation that encodes the spatial proximity information of the original data (encoding phase) and adopts an inverse strategy to re-expand this intermediate knowledge. RNN AutoEncoder (LSTM in our case) is based on a recurrent connection of hidden representations generated from multiple MLPs, which is exploited to compress and reconstruct data while preserving their sequentiality and order of occurrence.

4.1.3. Incremental learning for novelty detection

In the context of incremental learning, anomaly detection, clustering, and classification algorithms dealing with streaming data are integrated. These methods should have the following characteristics (Ahmad et al. 2017): (1) they have to identify the state of the current data point before the next data point is available; (2) they have to learn continuously without storing the entire data stream; (3) have to adopt an unsupervised approach; (4) have to deal with the concept drift problem, that means to have the ability to adapt model structures if a data point cannot be considered a true anomaly; (5) have to identify an anomaly as early as possible; (6) they should minimize false positives and false negatives. Recently, two novel online data-driven anomaly detection approaches have been proposed, namely ODDAD (Khalastchi et al. 2015) and TEDA (Bezerra et al. 2016). ODDAD algorithm, for each new input, looks at the latest *m* samples of input (sliding window) and quickly decides whether or not this online data present a fault by using the Mahalanobis Distance calculation. Above a calculated threshold, the algorithm declares an anomaly. TEDA (Typicality and Eccentricity Data Analytics) algorithm belongs to the one-class classification models, which assume that the label of nominal data is provided. It is based on the concepts of typicality, related to the similarity of a particular data sample to an entire data set in the sense of spatial proximity and eccentricity, which reflects how distinct is a data sample from the data group. This algorithm is entirely unsupervised and does not rely on any assumption on data distribution; it is recursive, fast, and has low computational complexity. Typicality and eccentricity are calculated and updated as new data points arrive based on a distance measure. However, it requires the definition of a certain threshold to assign the label nominal or faulty to a data point.

Angelov, Ramezani, & Zhou (2008) introduced the concept of Recursive Density Estimation (RDE) in the context of detection and object tracking in video streams. Aiming to decide whether a pixel belongs to the background or the foreground in real-time, the introduced approach substitutes the standard Gaussian kernel adopted for modeling the pixel probability density function (pdf) with the Cauchy function, which allows the recursive estimation of pixel pdf as a new image frame occurs. In Costa et al. (2015), concepts of RDE theory are adopted in the context of online fault detection in order to discover anomalous behaviors. The parameters involved are the global density, the mean value, and the scalar product, which can be recursively computed by Eq. (39), Eq. (40), and Eq. (41), respectively.

$$\mathrm{D}(\mathbf{x}_\mathrm{k}) = \frac{1}{1+\|\mathbf{x}_\mathrm{k}-\boldsymbol{\mu}_\mathrm{k}\|^2+\Sigma_\mathrm{k}-\|\boldsymbol{\mu}_\mathrm{k}\|^2} \tag{39}$$

$$\boldsymbol{\mu}_\mathrm{k} = \frac{\mathrm{k}-1}{\mathrm{k}}\boldsymbol{\mu}_\mathrm{k} + \frac{1}{\mathrm{k}}\mathbf{x}_\mathrm{k} \tag{40}$$

$$\Sigma_k = \frac{k-1}{k}\Sigma_k + \frac{1}{k}\|x_k\|^2 \tag{41}$$

where $\boldsymbol{x}_k \in \mathbb{R}^n$ is the feature vector at the time stamp k.

At the first iteration ($k = 1$), the parameters are initialized as $D(\boldsymbol{x}_1) = 1, \boldsymbol{\mu}_1 = \boldsymbol{x}_1, \Sigma_1 = \|\boldsymbol{x}_1\|^2$. The, for each $k > 1$, the parameters are updated, and the condition in Eq. (42) is checked to decide whether the current point represents an anomaly or not

$$\begin{aligned} &IF\ D(\boldsymbol{x}_k) < \mu_D \\ &for\ k = t_1, \dots, k-1, k \\ &THEN\ \boldsymbol{x}_k\ is\ an\ anomaly \end{aligned} \tag{42}$$

Where

$$\mu_D = \left(\frac{ks-1}{ks}\mu_D + \frac{1}{ks}D(\boldsymbol{x}_k)\right)(1 - \mathit{\Delta}_D) + D(\boldsymbol{x}_k)\mathit{\Delta}_D \tag{43}$$

is the mean value of the local density computed recursively, $\Delta_D = |D(\boldsymbol{x}_k) - D(\boldsymbol{x}_{k-1})|$ is the absolute value of the difference between the global density computed at two consecutive time stamps, and ks is the number of data samples from the last status changes. Indeed, if the condition is satisfied, then the status of the system switches from normal to anomalous, and ks is set to 0. The condition expressed in Eq. (42) means that if the global density is lower than the mean density for a certain number of time stamps (or seconds), the status becomes anomalous. Indeed, when a new data point arrives if it is close to the previous point, then $\boldsymbol{\mu}_k$ is close to the $\boldsymbol{x}_k$, $D(\boldsymbol{x}_k)$ stays close to 1 and μ_D stays close to the actual mean of the data points, as $(1 - \Delta_D)$ is very close to 1, which gives more importance to the first term of Eq. (43). However, when the new point is far from the previous ones, $D(\boldsymbol{x}_k)$ slightly decreases, while μ_D becomes closer to $D(\boldsymbol{x}_k)$, as the term Δ_D gives more importance to the second term of Eq. (43). As new points are closer to the previous point, then $D(\boldsymbol{x}_k)$ continues to decrease until the condition in the Eq. (42) is satisfied. Note that, when the status changes from normal to anomalous, ks is set to 0, leading μ_D to notably decrease. When the mean density is lower than the global density for a certain number of points, or seconds, the status returns normal. Thus, the condition expressed by Eq. (44) applies:

$$\begin{aligned} &IF\ D(\boldsymbol{x}_k) > \mu_D \\ &for\ k = t_2, \dots, k-1, k \\ &THEN\ \boldsymbol{x}_k\ is\ normal \end{aligned} \tag{44}$$

Note that, both t_1 in Eq. (42) and t_2 in Eq. (44) are set by the user and may represent either the data samples or seconds.

Based on the concepts of RDE, a clustering algorithm has also been introduced in Gu et al. (2018), in both offline and online versions. Here, the online version will be briefly described. At the first iteration ($k = 1$) the local parameters of each cluster are initialized as follows

$$C_k = 1;\ \boldsymbol{\mu}_k^1 = \boldsymbol{x}_1;\ S_k^1 = 1 \tag{45}$$

Where, C_K is the cluster at the time stamp $k = 1$, $\boldsymbol{\mu}_1^1$ is the focal point of cluster 1 at the time stamp $k = 1$, and S_1^1 is the number of data points belonging to the cluster 1 at the time stamp $k = 1$. In addition, the parameters expressed by Eq. (40), Eq. (41) are also initialized. Then, for each $k > 1$,

1. The mean value $\boldsymbol{\mu}_k$ and the scalar product Σ_k are updated by means of Eq. (40) and (41), respectively.
2. The condition expressed by Eq. (46) is checked to decide whether the current point has to be assigned to an existing cluster or should be a new focal point itself

   $$\begin{gathered} IF\ D(\boldsymbol{x}_k) > \max_{i=1,\ldots,C_k} D_k(\boldsymbol{\mu}_k^i)\ OR\ D(\boldsymbol{x}_k) < \min_{i=1,\ldots,C_k} D_k(\boldsymbol{\mu}_k^i) \\ THEN\ \boldsymbol{x}_k\ becomes\ a\ new\ focal\ point \end{gathered} \tag{46}$$

   The condition expressed in Eq. (46) means that if the global density is higher than, or lower than, the densities computed at each of the existing focal points (the density of each cluster), then the current point creates a new cluster.
3. If Eq. (46) is satisfied, then a new cluster is created, whose parameters are initialized by means of Eq. (47)

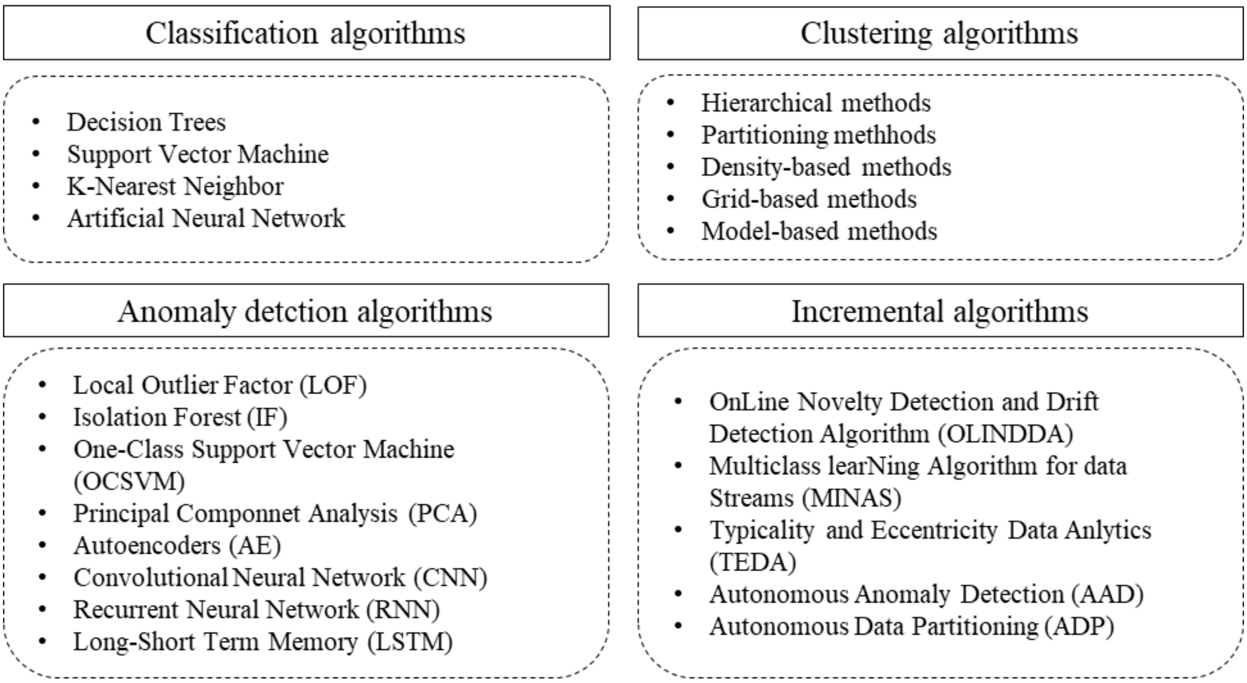
   $$C_k = C_{k+1};\ \boldsymbol{\mu}_k^{C_k} = \boldsymbol{x}_k;\ S_k^{C_k} = 1 \tag{47}$$
4. Otherwise, the distance between the current feature vector $\boldsymbol{x}_k$ and the focal point of each existing cluster is computed in order to decide whether the current point can be assigned to the closest cluster, using Eq. (48)

   $$\begin{gathered} IF\ \|\boldsymbol{x}_k - \boldsymbol{\mu}_k^n\| < \sqrt{\Sigma_k - \|\boldsymbol{\mu}_k\|^2} \\ THEN\ \boldsymbol{x}_k\ is\ assigned\ to\ \boldsymbol{\mu}_k^n \end{gathered} \tag{48}$$

   The condition expressed by Eq. (48) means that if the distance between the current point and the nearest focal point $\boldsymbol{\mu}_k^n$ is lower than the distance among all points arrived until the current timestamp, the current point is assigned to the nearest cluster.
5. If Eq. (48) is satisfied, then the parameters of the closest cluster C_k^{n} are updated through Eq. (49)

$$\boldsymbol{\mu}_k^n = \frac{S_k^n - 1}{S_k^n}\boldsymbol{\mu}_k^n + \frac{1}{S_k^n}\boldsymbol{x}_k;\ S_k^n = S_k^n + 1 \tag{49}$$

Summing up, this section has provided a classification of the novelty detection approaches, and a brief description of the methods usually adopted for fault diagnosis and fault detection in evolving environments, which are depicted in Figure 49.



Classification algorithms

- Decision Trees
- Support Vector Machine
- K-Nearest Neighbor
- Artificial Neural Network

Clustering algorithms

- Hierarchical methods
- Partitioning methhods
- Density-based methods
- Grid-based methods
- Model-based methods

Anomaly detction algorithms

- Local Outlier Factor (LOF)
- Isolation Forest (IF)
- One-Class Support Vector Machine (OCSVM)
- Principal Componnet Analysis (PCA)
- Autoencoders (AE)
- Convolutional Neural Network (CNN)
- Recurrent Neural Network (RNN)
- Long-Short Term Memory (LSTM)

Incremental algorithms

- OnLine Novelty Detection and Drift Detection Algorithm (OLINDDA)
- Multiclass learNing Algorithm for data Streams (MINAS)
- Typicality and Eccentricity Data Anlytics (TEDA)
- Autonomous Anomaly Detection (AAD)
- Autonomous Data Partitioning (ADP)

Figure 49 Algorithms for fault diagnosis, fault detection and novelty detection

In the following sections, the case studies to which these techniques will be applied are presented. In particular, section 4.2 describes the cases for which offline and batch analysis for fault diagnosis and setting recognition is conducted; section 4.3 describes the cases for which novelty detection in evolving environments is performed. Details about the adopted methods will be provided in each section.

4.2. Offline and batch analysis

This section will describe the results obtained from applying several classification methods to the feature sets extracted in the previous chapter. Classification is conducted for both operating condition recognition and fault diagnosis. Also, the goodness of the features extracted through data processing techniques is evaluated in terms of provided classification accuracy. To these purposes, the *Classification learner* and *Pattern recognition* toolboxes provided by MATLAB have been used for model building, during which the 5-fold cross-validation and the hyper-parameter optimization have been performed. In particular, the datasets have been divided into training and testing sets. The training sets are used during the model building step. Then, the so-built models are applied to the test set, and the predicted class is compared with the actual class for computing the accuracy on the test and evaluating the generalization ability of each model. Besides the classification accuracy, computed by Eq. (35) on both training and testing sets, the training time, which represents the time needed for model building, is also used to compare the performance of the classification models. The goals for each case study are summarized in Table 19, where applied feature selection and construction models are also summarized. The sets of features extracted in the time or time-frequency domain, the feature sets selected manually or through Pearson Correlation analysis, and the feature sets constructed through Genetic Programming and PCA are described sections 3.3.1, 3.3.2, and 3.3.3, respectively.

Table 19 Case studies goals for offline and batch analysis

Case study	Goal	Feature selection/construction method
Test rig	Operating Condition recognition	Experiment 1. Pearson Correlation analysis and manual feature selection
		Experiment 2. Genetic Programming
Sealing group (displacement)	Fault diagnosis	Pearson Correlation analysis + Recursive Feature extraction
Suction cups	Fault diagnosis	Pearson Correlation analysis, Principal component Analysis
Case Packer	Fault diagnosis	Pearson Correlation analysis, Principal component Analysis
Sealing group (vibrations)	Fault diagnosis	-
Automatic machinery	Operating Condition recognition	Principal component Analysis

The results are grouped according to the specific task, i.e., operating condition recognition and fault diagnosis. In addition, the effect of feature selection or construction on classification accuracy is also assessed for each case study.

4.2.1. Operating condition recognition

Case 1. Test rig: experiment 1. For this case study, three feature sets are considered, i.e., the statistical features extracted in the time domain, the selected feature set obtained after the application of Pearson Correlation analysis, and the feature set obtained by manually selecting the relevant features. Each feature set is used as input of four classification models, i.e., Decision Tree, K-Nearest Neighbor, Artificial Neural Network, and Support Vector Machine. The whole dataset has been divided into the training set, including the 70% of the samples (i.e., 15,554 observations, equal to 10 MB), and the testing set, including the remaining observations. Results of all cases are summarized in Table 20. It can be seen that the feature set providing the best results are the set of all time-features and the set of features selected through the Pearson Correlation analysis. In all cases, both training and testing accuracy are greater than 99.5%, while the training accuracy achieves the 100% in the case of KNN and SVM. However, both these feature sets require high training times, especially in the case of SVM. On the contrary, the feature set manually selected provides slightly worse results, especially in the case of DT, which requires more time for the training and is less generalizable. However, when KNN and SVM are used for condition classification, the training time are much lower, while the training and testing accuracies are lower of less than 1%.

Table 20 Test rig. Experiment 1: Features evaluation for operating condition recognition

Model	Features extracted in the time-domain			Features selected through Pearson Correlation analysis			Manually selected features		
	Training Time (s)	Training Accuracy (%)	Testing Accuracy (%)	Training Time (s)	Training Accuracy (%)	Testing Accuracy (%)	Training Time (s)	Training Accuracy (%)	Testing Accuracy (%)
DT	52.3	99.9	99.5	41.2	99.9	99.6	77.3	98.7	97,6
KNN	1,650.2	100	99.8	1,054.2	100	99.8	380.9	99.0	98.0
SVM	6290	100	99.8	10,496	100	99.8	1274.6	99.1	98.1

Case 1. Test rig: experiment 2. Consider the features constructed through the classification-based GP (Figure 35) and the clustering-based GP (Figure 36). These two features have been input, separately, to two classification models, i.e., Decision Tree and K-Nearest Neighbor, and the performances are summarized in Table 21. The whole dataset has been divided into training and testing sets. The training set consists of two run-to-failure trajectories for each of the two distinct settings (F1 – F2 for setting 5, and F3 – F4 for setting 6), the test set consists of one run-to-failure trajectory (F5), obtained under settings 5 and 6, put in sequence without stopping the system. The training accuracy and the testing accuracy of the cluster-based feature are higher than those provided by the classification-based feature. In addition, the training time in the clustering case is lower. In this case, the Decision Tree provides better results in terms of training accuracy. However, both classification-based and clustering-based features provide low accuracy when applying both models to the failure trajectory F5. Hence, both models applied to both the feature sets have not high generalizability.

Table 21 Test rig: experiment 2. Classification performance with GP-based feature construction

	Classification			Clustering		
Model	Training Accuracy (%)	Training Time (sec)	Testing Accuracy (%)	Training Accuracy (%)	Training Time (sec)	Testing Accuracy (%)
DT	90,4	2,6	67,03	96	0,71	67,30
KNN	90,8	1,25	67,03	94,5	0,72	66,85

The two classification models have also been applied to the whole set of extracted features (before GP) to evaluate the effect of feature construction on the classification accuracy. Finally, to evaluate the performance of the GP-based feature construction, other methods have been applied to the extracted features to construct or select a more relevant feature set. In particular, the PCA has been chosen for the comparison. Using PCA before classification, 4 PCs are selected, which explain 92% of the variance. Table 22 shows the results of this analysis. In both cases, the training accuracy is higher than 95%. However, the resulting models are less generalizable, and the training time is higher, especially in the case of KNN. In this case, the training accuracy is higher than GP-based feature construction. However, the testing accuracy is lower, meaning that models are less generalizable using PCs.

Table 22 Test rig: Training classification performance without feature selection and construction and with PCA

	Without Feature selection/construction			With PCA		
Model	Training Accuracy (%)	Training Time (sec)	Testing Accuracy (%)	Training Accuracy (%)	Training Time (sec)	Testing Accuracy (%)
DT	99,99	3,53	31,52	95,2	1,96	64,46
KNN	100	56,94	61,14	95,42	0,89	61,50

Finally, the performance of the extracted features for classification and cluster are also evaluated in relation to the ability to distinguish unknown conditions. To this aim, a run-to-failure trajectory of each "known" condition and a trajectory corresponding to a "novel" operating condition (F6) are used for training classification models. In Figure 50, the constructed features with the clustering GP-based (left) and the classification GP-based (right) are shown, where distinct colors represent distinct operating conditions. The results of classification models trained on these datasets are summarized in Table 23. It can be seen that when the system-level feature is constructed with the classification-based GP, classification models trained with the additional operating condition provide a class prediction

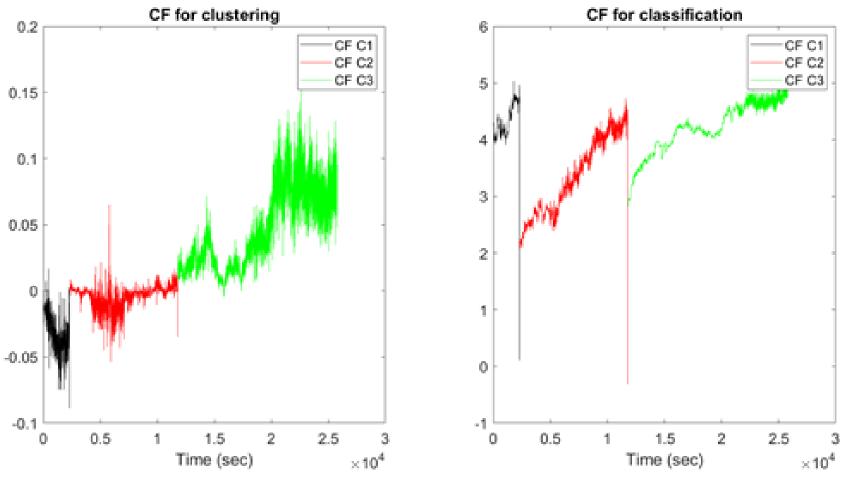


Figure 50 Constructed features through clustering-based GP and classification-based GP with an additional operating condition

accuracy lower than 75%. On the contrary, the classification accuracy is greater than 90% when using the clustering-based GP.

Table 23 Classification accuracy (%) with GP-based FCs for classification and clustering with additional operating condition

Model	Classification	Clustering
DT	72,9	93,8
KNN	63,4	91,3

Case 2. Automatic machinery. For the automatic machinery, the application of PCA provided 4 PCs for unit 1 and 5 PCs for unit 2. To evaluate its effectiveness, two classification models, i.e., Support Vector Machine (SVM) and K-Nearest Neighbor (K-NN), have been applied to both the original matrix ***X*** (without PCA) and the matrix of extracted PCs. Results for both the units are shown in Table 24. Both models, in both cases, provide 100% of accuracy in classifying the different operating conditions.

Table 24 Extruder: classification results with and without PCA

	Unit 1		Unit 2	
Model	Without PCA	With PCA	Without PCA	With PCA
SVM	100%	100%	100%	100%
K-NN	100%	100%	100%	100%

4.2.2. Fault diagnosis

Case 1. Test rig. Experiment 1. In this case, the goal is to detect the sudden fault that occurred to the electric motor. Since both the Pearson Correlation analysis and the manual selection do not take into account the classification accuracy to select the best feature set,_the input features are the same as section 4.2.1. However, only two classes are considered in this case. The first one contains the three nominal settings, and the second contains observations belonging to the fourth setting, during which a sudden fault to the electric motor occurred. The dataset has been divided as for the case of condition recognition (70% of observation for training and 30% for testing). Results are summarized in Table 25. In all cases and for all models, both training and testing accuracies are greater than 99.0%, which demonstrates that all models have good generalization ability. Training times are particularly high for the SVM, while the DT is the fastest model. Concerning the feature sets, the feature manually selected require less training time for all models, and the accuracies can be considered similar to those obtained with the other feature sets.

Table 25 Test rig. Experiment 1: Features evaluation for fault diagnosis

Model	Features extracted in the time-domain	Features selected through Pearson Correlation analysis	Manually selected features

	Training Time (s)	Training Accuracy (%)	Testing Accuracy (%)	Training Time (s)	Training Accuracy (%)	Testing Accuracy (%)	Training Time (s)	Training Accuracy (%)	Testing Accuracy (%)
DT	33.7	100	99.8	80.2	100	99.8	27.3	99.1	99.4
KNN	486.2	100	99.9	279.6	100	99.9	230.4	99.2	99.6
SVM	3866	100	99.7	3,194.8	100	99.9	1,133	99.1	99.7

Case 2. Sealing group (displacement). In this case, the aim is to build models able to classify four different conditions, one nominal and three fault conditions. Two feature sets are considered, i.e., time-domain features without feature selection/construction and selected features through Pearson Correlation analysis and Recursive Feature Elimination. Each feature set has been divided into a training set, including the 70% of observations, and a testing set, including the remaining samples. The original features extracted in the time domain provide better results in terms of training accuracy. However, the testing accuracy is much lower. On the contrary, the reduced feature set provides lower training accuracy, but the built models are more generalizable. The training time of models is similar in the two cases, except the SVM, which requires more time in case of the reduced feature set.

Table 26 Sealing group (displacement): Features evaluation for fault diagnosis

Model	Features extracted in the time-domain			Features selected through Pearson Correlation analysis + Recursive Feature Elimination		
	Training Time (s)	Training Accuracy (%)	Testing Accuracy (%)	Training Time (s)	Training Accuracy (%)	Testing Accuracy (%)
DT	23.37	91.3	72.71	26.1	86.4	85.6
KNN	34.33	89.4	72.24	40.6	86.4	82.1
SVM	78.32	90.4	71.76	363.6	86.7	82.3

Case 3. Suction cups. Given the few observations available, the whole dataset has been used for the training in this case, ,. Three sets of features are considered, i.e., the set of statistical time-features, the set obtained after Pearson Correlation analysis, and the set of extracted PCs after PCA. From Table 27, it can be seen that PCA provides the best training time and training accuracy for all models, while the feature set obtained after feature selection provide the worst results.

Table 27 Suction cups: Features evaluation for fault diagnosis

	Features extracted in the time-domain		Features selected through Pearson Correlation analysis		Features extracted through PCA	
	Training Time (s)	Training Accuracy (%)	Training Time (s)	Training Accuracy (%)	Training Time (s)	Training Accuracy (%)
DT	20.4	97.4	23.7	96.6	27.7	100
KNN	30.0	97.0	32.5	97.4	31.3	100
ANN	0.1	98.1	0.1	97.5	0.1	100
SVM	58.8	97.8	56.8	97.8	71.8	100

Case 4. Case packer group. Like the previous case, three sets of features are considered. Results are shown in Table 28. In this case, the time-domain features, without feature selection, provide the best results for all models. Except for ANN, both the reduced feature sets provide similar results. However, in the case of the features extracted through the PCA, the Decision Tree has low generalizability ability.

Table 28 Case packer group: features evaluation for fault diagnosis

Model	Features extracted in the time-domain			Features selected through Pearson Correlation analysis			Features extracted through PCA		
	Training Time (s)	Training Accuracy (%)	Testing Accuracy (%)	Training Time (s)	Training Accuracy (%)	Testing Accuracy (%)	Training Time (s)	Training Accuracy (%)	Testing Accuracy (%)
DT	22.34	99.0	83.7	19.55	97.4	85.1	24.33	96.2	76.3
KNN	35.15	99.7	88.15	27.38	98.7	86.6	31.72	98.8	85.1
ANN	0.1	98.4	97.0	0.1	89.1	85.0	0.1	97.0	97.0
SVM	124.28	99.4	85.93	185.42	98.1	82.9	158.06	96.8	86.6

Case 5. Sealing Group (Vibrations). The features extracted from raw vibration signals in the time and frequency domains have been divided into training and test sets. The training set is built by using the first run of each product, while the test sets contain the other two runs. Classification models applied in this case are the Decision Tree, the K-Nearest Neighbor, the Artificial Neural Network, and Support Vector Machine. The selected splitting rule in the DT training is the Gini-index, while the Euclidean distance is chosen as a similarity value of the k-NN. Also, the k-NN has been trained with different values of k to find the best value of the nearest neighbors. $k = 2$ provided the best training accuracy for both products. A Feed Forward Neural Network is trained using the backpropagation algorithm, with the minimum gradient (10E-10) and the maximum number of iterations (500) set as termination criteria. After several iterations, with different numbers of hidden layers and neurons in each layer, a neural network made of 1 hidden layer with six nodes has been chosen since it provided the best training accuracy. Finally, the Support Vector Machine is trained with a linear kernel function. As shown in Table 29, which summarizes accuracy values on the training set and the testing sets using time-domain features, the results of different models are similar for product 1. For product 2, the DT and the K-NN algorithm give higher accuracy than other models. However, the DT and the ANN give lower testing accuracy for both products. Therefore, they are less generalizable than other models. On the contrary, the SVM has the best trade-off between accuracy and generalization for both products.

Table 29 Sealing group (vibrations): accuracy of classification models using time-domain features

Model	Product 1			Product 2		
	Training	Testing 2	Testing 3	Training	Testing 2	Testing 3
DT	95,71	72, 65	94,06	95,71	75,93	73,15
K-NN	95,71	93,36	97,90	90,00	74,53	72,34
ANN	95,71	77,89	86,55	81,43	60,31	56,87
SVM	97,90	98,43	97,60	82,50	83,74	85,30

Results of both training and testing using time-frequency-domain features are summarized in Table 30. For product 1, models provide similar accuracy values, and no model seems to over-fit data. For product 2, the same consideration can be done.

Table 30 Sealing group (vibrations): accuracy of classification models using time-frequency-domain features

Model	Product 1			Product 2		
	Training	Testing 2	Testing 3	Training	Testing 2	Testing 3
DT	95,71	97,60	98,04	94,29	90,17	91,79
K-NN	97,14	97,65	97,65	91,43	91,86	91,79
ANN	95,71	96,87	98,04	95,71	90,66	90,93
SVM	98,30	97,65	97,65	88,20	93,51	92,88

In general, results obtained in the time-frequency domain are better than those in the time domain, especially considering the generalization ability of the models. In addition, both training and testing accuracy values of product 2 are lower than product 1. The reason could be the presence of anomalous peaks in the vibration signature of product 2. In particular, the z-axis seems to give some problems, which are not related to the health condition of the component. Hence, further analysis is conducted considering the value of the acceleration without the component along the z-axis. Results are shown in Table 31. Results provided by the decision tree are slightly better in the time domain but much worse in the time-frequency domain. For k-NN, the situation is the opposite. Instead, results obtained with ANN are similar in the time-frequency domain but much better in the time domain, especially from the generalization point of view. SVM provides similar or slightly better results.

Table 31 Sealing group (vibrations): accuracy of classification models using time-frequency-domain features and signal pre-processing

Model	Time-domain			Time-frequency-domain		
	Training	Testing 2	Testing 3	Training	Testing 2	Testing 3
DT	98,57	79,99	74,76	92,86	50,78	51,64
K-NN	91,43	62,65	63,43	94,29	91,79	92,18
ANN	82,86	90,23	87,81	95,71	89,84	90,54
SVM	82,10	86,71	84,99	89,90	96,32	92,57

To conclude, in this case, all models work better when signals are transformed into the time-frequency domain. In this particular case, wavelet transform is used to reduce the noise and the effect of anomalous peaks. The second experiment on product 2 highlights that the manual cleaning of signals in this domain does not improve the accuracy. However, it would be preferable for some models, particularly ANNs, in the time-domain analysis, in which the distribution of the acceleration values influences the classification results.

4.3. Online and streaming analysis

In this section, novelty detection in evolving environments is performed on the case studies. As for the offline and batch analysis, the ability of algorithms to detect a change in the system operating condition and to assess the component's health condition are evaluated. In particular, according to the classification of novelty detection approaches in evolving environments provided at the beginning of this chapter, two kinds of analysis are conducted. In the first case, a clustering-based approach is used, which can start from scratch or include a training phase. In this case, the performance of the algorithm are evaluated in terms of latency, i.e., the time difference between the moment in which the algorithm detects a change in the behavior of the component or system and the moment in which actually the change or fault occurs, and the number of created clusters, which expresses the ability of the algorithm to recognize novel behaviors. In addition, the number of false alarms, i.e., the number of anomalies that do not determine a change of the cluster to which the points are assigned, is also considered. This metric has to be intended as the number of false alarms that are avoided by applying the AAD and the ADP in sequence. The second kind of analysis uses a classification-based approach, which may be incremental or non-incremental. In this cases, the metrics adopted are the precision, recall, the training accuracy, the training time and the execution time.

The case studies will be presented in order of increasing complexity. First, the Autonomous Anomaly Detection (AAD) algorithm is applied to current signals extracted from the case packer group to evaluate its ability to detect a change in the component's health condition. Then, a clustering-based approach with pre-training is applied to the first dataset of the test rig for operating condition recognition and fault detection. A similar approach, with no pre-training, is then applied to the section cups dataset for fault detection and the extruder dataset for operating condition recognition. Finally, a classification-based approach consisting of an anomaly detector and a pre-trained classification model is used to detect and diagnose faults in the sealing group (displacement).

Table 32 Case studies goals and methods for novelty detection in evolving environments

Case study	Goal	Approach	Method	Feature selection/construction method
Case packer	Fault detection	Clustering-based from scratch	AAD	Pearson Correlation analysis and Principal Component Analysis

Test rig	Operating Condition recognition and fault detection	Clustering-based with pre-training	AAD + ADP	Manually selected features
Suction cups	Fault detection	Clustering-based from scratch	AAD + ADP	Pearson Correlation analysis
Automatic Machinery	Operating Condition recognition	Clustering-based from scratch	AAD + ADP	Principal Component Analysis
Sealing group (displacement)	Fault detection and diagnosis	Clustering-based with pre-training Classification-based (non-incremental)	kitNET + DT	Pearson Correlation analysis and Recursive Feature Elimination

Case 1. Case packer group. In this case, the main goal is to detect a known fault as soon as possible, without any training phase. Faults have been separately considered since they cannot occur in sequence. Hence, four scenarios have been simulated, in which the dataset corresponding to one fault is appended after the nominal behavior. In all cases, the main parameters of the AAD algorithm (t_1 and t_2, which compare in Eq. (42) and Eq. (44), respectively, and define the number of past observations to compare with the current point for establishing if a point is nominal or anomalous) are set equal to 5 and 3. Results of the AAD algorithm applied to the two sets of features, i.e., features selected through the Pearson Correlation analysis and the features extracted through the PCA, are shown in Figure 51a and Figure 51b, respectively. In this figure, the points assigned to different clusters are depicted in different colors and anomalies with red crosses. The performance of the algorithm applied to the features selected through the Pearson Correlation analysis is summarized in Table 33. In all cases, a false positive (red cross) is detected during the nominal behavior. Fault 1, Fault 2, and Fault 3 are detected after 5 cycles (0.75 seconds) of their occurrence, and a novel cluster is created when the anomaly corresponding to the fault is detected. Instead, when Fault 4 occurs, an anomaly is detected. However, the following points continue to be assigned to the cluster corresponding to the nominal behavior.

Table 33 Case Packer. Performance of the AAD

Fault	Detected Fault (cycle)	Latency (s)	N created clusters	N points	False Alarms
F1	105	0.75	2	Cluster 1 = 104 Cluster 2 = 96	69
F2	105	0.75	2	Cluster 1 = 104 Cluster 2 = 96	56
F3	105	0.75	2	Cluster 1 = 104 Cluster 2 = 96	41
F4	-		1	Cluster1 = 147	54

Finally, As shown in Figure 51b, the features extracted through the PCA do not allow distinguishing the nominal behavior from any of the four faults.

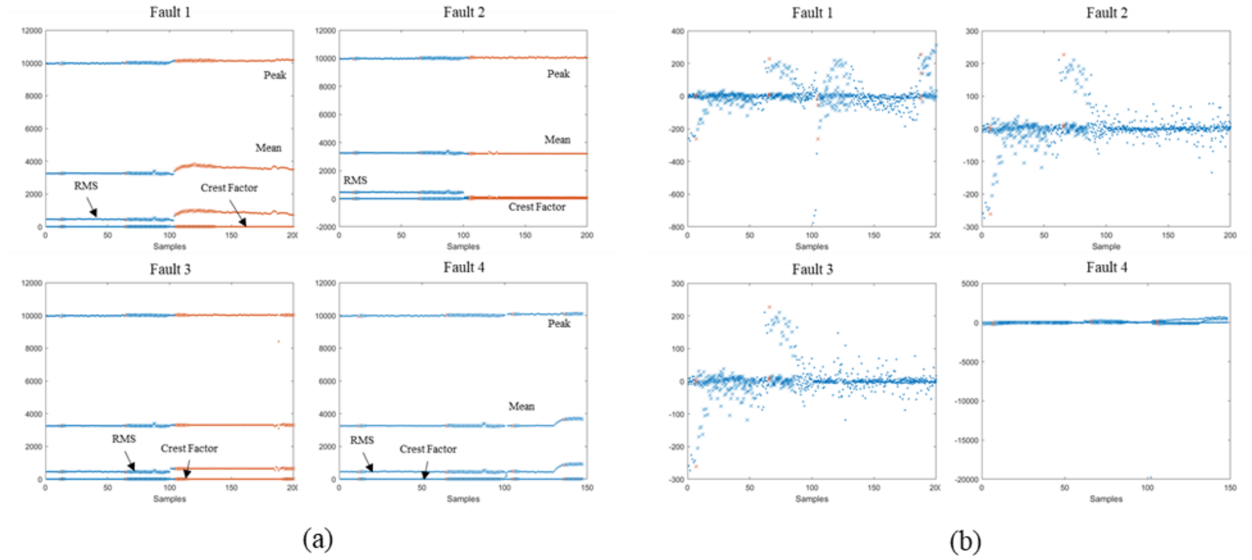


Figure 51 Case Packer. AAD for fault detection using features selected through the Pearson Correlation analysis (a) and constructed through the PCA (b)

Case 2. Test rig. For this case study, the manually selected features are chosen for online novelty detection. Indeed, they provide the best trade-off among the training time, the training accuracy, and the testing accuracy (Table 20 and Table 25). In particular, the low number of features of that set is more suitable for streaming applications. Unlike the previous case study, in this case a pre-training phase is conducted. Therefore, the ADP algorithm is first trained on the features corresponding to setting 1 and setting 2 to find the local parameters of the corresponding clusters, whose results are shown in Figure 52, where the blue dots correspond to the first setting, while the orange dots to the second one. Similarly, only one cluster is created for the fault detection task, meaning that the selected

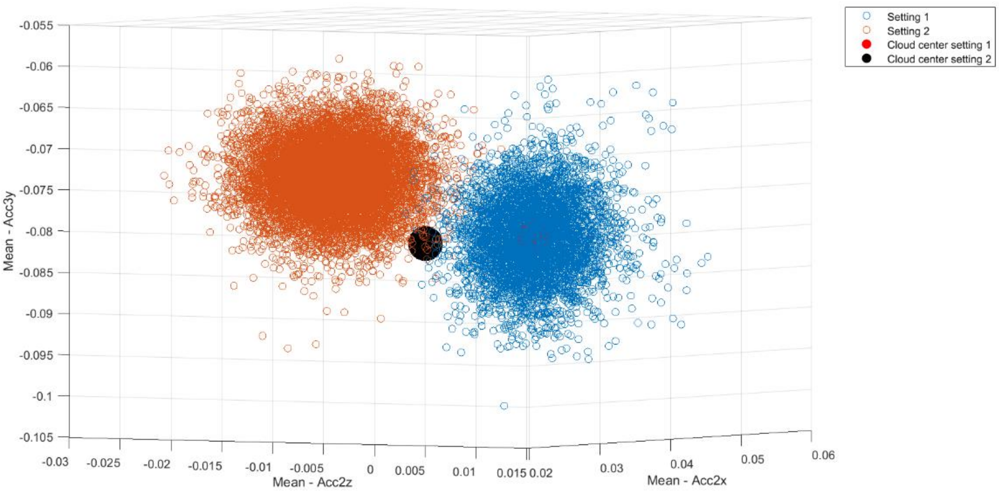


Figure 52 Test rig: results of clustering pre-training for operating condition recognition

component-level feature is not sensitive to the setting change. Then, the AAD+ADP algorithm is applied to all observations of each feature set. For the operating condition recognition task, streaming novelty detection aims to create two new clusters when setting 3 and setting 4 occurs. The goal of the fault detection task is to create a new cluster when the fault occurs. Results are shown in Figure 52, and the algorithm's performance is summarized in Table 34. Note that anomalies that do not determine a system's state change and are assigned to the current cluster can be considered measurement errors or anomalous peaks in the signal. In this case, the two parameters of the AAd algorithm (t_1 and t_2), are set equal to 15 and 3, respectively. For the operating condition recognition, the points corresponding to settings 1 and 2 are rightly assigned to the initialized clusters. The algorithm detects the second setting change after 15 seconds and the third setting change after 10 seconds. However, the fourth setting is not detected. For the fault detection task, the algorithm creates two new clusters. The first one is created after 18 seconds after the fourth setting change (yellow dots in Figure 53), while the second cluster is created after 175 seconds, which corresponds to almost 23 minutes before the system breakdown. This means that, if an alarm was triggered at the second cluster creation time, the system breakdown could have been avoided, the cluster 2 labeled as faulty, and classification models trained for future applications.

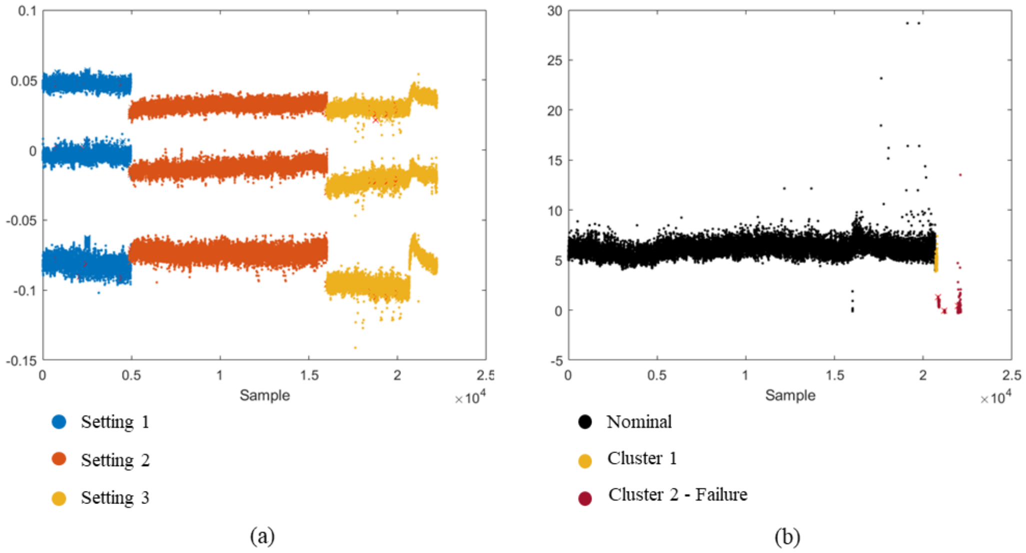
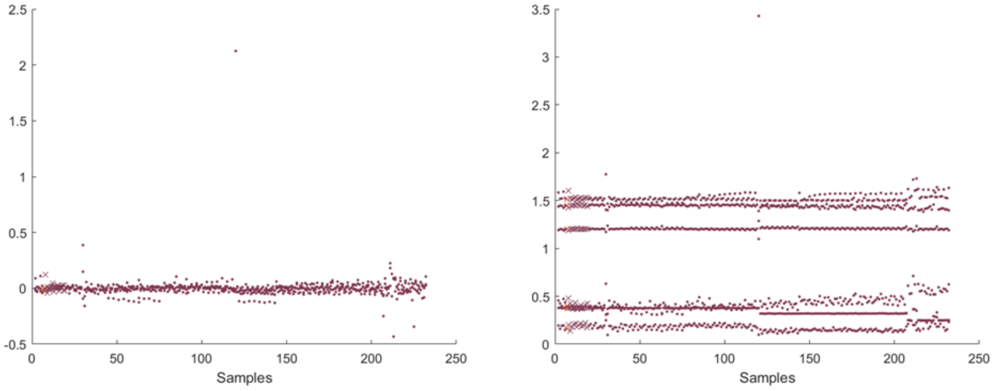


Figure 53 Test rig: results of Clustering-based novelty detection (with pre-training) for operating condition recognition

Table 34 Test rig: results of Clustering-based novelty detection (with pre-training)

Condition	Detected setting change (sample)	Latency (s)	N points assigned to the corresponding cluster	False Alarms
Setting 1	-		4,982	168
Setting 2	4.984	15	11,048	
Setting 3	16.032	10	6,187	
Setting 4	-	-	-	
Motor Fault			20,681	488
	20,682	-1,536	157	
	20,839	-1,379	1,380	

<u>Case 3. Suction cups</u>. The goal of this case study is to detect the occurrence of a failure as soon as possible. Unlike the case packer case study, in which the available fault conditions could not occur in sequence, in this case, the two available fault conditions correspond to two different severities of the same fault mode. Hence, the AAD + ADP algorithm is applied to detect the whole dataset. Initially, it was applied from scratch to the feature sets constructed in section 3.3.3, through the Pearson Correlation analysis and the PCA. However, as shown in Figure 54, any of these sets allow distinguishing the three conditions. On the contrary, if some of the peal, the peak-to-peak, the mean, and the RMS are selected and used as input of the streaming algorithm, the two different conditions are detected. By setting t_1 and t_2 equal to 5 and 3, respectively, results shown in Figure 55 and summarized in Table 35 are obtained. The first behavior change is detected after 5 cycle, equal to 0.67 seconds, while the second change is detected as it occurs, with zero latency. For each condition, a cluster is created, and points are correctly assigned to the clusters. In addition, 55 false alarms have been avoided.



(a) (b)

Figure 54 Suction Cups. Clustering-based Novelty Detection for fault detection using features constructed through the PCA (a) and selected through the Pearson Correlation analysis (b)

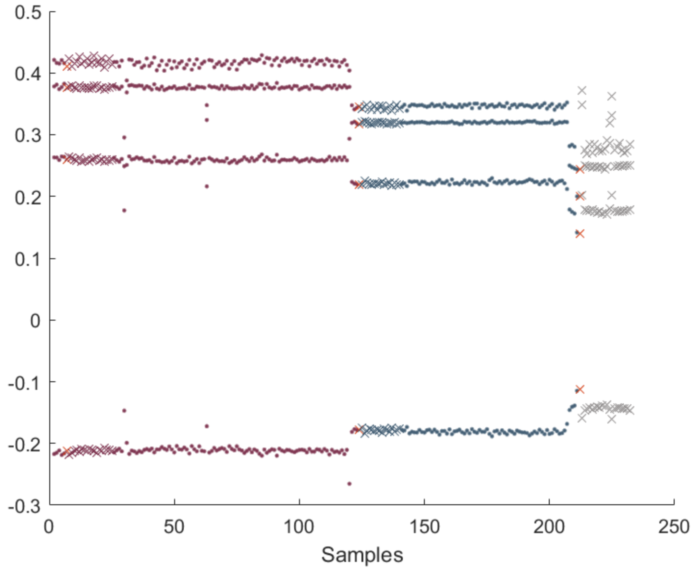


Figure 55 Suction Cups. Clustering-based Novelty Detection for fault detection using features manually selected

Table 35 Suction Cups. Performance AAD + ADP with manual feature selection

Fault	Detected Fault (cycle)	Latency (s)	N created clusters	N points belonging to the corresponding cluster	False Alarms
Severity 1	124	0.67	1	88	55
Severity 2	212	0	1	21	

Case 4. Automatic machinery. In this case, the AAD and ADP algorithms are applied to the extracted PCs with no prior training. However, since one of the settings is repeated, this case demonstrates that the algorithm is incremental. In other words, it can learn and recognize a condition that occurred a few minutes before. As summarized in Table 36, in both cases, the algorithm recognizes the change from setting 1 to setting 2, and from setting 3 and setting 4, after 9 data samples. In addition, for unit 2, the algorithm also recognizes the switch from setting 4 to setting 3, as data points are assigned to the existing cluster corresponding to the first operating condition. Results are also shown in Figure 57 and Figure 56, where the black dots correspond to the first set, the grey dots to the second set, and the red crosses represent the moment in which a changing behavior is detected. Note that, in both cases, there are several data points considered anomalous. However, none of them creates a new cluster. These points correspond to true anomalies in the data, like measurement errors or anomalous peaks, which are evident in the raw signals.

Table 36 Extruder: clustering-based

Unit	Setting change (detected)	Setting change (actual)
1	114	105
2	4257	4248
	5674	4904

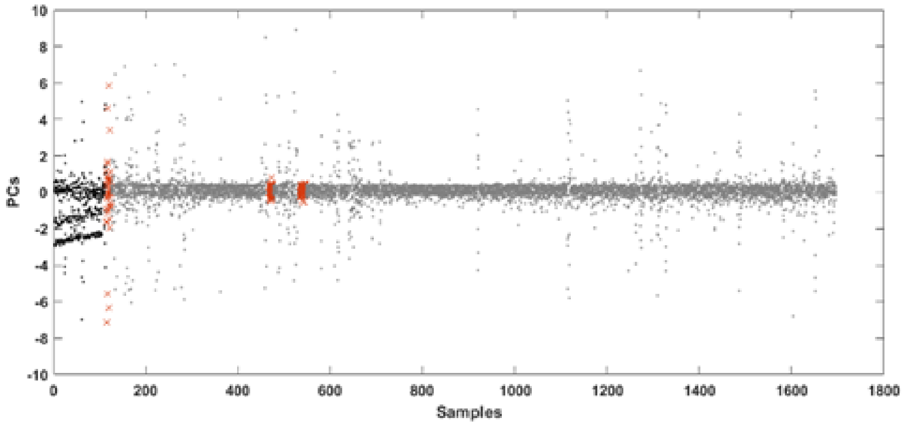


Figure 57 Automatic machinery: Online Anomaly Detection and Clustering (unit 1)

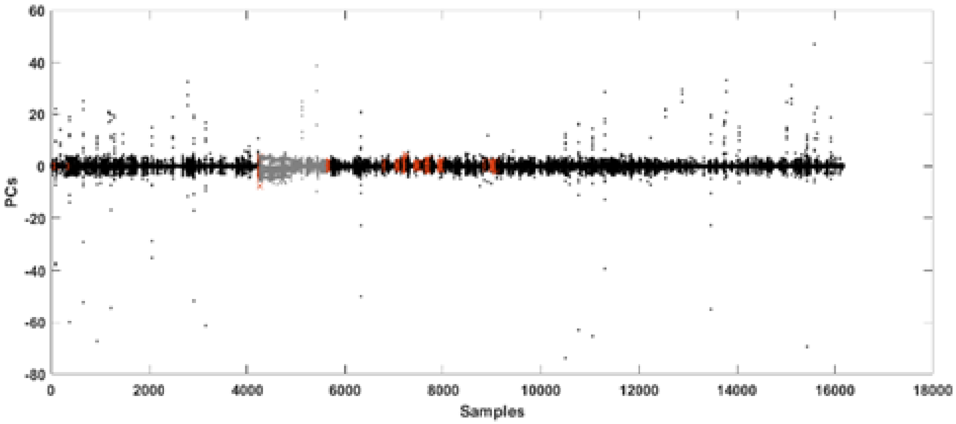


Figure 56 Automatic machinery: Online Anomaly Detection and Clustering (unit 2)

Case 5. Sealing Group (Displacement). For this case study, two different approaches are applied, i.e., the clustering-based with pre-training, and the non-incremental classification-based approach.

Concerning the first approach, the set of time features selected through the Pearson Correlation Analysis has been used as input for the AAD + ADP. First, the training set, including the 70% of observations for each condition, are used for clusters initialization. Then, the remaining observations are used to see whether the algorithm assigns the points belonging to a certain fault class to the corresponding cluster. Only 35 points belonging to the nominal condition are correctly assigned to the first cluster. Then, at point 36, the algorithm starts to assign points to the second cluster, which actually corresponds to the first fault. Then, the occurrence of an anomaly is detected in correspondence with the first fault, with a latency of 5 points (4 seconds). However, a new cluster is wrongly created. Then, at sample 230, the algorithm returns to assign points to the cluster corresponding to the first fault, which is correct. However, after only 15 points, points start to be assigned to the third cluster (wrongly created). A the end, another wrong switch from cluster 3 to cluster 2 is detected. Results of training and testing are shown in Figure 58.

Table 37 Sealing group (displacement). Performance of AAD + ADP training on features selected through Pearson Correlation Analysis

Condition	Detected setting change (sample)	Latency (s)	N created clusters	N points assigned to the corresponding cluster	False Alarms
Fault 1	492	4	2	470	307
Fault 2	-	-		-	
Fault 3	-	-		-	

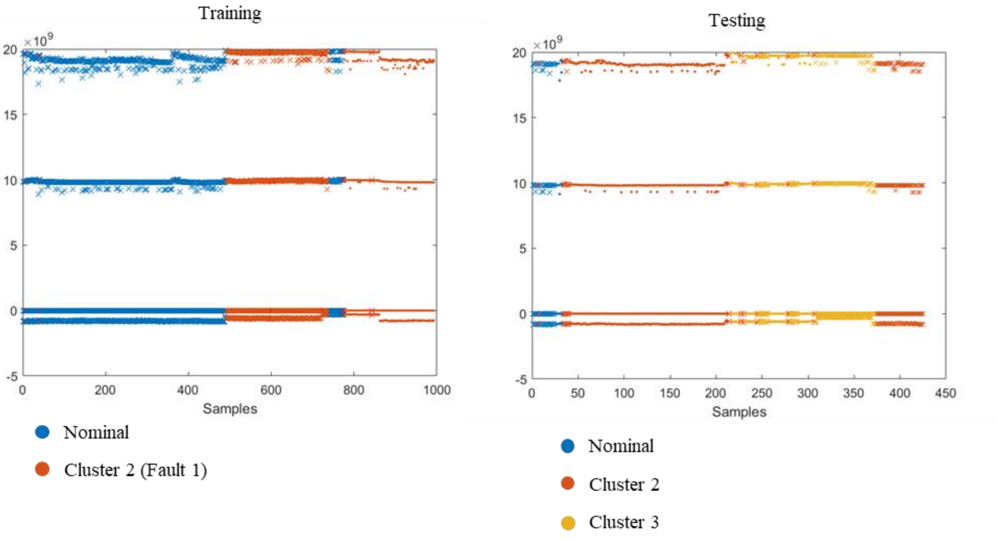


Figure 58 Sealing group (displacement). Training and testing results of clustering-based Novelty Detection for fault detection using features selected through Pearson Correlation Analysis and Recursive Feature Elimination

Concerning the second approach, two steps are performed. First, a training step is conducted to train models for anomaly detection and fault diagnosis. Then, the built models are applied in streaming to evaluate their effectiveness in recognizing the change of the component's health condition (fault detection) and determining whether it belongs to a known class or is a novel fault behavior. To these aims, two scenarios have been simulated. First, datasets corresponding to all available machinery conditions are considered in the training phase in order to select the best feature subset, the best classification model, and the best anomaly detection model in the hypothesis to know all the machinery behaviors. In this scenario, the online monitoring phase has the only objective of identifying points corresponding to a condition change as anomalies and classifying them in the correct fault class. Note that one dataset for each condition (e.g., dataset D1 for the nominal condition) is considered for the training, except for fault 2, for which the only available dataset is used for both training and online testing. In a second scenario, fault 2 has been removed from the training dataset to evaluate the novelty detection ability of the methodology. The datasets used in both scenarios for the training and the online testing are summarized in Table 38.

Table 38 Sealing group (displacement signals). Testing Scenarios

Scenario	Training datasets	Online testing datasets
1	D1 + D2 + D3 + D5	D1 + D7 + D8 + D2 + D9 + D4 + D6 + D3 + D5 + D10
2	D1 + D2	D1 + D7 + D8 + D2 + D9 + D4 + D6 + D3

Three different classification models are trained for the classification of four conditions, i.e., Decision Tree (DT), Support Vector Machine (SVM), and k-Nearest Neighbor (k-NN). For the training of each model, the k cross-validation with k = 10 is used to prevent overfitting, while the Grid Search (GS) is used for hyperparameters optimization. Table 39 shows the set of hyperparameters used for each model and the values range. In particular, the polynomial kernel function considered in the SVM is a third-degree function, and the γ value used in the Radial Basis Function (RBF) kernel function is equal to $\gamma = \frac{1}{2\sigma^2} = \frac{1}{(n_{features} * \sum_j (x_j - \bar{x})^2)} =$, where σ^2 is the sigma square value, $n_{features}$ is the number of features, and $\sum_j (x_j - \bar{x})^2$ is the variance of input variable X.

Table 39 Sealing group (displacement signals). Hyperparameter range values

Model	Hyperparameter	Range
Decision Tree	Maximum depth	{1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,20}

Support Vector Machine	C
	Kernel
		{1,5,50,100,500,1000}
		{linear, rbf, polynomial}

k-NN	K	{1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,20}

Several considerations lead to choosing a model rather than another. First, the best model should have the highest precision and recall. In addition, since the selected model will be applied to streaming data and provide real-time results, the execution time, i.e., testing time, is also critical. Finally, the model should potentially deal with an infinite amount of data. Hence, it has to be scalable and quickly trained on a large data set. Table 40 summarizes the precision, recall, velocity, and scalability obtained from applying the grid search algorithm. In addition, the selected hyperparameters for each model are also present in the table. In scenario 1, the model with the best precision and recall is the k-NN that with $k = 3$ achieves the 95.1% of precision and the 93.7% of recall. However, k-NN has low scalability since it computes the distance between a point and all other points at each iteration, making the process time-consuming for large datasets. Similar performances are also provided by the SVM, which also has high scalability. In addition, its execution time is equal to 0.02 seconds. However, the basic SVM does not provide the class estimation probability. Since one of the main goals of the present study is to provide an industrial solution that is easy to implement, and since both precision and recall of DT are slightly worse than those of the SVM, the DT has been chosen for the online fault diagnosis. In scenario 2, all models have the 100% of precision and recall.

Table 40 Sealing group (displacement signals) Fault diagnosis training performance of scenario 1 (S1) and scenario 2 (S2)

		Scenario 1			Scenario 2			
Model	Hyperparameters (Hyp.)	Optimal Hyp.	Precision	Recall	Optimal Hyp.	Precision	Recall	Velocity and scalability
DT	Maximum depth	9	92.3%	92.5%	5	100%	100%	High
SVM	C Kernel	1000 RBF	95.0%	93.6%	1 Linear	100%	100%	High
k-NN	K	3	95.1%	93.7%	1	100%	100%	Low

Three different anomaly detection algorithms are trained to determine the most suitable model for detecting anomalous behaviors. Hence, models are only trained on the nominal class, and then they are tested and evaluated using two different metrics. First, a variant of the traditional k-means clustering method is applied. K-means is a distance-based algorithm that requires only the number of clusters, k, to be set a priori. For novelty detection, k-means is trained only on a small subset of the available nominal dataset ($k = 1$). In this application, the training set size is set equal to 100 cycles. Then, a sliding window method is used to make it suitable for streaming applications. Hence, for each new data sample, the previous x data samples and the current sample are input to the algorithm. The sliding window method ensures a faster application since it allows keeping in memory only the last x values instead of the whole dataset. In this application, a sliding window including ten points is considered. In addition, the algorithm makes the inference only on one point at iteration since the x values have all been already assigned, except the last one, that is, the current one. Finally, since the classic k-means is expensive for large datasets, the mini-batch k-means (MBKm) proposed in (Sculley 2010) is used for streaming novelty detection. The second trained model is the One-Class Support Vector Machine (OCSVM). Finally, the last trained model for anomaly detection is an ensemble of autoencoders named kitNET (Mirsky et al. 2018). In this architecture, a feature mapper level maps the input features into k sub-instances, one for each autoencoder in the next level. The ensemble of autoencoders constitutes the anomaly detector, which aims to detect the anomalies based on a threshold on the reconstruction error, which is expressed in terms of Root Mean Square Error (RMSE). The training phase takes place in the ensemble layer, in which the autoencoders learn the nominal behavior of the input sub-instances and reconstruct the input minimizing RMSE. Then, the minimum RMSEs are sent to the output layer responsible for the anomaly score assignment during the execution phase. The higher the error, the higher the anomaly score of the test samples. In this application, the threshold on the reconstruction error is set to 0.5, while the number of autoencoders is set to 10. For the training phase, 500 nominal observations are used.

In order to evaluate how well the trained models recognize anomalies, two metrics have been used, i.e., the probability of detection or recall and the false positive rate (FPR). The first metric is computed through Eq. 36. The second metric evaluates how well the model recognizes the nominal observations and is defined by Eq. 50. In other words, a high value of this metric implies a high number of false alarms, i.e., points belonging to the nominal class, recognized as anomalies by the model.

$$FPR = \frac{FP}{FP+TN} \quad (50)$$

Where TN (True Negative) is the number of observations correctly classified as negative.

The application of mini-batch k-means, one-class SVM, and kitNET led to the results summarized in Table 41, Table 42, and Table 43, respectively. In addition, the parameters set by the user for each model are also present.

Table 41 Sealing group (displacement signals). Parameters and results of the mini batch k-means of scenario 1 (S1) and scenario 2 (S2).

		Scenario 1			Scenario 2		
Parameter	Value	Execution Time (sec)	$Recall$	FPR	Execution Time (sec)	$Recall$	FPR
K	2	6.7	0.3%	37.21%	1.98	0.41%	10.38%
Initialization time	100 cycles						
Amplitude of the sliding window	10						

Table 42 Sealing group (displacement signals). Parameters and results of OC-SVM of scenario 1 (S1) and scenario 2 (S2)

		Scenario 1			Scenario 2		
Parameter	Value	Execution Time (sec)	$Recall$	FPR	Execution Time (sec)	$Recall$	FPR
Initialization time	100 cycles	1.2	100%	61,42%	0.13	97.2%	-%
Amplitude of the sliding window	10						

Table 43 Sealing group (displacement signals). Parameters and results of kitNET of scenario 1 (S1) and scenario 2 (S2)

	Scenario 1				Scenario 2			
Parameter	Value	Execution Time (sec)	$Recall$	FPR	Value	Execution Time (sec)	$Recall$	FPR
Initialization time (feature mapping)	100 cycles	0.13	100%	6.54%	100 cycles	0.04	100%	0%
Initialization time (anomaly detection)	400 cycles				400 cycles			
Number of autoencoders	10				10			
Threshold	0.5				0.5			

In both scenarios, kitNET is the best model from all perspectives. It is fast, has the highest recall and the lowest false positive rate. On the contrary, the mini batch k-means has low recall values, and the value of the false positive rate is lower than the FPR of the OCSVM in the first scenario. In the second scenario, the FPR of OCSVM cannot be computed since both false positives, and true negatives are equal to zero. As a result, the kitNET has been chosen for online monitoring.

After the training phase, the online condition monitoring has been tested. The pseudo-code of that phase is shown in Table 44. For the scenario 1, the data flow has been generated by sequence the observations corresponding to the nominal condition (D1 + D7 + D8), fault 1 (D2 + D9), restored condition (D4+D6), fault 2 (D3), and fault 3 (D5 + D10). In scenario 2, the online testing has been carried out on datasets D1, D7, D8, D2, D9, D4, D6, and D3. The first 300 cycles are used for the initialization. Hence, the anomaly detector is only activated after 300 cycles. Then, at each iteration k, i.e., every 800 ms, the following steps are carried out. First, the selected features are extracted. Then, the feature vector is input to kitNET, which computes the anomaly score. If the reconstruction error is lower than the fixed anomaly threshold (Th_a), the feature vector is considered nominal, and the algorithm goes to the next iteration. If the reconstruction error is greater than or equal to Th_a for two consecutive feature vectors (stored in the counter a), the feature vectors are input to the Decision Tree, which assigns the class probability membership to the labeled observations. If the maximum probability is greater than the fixed classification threshold (Th_c), the predicted class is considered accurate. Hence, if the predicted class is nominal, the algorithm goes to the next iteration. Otherwise, a warning message indicating the occurred fault is generated, and the machinery is stopped. If the maximum probability is lower than Th_c, a possible novel behavior occurred, and a human check may be required to establish the anomaly's cause. Note that the choice of the minimum number of detected anomalies that have to occur before the classifier activation is made with the goal of finding a good trade-off between the false alarm rate and the latency of the classifier. A single anomaly would generate a greater probability of error. On the other hand, if the classifier is activated after more than two anomalies, the fault or the novel behavior would be recognized after three cycles. In this case, it would mean that at least 12,8 products would be discarded for quality issues.

Table 44 Sealing group (displacement signals). Pseudo-code of the online condition monitoring.

```
Initialize:
Input: Optimal feature subset, cycle duration ($d$), trained DT, trained kitNET, Anomaly threshold
($Th_a$) classification threshold ($Th_c$)
While $i < d$
Do nothing
$k = 1$
Compute optimal features
Apply kitNET and assign the anomaly score
    If anomaly score < ($Th_a$)
        Increment $k$ by 1
    End
    Else
        Increment $a$ by 1
        End
If $a \geq 2$
        Apply DT and compute the CPE for each known class $c$
                If $\max_c CPE > (Th_c)$
                  If $\max_c CPE$ for $c$ = nominal
                    Increment $k$ by 1
                  End
                  Else
                    Fault corresponding to class $c$ occurred
                    Stop the machinery
                  Break
                Else
                  A possible novel behavior has been detected
                  Generate an alarm
                Break
End
End
```

Note that the algorithm was not stopped in the online condition monitoring if a known fault was recognized in order to test the methodology. However, if a fault is detected in real applications, the machinery should be stopped as soon as possible. In the first scenario, the algorithm correctly classified observations belonging to the nominal class, fault 1, and fault 2. On the contrary, the change from the restored condition and fault 3 was not detected. Hence, observations belonging to fault 3 were wrongly considered nominal. Moreover, in eighteen cases, the kitNET detected two anomalies in a row. However, the DT classified them as nominal with a class probability membership value equal to 1. Therefore, no alarm was generated. Note that because models were trained with four classes in this scenario, the classification threshold was set equal to 0.5. In the second scenario, observations belonging to the nominal condition and fault 1 were correctly classified. When fault 2 occurred, the kitNET detected two anomalies in a row. Hence, the DT was applied, which assigned a class probability value of 0.87 and 0.13 for the fault 1 and nominal class, respectively. Because models were trained with only two classes in this scenario, the classification threshold was set equal to 0.9. Hence, when fault 2 occurred, an alarm indicating that a possible novel behavior occurred was generated.

4.3.1. Deep Learning-based novelty detection

This section applies an incremental and non-incremental classification-based approach to the test rig dataset for operating condition recognition. Both Machine Learning (Online clustering, LOF, PCA, SVM, IF, MLP) and Deep Learning (CNN, LSTM) models are applied to the dataset and evaluated in terms of effectiveness and efficiency to demonstrate the outperforming results of Deep Learning approaches. In particular, two scenarios are considered. In the first scenario, which is non-incremental, the models are first trained on a single operating condition. Then, their ability to discriminate between the known condition and the others, i.e., novel conditions, was analyzed. This scenario corresponds to the typical approach requiring the models' re-training each time a new condition occurs. In the second scenario, the models are evaluated in terms of their ability to incorporate new knowledge. This scenario evaluates an incremental learning approach, in which the ability to learn machinery conditions that were unknown at the time of the initial offline training is assessed. In addition, for each scenario, two levels of analysis are conducted. In the first case, each sample is considered separately; the prediction accuracy is computed by Eq. 51, where N is the number of samples, *1(x)* is converts the outcome of a boolean condition (true or false) into 1 or 0, y_i and $\tilde{y}_i$ represent the true and the predicted labels for the i-th sample, respectively

$$Accuracy\ (Acc) = \frac{1}{N}\sum_{i=1}^{N} 1(y_i = \tilde{y}_i) \qquad (51)$$

A second level of analysis considers a batch of samples instead of single samples. In this case, the batch accuracy is given by Eq. 52, where $|B_k|$ indicates the cardinality of the k-th batch, with $k = 1, \dots, M$ and M the number of batches, $y_{k,j}$ and $\tilde{y}_{k,j}$represent the true and the predicted labels for the j-th sample in the k-th batch, respectively.

$$Batch\ Accuracy\ (B.Acc) = \frac{1}{M}\sum_{k=1}^{M} 1(\sum_{j=1}^{|B_k|} 1(y_{k,j} = \ \tilde{y}_{k,j}) = |B_k|) \qquad (52)$$

Hence, a prediction is considered correct when all the samples of a batch are correctly predicted. Finally, the models' performances are also evaluated in terms of the computational time of both training and testing.

<u>Non-incremental scenario</u>. To carry out this evaluation, the models are in turn on one batch at a time, and the models' evaluation is done on the remaining batches. This evaluation is repeated until each batch has been used for model training. The expected behavior is that no novelty state is detected for all the test data associated with the same machine condition used for the training, while new machinery conditions are detected for the other samples. The results of this experiment are shown in Table 45, where for each model and training machinery condition, the accuracy, given by Eq. 51, and the batch accuracy, given by Eq. 52, computed over all the datasets are reported. The models that provide the worst performance are SVM, IF, and LOF, while the remaining models are almost equivalent. With the exception of C3, where they produce poor results, these three models generate good performance on single samples (i.e., they obtain accuracy values in the range 0.66-0.97). However, they produce many false alarms (i.e., they obtain batch accuracy values in the range 0.3-0.6) in batch-level evaluation. Furthermore, it is possible to see how the most difficult operating condition to identify is C3, in which SVM, IF and MLP show the most significant reductions in

performance, while the models with the highest effectiveness recognize C1. This differentiation in performance does not apply to LSTM and CNN which are highly effective on all scenarios without distinction.

Table 45 Test rig: Breakdown of model performance by operating condition

Algorithm	All		C1		C2		C3	
	Acc.	B. Acc.	Acc.	B. Acc.	Acc.	B. Acc.	Acc.	B. Acc.
Clustering	0.988	0.758	0.998	0.941	0.978	0.627	0.995	0.830
LOF	0.817	0.572	0.990	0.840	0.915	0.544	0.411	0.326
PCA	0.965	0.808	0.999	0.945	0.939	0.752	0.981	0.772
SVM	0.658	0.351	0.982	0.715	0.703	0.290	0.189	0.067
IF	0.880	0.619	0.994	0.867	0.967	0.598	0.561	0.379
MLP	0.957	0.911	1.000	1.000	0.945	0.906	0.933	0.821
LSTM	0.989	0.944	0.998	0.977	0.984	0.927	0.990	0.942
CNN	0.989	0.939	0.998	0.980	0.984	0.919	0.989	0.933

Incremental scenario**.** In this experiment, we simulate the adoption of the models in a dynamic scenario where a continuous monitoring of the machinery is performed, and an incremental knowledge of the operating conditions of the machinery is learned by the diagnostic system. To create this experimental scenario we have considered the three settings shown in Table 46.

Table 46 Test rig: Incremental scenario configurations

Conf	Training Set	Test set	
		Known Set	Novel Set
S1	C1 (10 min.)	C1 (70 min)	C2, C3, C4 (150 + 70 min + 10 min)
S2	C1, C2 (10 + 10 min.)	C1, C2 (70 + 150 min)	C3, C4 (70 min + 10 min)
S3	C1, C2, C3	C1, C2, C3	C4

In the first configuration, each model is trained exclusively on a 10-minute batch of C1 and an operating cycle is then applied to the other machinery settings. In the second configuration, it is assumed that the model has also learned of the existence of C1 and C2, and the same operating cycle is applied to other machinery settings. Finally, the last configuration evaluates the behavior of each model when trained jointly on all three states. Note how each model stores a limited amount of data for each state compared to the totality of measurements made (i.e. only 10 minutes of data for each state are considered). In this way 1) the model is trained quickly and it can continue to monitor the behavior of the machinery and 2) no dedicated storage is needed to store the entire measurement history. Results of this scenario are reported in Table 4, where for each model the batch accuracy is reported both for the entire test set, i.e., all observations included in both known and novel tests as defined in Table 46, and the known and novel sets as defined in Table 3, individually. Similar results were obtained considering the record-level accuracy, which were not reported due to space constraints. From Table 47, it is possible to observe that the configuration where the models perform best is the first one (S1), where the models are trained exclusively on C1. This confirms that C1 is significantly different from the other states, thus facilitating its distinction with respect to the other states. The worst-performing models are SVM, LOF, and IF in this configuration, which correctly recognize C2, C3, and C4 as new operating conditions. However, they tend to categorize batches belonging to C1 wrongly. In the second configuration, on the other hand, there is a significant reduction in the models' effectiveness, except LSTM and CNN. In more detail, the LOF, PCA, and Clustering models hardly recognize C1 and C2 as already known operating conditions (low accuracies on the known set). This is probably due to the integration of C2 with the training data, which has made the separation between the operating conditions less marked. Finally, analyzing the third configuration makes it possible to note how all the models produce a good performance in recognizing the known set. However, they are no longer able to discriminate it with respect to the state C4, which presents very similar characteristics in the phase preceding the failure compared to other conditions. In particular, all models except LSTM and CNN have an accuracy equal to 0, i.e., they cannot correctly predict even a batch. A more detailed analysis of these results revealed that the record-level accuracy of these models varies in the range 0.1-0.33, while for LSTM and CNN in the range 0.81-0.83.

Table 47 Test rig: Models batch accuracy in online scenario

Algorithm	C1			C2			C3		
	Test	Known	Novel	Test	Known	Novel	Test	Known	Novel
Clustering	0.941	0.732	1.000	0.567	0.503	0.700	0.410	0.455	0.000
LOF	0.840	0.304	0.990	0.456	0.395	0.586	0.690	0.767	0.000
PCA	0.945	0.750	1.000	0.235	0.000	0.729	0.000	0.000	0.000
SVM	0.715	0.321	0.825	0.618	0.565	0.729	0.790	0.878	0.000
IF	0.867	0.411	0.995	0.618	0.626	0.600	0.757	0.841	0.000
MLP	1.000	1.000	1.000	0.871	1.000	0.600	0.900	1.000	0.000
LSTM	0.997	0.893	1.000	0.935	0.952	0.900	0.743	0.772	0.476
CNN	0.980	0.911	1.000	0.922	0.912	0.943	0.790	0.847	0.286

To further inspect the superiority shown by the LSTM-based AutoEncoder over competitive methods, we propose in Figure 59 a visual inspection of its internal representation for the examined operating conditions. In the figure, this representation is compared with the raw distribution of operating states when projected into a two-dimensional space generated by the popular t-SNE technique [15]. As you can see, the embedded space created by the LSTM emphasizes the separation between the different operating conditions more than in the original feature space.

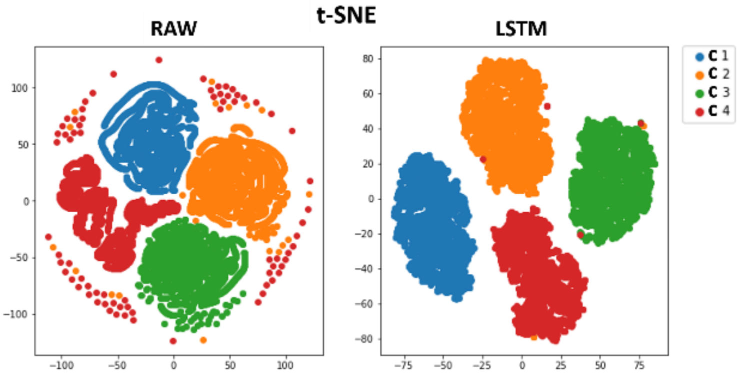


Figure 59 Test rig: LSTM-based AutoEncoder embedded space compared with the original feature space

Impact of data size on performance. In this subsection, the impact of the size of the training data on model performance and computation time is assessed. For the evaluation of the performance, each model is trained on a variable number of batch-es associated with the same machinery state, i.e. 5, 10, 20, and 30 minutes of data. Each model is then required to recognize the test data as referring to a novel or already known machinery condition. Results of this experiment are shown in Figure 60, where for each model, the evolution of batch accuracy as the training size increases is reported for all the three states considered. Results show that the performance of deep learning models is high even with a small amount of training data (e.g., 5 minutes) and is not influenced by the availability of further training data. As for the other approaches, the variable availability of training data influences their performance (except for the PCA, which produces equivalent results for all the settings considered). In particular, Clustering and SVM are more effective as the size of the training set increases: SVM achieves accuracy improvements in the range between 9-20%, while Clustering between 6-20%. Similar trends are also confirmed for the LOF and IF models on C1 and C2. However, the latter two models perform worse as the size of the training set increases when trained on C3. A more detailed analysis of these results has revealed that in this setting, they are unable to distinguish new states (i.e., LOF and IF generate a batch accuracy of 3 and 6%, respectively). The evaluation of computational time with varying training sizes is conducted to assess 1) the impact of a training process on the inactivity of each model and 2) their velocity in detecting possible new operating conditions of the machinery (i.e., the prediction time). Note that models are run on a VM deployed on Google Cloud with 12 GB of RAM, GPU K80, and Intel(R) Xeon(R) CPU @ 2.30GHz. For each model, both training and test times are considered. Each model is trained on 5, 10, 20 and 30 minutes of data, and the relative times are recorded. In addition, the prediction times over 1 sample, 10-, 20- and 30-minute batches were also recorded. The results of these two experiments are displayed in Figure 61 on the left and right plots, respectively. From the plot on the left in Figure 61, it is possible to observe how the models require very different training times. Models like SVM and PCA only take a few milliseconds to complete the training. Times equal to almost two orders of magnitude are instead produced by IF and MLP models. On the other hand, clustering and CNN require times in the order of a second or a few tens of seconds. Finally, the LSTM model is the one that produces the highest training times: from 4 minutes in the configuration with fewer data to a maximum of 20 minutes. These trends are confirmed by analyzing the prediction times shown in the right plot, although the latter are two orders of magnitude lower. From these results, it is also possible to note that the Clustering approach produces significantly higher prediction times than the other techniques (except for LSTM). This is due to the quadratic nature of the approach, although it was partially alleviated through a mixed training strategy where a first clustering solution was produced from 1000 samples, and then the remaining data were included in an online fashion. Finally, it is possible to observe how the time required to evaluate whether a single sample be-longs to an already known or novel state is a few milliseconds, making them all suitable for operating in an online scenario.

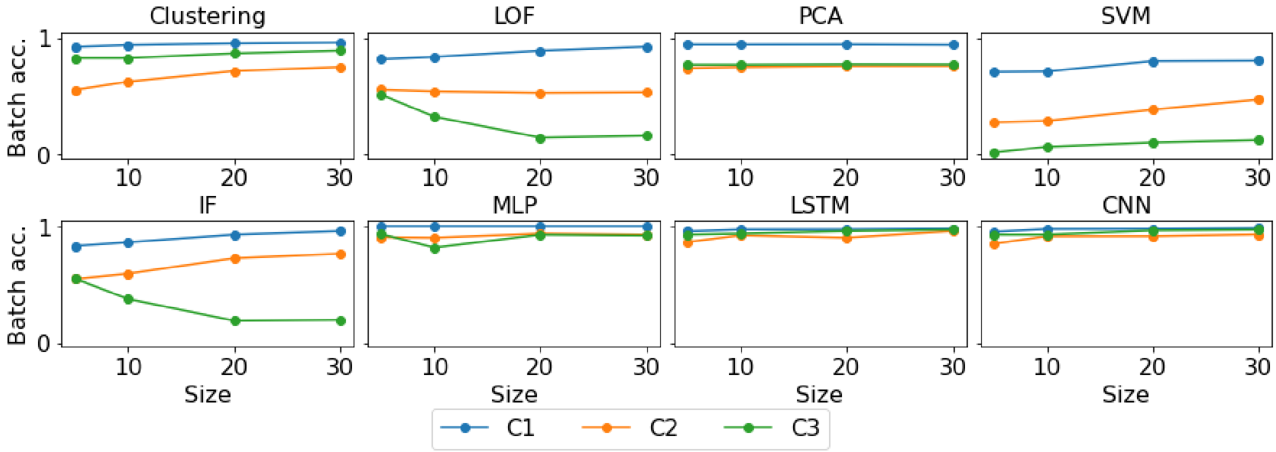


Figure 60 Test rig: performance evaluation as the training size increase

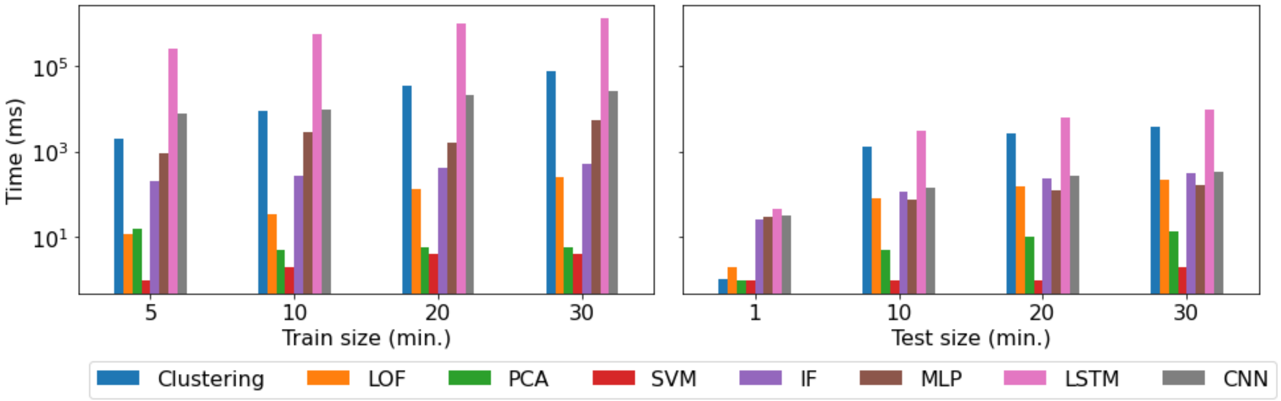


Figure 61 Test rig: train (left) and test (right) performance by varying the data size

4.3.2. Results and discussion

In this chapter, data-driven fault diagnosis and detection in evolving environments is addressed, which aims to solve the issues that emerged from the data collection process in industrial contexts, i.e., data availability, data quantity, and data incompleteness. In particular, unsupervised learning and semi-supervised learning paradigms are integrated with supervised learning models to determine, in real-time, the health condition of monitored components and the operating condition in which systems are working. To this aim, fault detection and diagnosis models have to make predictions on continuous data flow, i.e., data streams, which require an incremental learning approach that can distinguish what is known from what is new, i.e., not included in the model training step. This is what the literature calls novelty detection in evolving environments.

The main considerations and outcomes of data-driven novelty detection in evolving environments are summarized in this section, according to the three research outcomes pointed out in section 1.5:

1. How to build an effective diagnostics model on the available training data
2. How to build effective novelty detection models for the concept drift detection and multi-class classification of detected anomalies
3. How diagnostics and novelty detection can be integrated in an IIoT environment, including edge and cloud computing.

The first outcome is related to the analysis conducted in section 4.2, where the features extracted in the previous chapter for each case study are used as input of classification models for both fault diagnosis and operating condition recognition. Besides the evaluation of the classification models, the effectiveness of the extracted and selected feature sets is also evaluated to determine the effect of data reduction on classification performance. In particular, feature sets are evaluated according to two objectives. First, a comparison among different dimensionality reduction methods is performed to determine the best technique, if any. Results are shown in Figure 62, where the training accuracy, training time, and testing accuracy are shown for each case study and each technique. These values have been computed as the average of the results of each classification model since the goal is not to evaluate the specific classification model performance. Then, the reduced feature sets are evaluated against the original feature set, without any dimensionality reduction technique, to evaluate the effect of reducing the number of variables of the input dataset on classification models' performance. Results of the feature evaluation process are summarized in Table 48, where the applied feature selection and construction methods, the resulting number of features, the mean increment in both training and testing accuracy (indicated with $\uparrow Acc_{train}$ and $\uparrow Acc_{test}$,respectively), and the mean training time reduction (indicated with $\downarrow T_{train}$) obtained from reducing the number of variables are reported for each case study. The mean values of accuracy increment and the mean training time reduction are computed as follows:

$$\overline{Acc} = \left(\sum_{i=}^{N} \frac{Acc_{2i}-Acc_{1i}}{Acc_{1i}}\right) * \frac{1}{N} \tag{53}$$

$$\overline{TT} = \frac{\left(\sum_{i=}^{N} TT_{2i}-TT_{1i}\right)}{N} \tag{54}$$

Where N is the number of applied classification models, Acc_{1i} and TT_{1i} are the accuracy and the training time obtained by training model i without feature construction, and Acc_{2i} and TT_{2i} are the accuracy and the training time obtained by training model i with feature construction. Note that a negative value represents a decrement of the accuracy value and an increment of the training time.

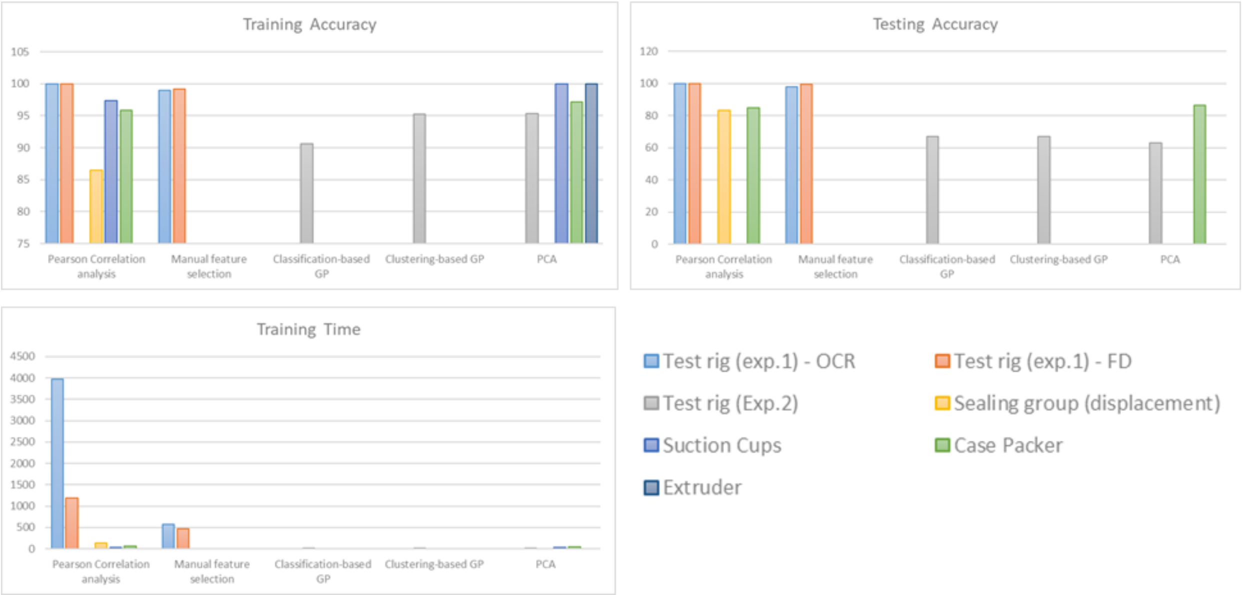


Figure 62 Comparison of feature extraction and selection methods on classification performance

From Figure 62, it can be seen that the Pearson Correlation analysis provides good training accuracy values (greater than 95%) for all case studies, except for the sealing group (displacement signals), in which the mean training accuracy is lower than 90%. Testing accuracies range from 80% to 100%, while training times are higher than other methods, especially considering the case of the first experiment conducted on the test rig for the operating condition recognition task. However, in this case, the number of selected features was 32, which is four times, on average, greater than other feature sets. Indeed, the mean training time of other techniques is much lower. Similar to the Pearson Correlation analysis, the PCA provides good training accuracy and training time results. However, the testing accuracy is lower. Finally, while the clustering-based GP provides similar results to other techniques, the classification-based GP provides the worst results in training and testing accuracy.

Table 48 Feature effectiveness evaluation in offline and batch fault diagnosis

Case study		Data reduction method	N features	↑ Acc_{train}	↓ T_{train}	↑ Acc_{test}
Test rig (exp. 1)		1. PC analysis	32	0 %	− 1,199.6 %	− 0.1 %
		2. Manual Feature selection	4	− 1.03 %	2,086.5 %	− 1.13 %
Test rig (exp. 2)		3. Classification-based GP	1	− 10.35 %	28.21 %	30.84 %
		4. Clustering-based GP	1	− 4.97 %	29.22 %	30.82 %
		5. PCA	4	− 4.85 %	28.81 %	25.84 %
Sealing group (displacement)		4. PC analysis + RFE	4	−3.8 %	− 98.1 %	11.1 %
Suction cups		1. PC analysis	6	−0.2 %	1.2 %	-
		2. PCA	3	2.4 %	− 5.4 %	-
Case packer		1. PC analysis	4	− 3.3 %	− 12.6 %	− 3.7 %
		2. PCA	4	− 1.9 %	− 7.8 %	− 2.4 %
Automatic Machinery	Unit 1	PCA	4	0 %	0 %	-
	Unit 2		5			

From Table 48, it can be seen that, in most cases, the training accuracy decreases with the application of dimensionality reduction techniques. Indeed, except for the first experiment of the test rig, the original feature sets only include nine features, which cannot be considered a high-dimensional dataset. Even for the test rig, the advantage of reduction techniques on the training accuracy is not evident. However, the testing accuracy generally increases, especially for the features constructed with the GPs. Indeed, these models evaluate the feature set against a classification or clustering model, for which the cross-validation is performed. An unexpected result is an increase in the training time in most cases. The reason may be the difficulty of learning more complex relationships between a reduced number of variables since the model that mainly contributes to this result is the SVM, which is based on an optimization model.

Concerning the first outcome, the evaluation of classification models' performance for operating condition recognition and fault diagnosis is also conducted, whose results are depicted in Figure 63. For each case study, the built classification models, the number of samples used during the training phase, the average training time, and the average of resulting accuracies computed on the training and testing sets using Eq. (35) are reported. The mean values of training accuracy, the testing accuracy, and the training time are computed considering all extracted feature sets, with and without feature construction. In the first case, if more models are applied for feature extraction, construction, or selection, the average of these values is computed on all the cases. From the graphs, it can be seen that the performance of each model strongly depends on the specific case study and the specific set of features. In general, the SVM requires more training time, followed by the KNN. However, the SVM has better generalization ability. The DT works well with original feature sets, but the training accuracy decreases with smaller sets.

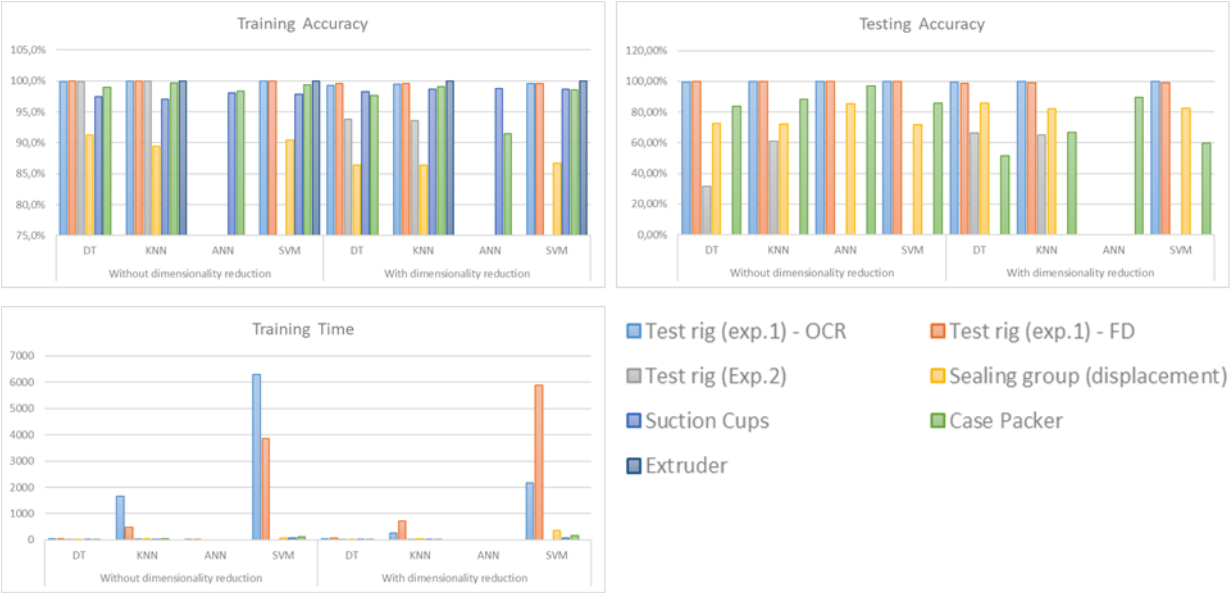


Figure 63 Comparison of classification models using original feature sets and reduced feature sets

The second outcome is related to the analysis conducted in section 4.3 on the online and streaming analysis. To this aim, the features extracted in the previous chapter for each case study are used as input of novelty detection approaches for both fault detection and diagnosis and operating condition recognition. On the contrary, subsection 4.3.1 uses Deep Learning models directly applied to raw signals. Table 49 summarizes for each case study and each feature set the results of the clustering-based novelty detection using AAD+ADP, in terms of the mean latency in detecting a setting or health condition change and the number of avoided false alarms. The mean latency is computed over all detected changes (only late detection is considered). In addition, the signal segments from which the features are extracted are also reported.

Table 49 Clustering-based novelty detection performances /AAD + ADP)

Case study	Window length (s)	Training	Dimensionality reduction method	N created clusters	Mean latency (s)	N false alarms
Case packer	0.15	No	1. PC analysis	2 (for each scenario)	0.75	55
			2. PCA	-	-	-
Test rig (OCR)	1	Yes	1. Manual Selection	3	12.5	168
Test rig (FD)				3	-1,457,5	488
Suction cups	0.134	No	1. PC analysis	-	-	-
			2. PCA	-	-	-
			3. Manual selection	3	0.33	55
Automatic Machinery	1.800	No	1. PCA		$4.72*10^5$	-
Unit 1				2		
Unit 2				2		

Sealing Group (displacement)	0,008	Yes	1.	PC analysis	2	4	307

From the table, it can be seen that the AAD+ADP algorithm recognizes the change of the component/system behavior in most cases. In addition, when the pre-training phase is conducted, the algorithm correctly assigns the points to the existing clusters or correctly creates new clusters. The latency of the algorithm is also acceptable in all cases. Indeed, in the case of fault detection, the latency is always lower than one second. In the case of the operating condition recognition, the latency is much higher. However, real-time feedback on the implemented setting is unnecessary, and higher latency is acceptable. The primary issue of clustering-based approaches concerns the strong dependency on the selected features. In most cases, the extracted PCs do not recognize any change in the system's behavior, although they provided the best training accuracy in the batch fault diagnosis.

The two different approaches for novelty detection in evolving environments, i.e., classification-based and clustering-based, applied in section 4.3, are depicted in Figure 64 and Figure 65.

The traditional classification-based approach takes as input the vector of features extracted and selected during the training phase, and determines, for each observation, whether it is anomalous or not, based on a pre-trained anomaly detection model. If an anomaly is detected, a pre-trained classifier assigns to it a class label. The current behavior is correctly classified if the class probability estimation is greater than a certain threshold. Otherwise, a novel behavior is detected. Hence, this approach consists of two phases, i.e., training and online condition monitoring. Since the training is the same as the offline and batch fault diagnosis, only the online condition monitoring is discussed in this section. Although results show that the models' performances in terms of latency and accuracy are better than the clustering-based approach, this classification-based novelty detection approach has several drawbacks. First, it needs to train two different models, one for anomaly detection and one

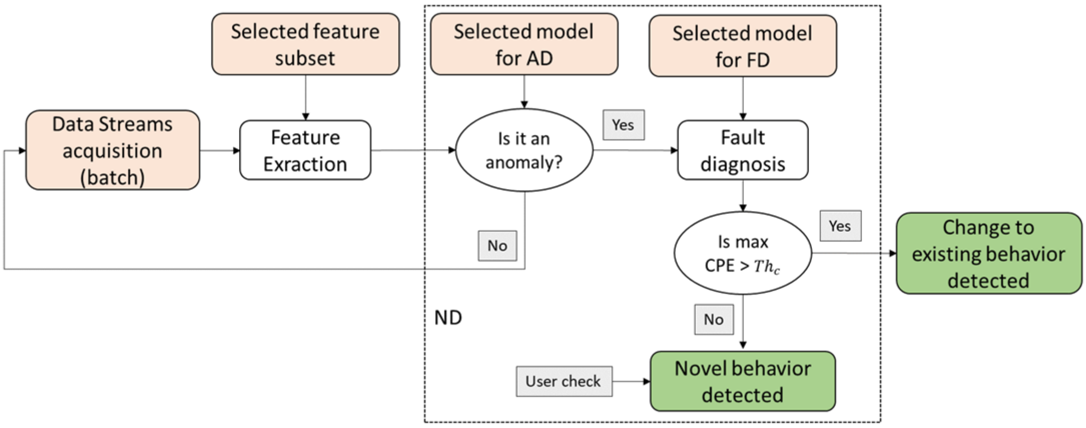


Figure 64 Classification-based approach for novelty detection in evolving environments

for classification. Therefore, as in the case of offline batch diagnosis, more data are required to avoid overfitting; it depends on the extracted features and the pre-processing. In addition, results strongly depend on the probability class estimation, which is not a hyperparameter and is defined by the user. Finally, classification models have to be re-trained each time a novel behavior is detected, resulting in a non-incremental approach

The clustering-based approach takes as input the feature vector extracted in the training phase and the local parameters of existing clusters (known behaviors), if any. Then, if the arriving observations are considered anomalous for a specific time, or the number of samples, a change in the behavior is detected, and the nearest cluster is identified. Finally, if the current global density is greater than the local density of the nearest cluster, the point is assigned to a novel cluster, meaning that a novel behavior has been detected. Otherwise, the point is assigned to the nearest cluster, meaning that the point is considered to belong to existing behavior. This approach does not need any training phase, except for feature extraction and selection, which strongly affects the algorithms' performances. In addition, although it is an incremental approach, offline labeling of clusters is needed.

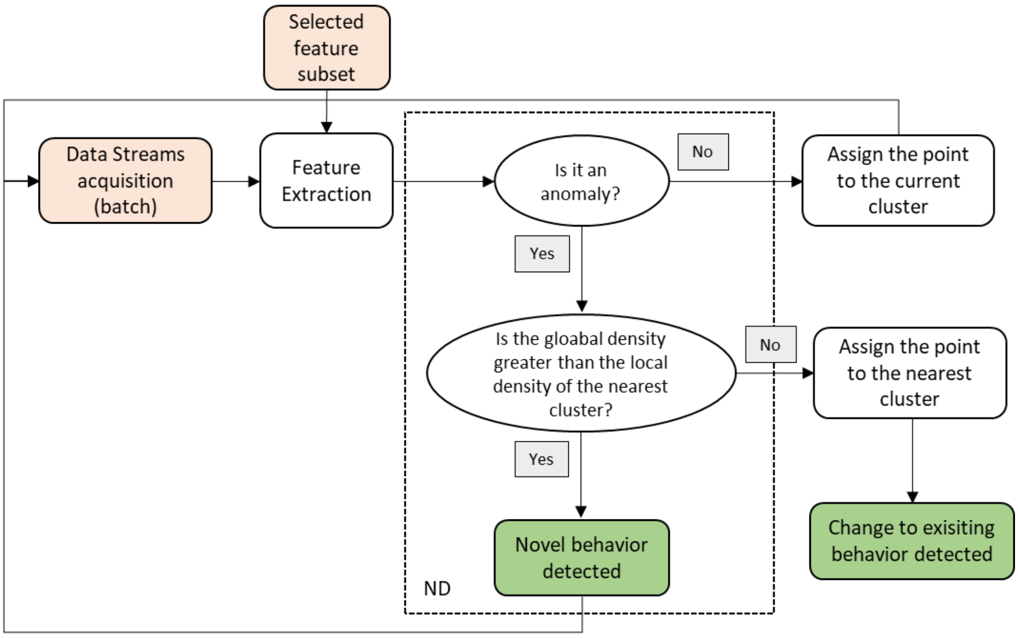

Selected feature subset
Data Streams acquisition (batch)
Feature Extraction
Is it an anomaly?
No
Assign the point to the current cluster
Yes
Is the gloabal density greater than the local density of the nearest cluster?
No
Assign the point to the nearest cluster
Yes
Novel behavior detected
Change to exisiting behavior detected
ND

Figure 65 Clustering-based approach for novelty detection in evolving environments

On the contrary, the DL-based approach for novelty detection does not depend on previously extracted features and works well with few training data (as shown in Figure 61)—the less the training data, the less the generalization ability.

The third outcome is related to the computation distribution and information transmission in edge-cloud IIoT infrastructure. The proposed architecture is shown in Figure 66. Only the model inference on the extracted feature vectors is performed concerning the computation at the edge. On the contrary, models' training and selection are performed in the cloud. Data transmission is the relevant aspect of this framework. Two kinds of data flow have to be considered. The first one is the flow of low-frequency data from the edge to the cloud. In this process, the labels of the different observations, corresponding to event data and machine settings, are transmitted. This information has to be sent as soon as possible (in real-time) to make decisions on maintenance interventions and spare parts supply. The second data flow concerns the raw signals, which should be sent to the cloud when a novel behavior is detected, and the extracted feature sets, which have to be sent into the cloud in any case. Given the high quantity of data, a real-time transfer is not feasible and perhaps not needed. Indeed, these data will serve for batch and offline analysis, which can be made when the machine is not working (for instance, during the night). In these cases, the data can be sent in pre-fixed moments, corresponding to the moments in which the edge storage memory reaches a certain level.

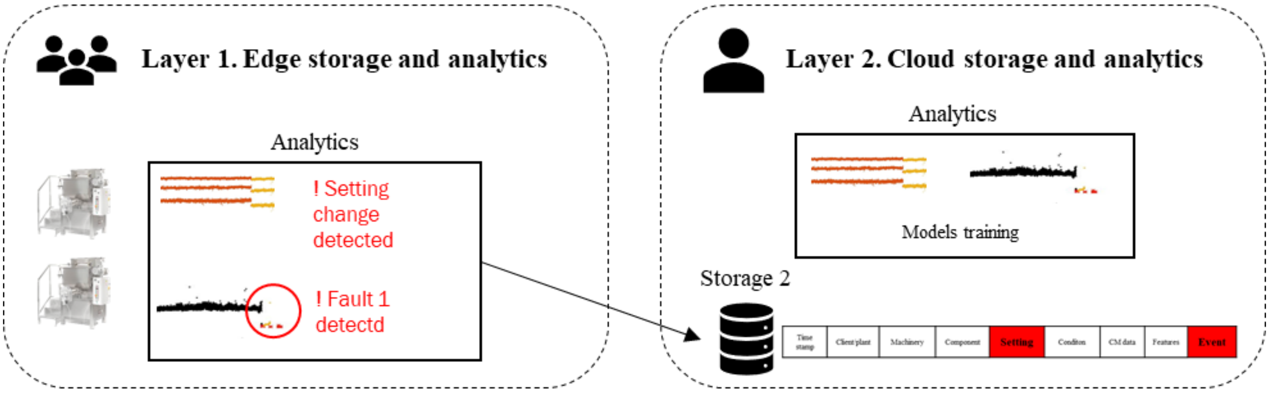


Figure 66 Edge-cloud infrastructure for Novelty Detection

4.4. The methodology design

This section presents the last strategic lever, acting on the methodology design. Besides the specific methods and models to perform fault diagnosis and novelty detection, One of the main issues of implementing a Predictive Maintenance system in industries is the design of a methodology including the characteristic elements of the system for a specific sector, e.g., aviation, industrial, wind energy, and others. From the methodological point of view, Li, Verhagen, and Curran (2018), (2020a), (2020b) provided a relevant contribution. They introduced an architecture definition process for Predictive Maintenance applications based on the 'RFLP' method, which consists in the definition of Requirements, and Functional, Logical, and Physical architectures, together with the framework, i.e., a layered structure of a system for a set of functions in a conceptual view. Similarly, the framework and the three architectures will be described, considering manufacturing industries as the application sector. Thanks to the collaboration with leading producers of automatic and grinding machines located in the well-known district of the Emilia Romagna region, it was possible to understand the benefits of a Predictive Maintenance methodology and the issues in implementing the existing ones. These aspects have been discussed largely in previous chapters, thanks to the described case studies. Therefore, this section focuses on defining stakeholders' requirements, the framework, and the description of the functional and logical architectures, which integrate the frameworks and logical connections resulting from chapters 3 and 4. In addition, the methodology also considers prognostics.

4.4.1. Stakeholders requirements definition

In this dissertation, the stakeholders are the industrial machine producers. They are interested in implementing a Predictive Maintenance methodology at machines installed at their clients' plants. It is an opportunity to offer a full maintenance service to their clients, optimize the spare parts supplying process, and improve the machine design phase. Besides, machinery users could benefit from a Predictive Maintenance system since they can reduce maintenance investments and improve productivity and availability. Stakeholders requirements, collected through interviews conducted on several machinery producers operating in the North of Italy, can be summarized as follows:

1. The system should allow the simultaneous monitoring of machines spread worldwide. Producers want to gather relevant data from installed equipment to expand their knowledge about the machinery's behavior during the actual functioning. Also, the remote monitoring of machinery the health conditions would avoid inspections; hence, maintenance scheduling and spare parts management are improved. This requirement implies building proper infrastructure for real-time data collection, transfer, and analysis;
2. The system should be able to detect novel behaviors and learn from new incoming data. Machine producers have no sufficient data for training ML models because of the scarce possibility to conduct tests in their plants. The main issue is the lack of knowledge of all possible working conditions. Besides, many components fail after years of functioning, making it hard for the real-time application of pre-build prognostic models.
3. The system should allow the collection of the data in a more structured way. Even if machine producers can get historical data from their clients, this data is not easy to process since the label (e.g., health status, implemented setting, and so on) is hardly available, and signals often present uncomprehensive trends. Low-frequency data, e.g., the setting of the machinery, and event data, e.g., the tuning made by operators or the anomalies that occurred during the machine functioning, should be collected together with high-frequency data. Hence, data is automatically labeled, and every event affecting the trend of collected data is recorded, speeding up the data pre-processing step.

4. The system should accomplish various functions based on a specific component, which means that functions should be performed separately depending on the analysis goal. A complex machine consists of many elements, e.g., suckers, or sub-systems, e.g., extruder, for which various analyses may be helpful. For example, an extruder's screw degrades over time, and the goal is to predict its RUL. Instead, the detachment of a sucker happens suddenly. Thus, the goal is to detect the anomaly. Hence, the real-time analysis should consider these two components separately to understand the problem's cause and, if possible, let the machine operator intervene as soon as possible. However, as long as these two components work simultaneously under the same operating conditions, machine producers are also interested in how the anomalies or failures affect each other, considering a given working condition.
5. The system should allow the CM data integration with historical data usually recorded in an external database (e.g., Computerized Maintenance Management System – CMMS), collecting work orders, spare parts management, and other information to develop a whole maintenance program. For machine producers that want to provide their clients with full-service maintenance, this is extremely important to get an internal optimization.

Table 50 shows the elements of the framework and architectures that can address each requirement. In the following sub-sections, each element will be investigated in detail.

Table 50 Framework and architectures connections with stakeholders' requirements

Stakeholders' Requirements	Framework	Functional Architecture	Logical architecture
Remote monitoring of several machinery	Distributed edge devices connected to a centralized cloud server	Features extraction, fault detection, diagnosis, and prognosis (edge DP, FDA, PA)	Transmission into the cloud of features and HIs at defined instants and event-data in real-time
Data collection of machinery settings	Streaming analysis at distributed edge devices	Novelty detection (OCR) to system-level features	Transmission into the cloud of system-level features with the associated label
Collection of structured and labeled data	Real-time computing at the edge and batch analysis into the cloud	Fault diagnosis and prognosis (edge and FDA and PA) to component-level features	Simultaneous streaming inference by trained models and anomaly and novelty detection on component-level features
System-level and component-level PHM	Model training into the cloud	System-level and component-level feature extraction (cloud DP)	Streaming extraction of features selected during the training phase
Integration of databases	Cloud server accessible by local databases	Fault diagnosis and RUL prediction (cloud FDA and edge PA)	Integration of information collected by the PHM system into local databases

4.4.2. *The framework*

This section proposes a framework to address the stakeholders' requirements. The framework consists of three layers: the edge storage and analytics layer, the cloud storage and analytics layer, and the knowledge integration layer, as shown in Figure 67.

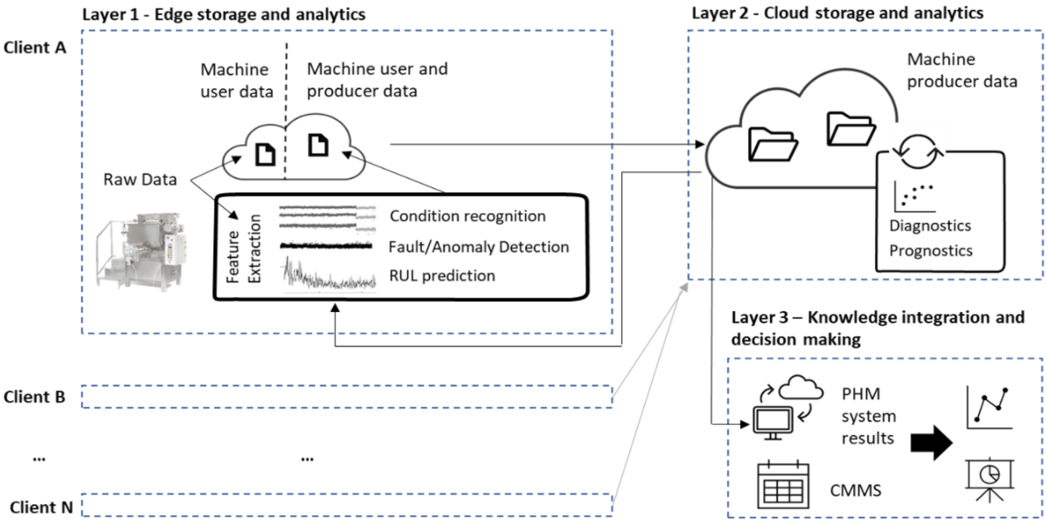


Figure 67 The framework for IIoT-based condition monitoring

The edge storage and analytics layer is a network of unlinked edge nodes installed close to the machines. Here, data processing (i.e., feature extraction and data reduction) and real-time analytics are performed (i.e., operating condition recognition, anomaly detection, RUL prediction), allowing the continuous monitoring of machines. In particular, the edge layer provides machine users with information regarding the health status of the equipment at any point in time. From the machine producers' point of view, this level also allows the collection of low-frequency and event data, addressing the first and third requirements. Indeed, the reduced amount of data obtained from streaming data processing allows producers to monitor several machines. When they need to know the health condition of a given component, they can access the information extracted at the edge node during the machine's functioning (i.e., features and HI values). Here, the current health condition and the implemented setting, the information related to anomalous behaviors, setting changes, and activities carried out by the operator on the machine are temporarily stored. Finally, sensitive information (such as the parameters that determine the machinery setting) is hidden from the producer since only processed data are accessible. At this level, there is no information aggregation. Results of data processing and real-time analytics are shown for each component of the machine so that the machine user can easily recognize where the anomaly is and intervene when necessary to restore the correct functioning. On the other side, the machine producer can select the component of interest. Also, as the setting is also recorded, the data related to each component in different operating conditions are separated, i.e., there is a label representing the implemented setting. This way, the fourth requirement is fulfilled.

The cloud storage and analytics layer represents the link between users and producers. In this layer, relevant information extracted at each edge node through streaming analysis is stored permanently. This information includes labels on the operating condition, component-level features, system-level features, Health Indicators, and RUL of each component. These data are integrated with historical data to train diagnostic and prognostic models, which will be applied at the edge during machinery functioning for real-time inference. At this level, which is only accessible by the machine producer, it is possible to aggregate the information depending on the analysis objective (component-level, system-level, client-level). In the last case, the producer uses this information to build an optimal maintenance plan in terms of time and human resources. In other words, this layer allows addressing the second producer requirement.

Finally, in the knowledge integration and decision-making layer, the PHM system integrates existing offline databases, e.g., CMMS. At this level, information related to the available resources, cost, and scheduled maintenance interventions on less critical components (e.g., preventive maintenance) are integrated with data collected through the PHM system to have a global vision of maintenance activities and their costs. The global vision results in optimized maintenance interventions and production scheduling, as long as the spare parts production can be triggered when necessary. Finally, the most critical components, or activities, can be identified, giving the possibility to better design both the machine and the maintenance service.

The definition of physical architecture, i.e., the set technologies for the communication between layers, is not the main topic of the present paper. However, one can consider the architecture model presented in (Bellavista et al. 2019) and (Bosi et al. 2020) as a possible guideline for the physical implementation of the proposed framework.

4.4.3. The functional architecture

The functional architecture includes all the activities within the framework that fulfill the defined requirements of machine producers.

The proposed functional architecture is shown in Figure 68. The functions have the same objectives as those introduced in (R. Li, Verhagen, and Curran 2020a), i.e., DA, DP, FDA, PA, HM. However, some modifications based on the previous observations and producers' requirements are proposed here.

First, the DP function is split into two sub-functions: the system-level feature extraction and the component-level feature extraction. The former deals with the extraction of system-level features, revealing the machine operating condition. The latter deals with the extraction of component-level features that can reveal the health condition of the components. In particular, component-level features can be used for two aims: diagnostics and prognostics. In the first case, the features allow establishing if a fault condition has occurred. In the second case, the features are HIs and allow to predict the RUL of the component. These sets of features change from one component to another. Thus, they need to be selected and computed for each critical machine component.

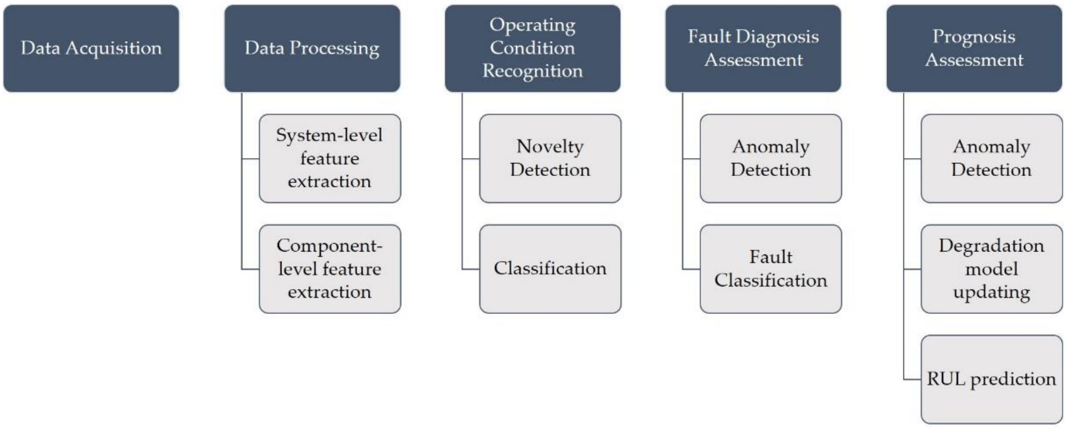


Figure 68 The functional architecture

Second, the Operating Condition Recognition (OCR) function is introduced between the DP function and the FDA. In general, the producer wants to know the machine’s working condition at any point in time. The operating condition can be known or unknown. If data collected during the machine functioning under one setting are available, it means that the setting is known, and the data can be used for diagnostics and prognostics models training. Thus, the so-built predictive models can be applied to make inferences on streaming data. Otherwise, the collected data cannot be associated with a specific setting if the operating condition is unknown. Hence, existing models cannot predict the operating condition with high accuracy. However, this data can be assigned to the same unknown group that corresponds to the same novel situation. Then, the new group can be used to train the existing models and include the new class. Hence, the OCR function aims to recognize the setting under which the machinery works at any point in time. It can be seen as a *novelty detection problem*. Its output is system-level information that determines the choice of existing predictive models. Thus, it can also be considered a function that automatically labels the collected observations as belonging to an existing class or novel behaviors.

Finally, the proposed functional architecture also includes anomaly detection into the FDA and PA functions. In the first case, an anomalous behavior could correspond to a sudden or unknown fault, while in the second case, the irregular behavior could mean that the component started a new HS. Thus, it can be considered as the point for starting the RUL prediction (Aydemir and Acar 2020).

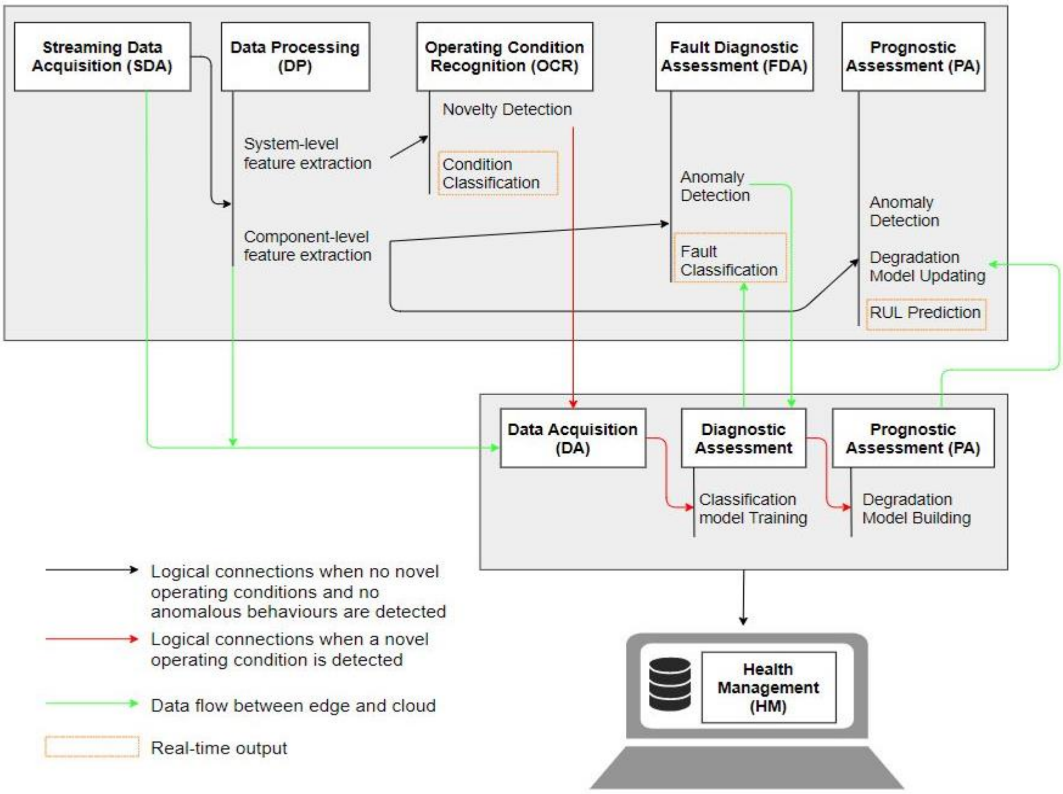
4.4.4. The logical architecture

The definition of logical architecture aims to explain the relationships between functions, and the data flows within the framework layers.

The proposed logical architecture is shown in Figure 69. The DA function is performed continuously at the edge to read, collect and temporarily store streaming data as the machine works. Then, when a signal segment of a pre-defined length is stored, the DP function is activated to extract the system-level features and the component-level features. The length of the signal segment depends on the task that will be performed, the maximum accepted latency of the inference, and the data transmission. Therefore, the time instant in which each sub-function of the DP is activated may be different. The activation of the DP function implies the extraction of relevant features from raw signal segments; then, the OCR function is triggered to assign the label “known” or “novel” to the system-level features.

The FDA and the PA function are activated if the operating condition is known. The FDA aims to recognize abrupt faults (for instance, the detachment of a sucker). Thus, a model trained into the cloud assigns to the current observation a label $l \in \left(Nominal, Fault_{1,} \dots Fault_{N}\right)$ among the $(N + 1)$ known fault classes. At the same time, the signal batch is eliminated from the storage memory of the edge device; in addition, in the FDA function, anomaly detection is also performed. It requires the same input as the classification models (i.e., component-level features). However, it aims is to detect unknown faults, i.e., faults not considered during the training of the models for diagnostics. The PA function aims to recognize degradations occurring during the machine functioning (for instance, the wear of a screw). Thus, a pre-built degradation model (which may depend on the implemented setting) is used to predict the future values of the HIs in order to compute the RUL of the corresponding component, based on pre-defined Failure Thresholds (FT). However, the degradation begins after several hours of machine functioning for many components; moreover, the computation of the future values of the HI and the RUL prediction would be expensive from the computational point of view. For these reasons, anomaly detection is also applied to the HIs to detect the degradation starting time. Thus, the RUL is computed only when an anomaly is detected.

A new cluster is created if the OCR function identifies a novel operating condition for the current observation. In this case, since the setting is novel, neither classification nor degradation models are available. However, anomaly detection can be applied to component-level features to detect anomalous behaviors that may represent a symptom of problems in the corresponding component.



Streaming Data Acquisition (SDA)
Data Processing (DP)
Operating Condition Recognition (OCR)
Fault Diagnostic Assessment (FDA)
Prognostic Assessment (PA)
Novelty Detection
System-level feature extraction
Condition Classification
Anomaly Detection
Anomaly Detection
Component-level feature extraction
Fault Classification
Degradation Model Updating
RUL Prediction
Data Acquisition (DA)
Diagnostic Assessment
Prognostic Assessment (PA)
Classification model Training
Degradation Model Building
Logical connections when no novel operating conditions and no anomalous behaviours are detected
Logical connections when a novel operating condition is detected
Data flow between edge and cloud
Real-time output
Health Management (HM)

Figure 69 The logical arhitecture

All the above-described activities are conducted at the edge. Now, the extracted feature vectors and the associated labels assigned by the streaming application of prediction models are transmitted to the cloud. Here, data related to known conditions (anomalies, fault classes, and RULs) are stored in a permanent database to help producers make decisions about maintenance interventions and spare parts supply. Instead, the data associated with novel conditions are used to re-train diagnostic and prognostic models. For this reason, the DA, FDA, and PA functions are also included in the cloud. Here, the DA function aims to acquire the data in different moments, depending on the data type. When the condition is known, the fault or anomaly detection triggers the transmission of the predicted fault class and the predicted RUL value to the cloud so that the producer can react as soon as possible. When the condition is unknown, or no anomalies or faults are detected in known conditions, the data transmission occurs at fixed moments, such as at the end of a shift or during the machine's set-up. The data transmitted in these cases are the extracted feature vectors and the associated label (belonging class and RUL).

4.5. References

Agelov, Plamen, and Xiaowei Zhou. 2006. “Evolving Fuzzy Systems from Data Streams in Real-Time.” *2006 International Symposium on Evolving Fuzzy Systems*, 29–35.

Ahmad, Subutai, Alexander Lavin, Scott Purdy, and Zuha Agha. 2017. “Unsupervised Real-Time Anomaly Detection for Streaming Data.” *Neurocomputing* 262: 134–47. https://doi.org/10.1016/j.neucom.2017.04.070.

Andonovski, Goran, Gašper Mušič, Sašo Blažič, and Igor Škrjanc. 2018. “Evolving Model Identification for Process Monitoring and Prediction of Non-Linear Systems.” *Engineering Applications of Artificial Intelligence* 68: 214–21. https://doi.org/10.1016/j.engappai.2017.10.020.

Angelov, Plamen, and Ronald Yager. 2011. “Simplified Fuzzy Rule-Based Systems Using Non-Parametric Antecedents and Relative Data Density.” *IEEE SSCI 2011: Symposium Series on Computational Intelligence - EAIS 2011: 2011 IEEE Workshop on Evolving and Adaptive Intelligent Systems*, 62–69. https://doi.org/10.1109/EAIS.2011.5945926.

Aydemir, Gurkan, and Burak Acar. 2020. “Anomaly Monitoring Improves Remaining Useful Life Estimation of Industrial Machinery.” *Journal of Manufacturing Systems* 56 (February): 463–69. https://doi.org/10.1016/j.jmsy.2020.06.014.

Bellavista, Paolo, Filippo Bosi, Antonio Corradi, Luca Foschini, Stefano Monti, Lorenzo Patera, Luca Poli, Domenico Scotece, and Michele Solimando. 2019. “Design Guidelines for Big Data Gathering in Industry 4.0 Environments.” *20th IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks, WoWMoM 2019*. https://doi.org/10.1109/WoWMoM.2019.8793033.

Bezerra, Clauber Gomes, Bruno Sielly Jales Costa, Luiz Affonso Guedes, and Plamen Parvanov Angelov. 2016. “An Evolving Approach to Unsupervised and Real-Time Fault Detection in Industrial Processes.” *Expert Systems with Applications* 63: 134–44. https://doi.org/10.1016/j.eswa.2016.06.035.

Bosi, Filippo, Antonio Corradi, Giuseppe Di Modica, Luca Foschini, Rebecca Montanari, Lorenzo Patera, and Michele Solimando. 2020. “Enabling Smart Manufacturing by Empowering Data Integration with Industrial IoT Support.” *2020 International Conference on Technology and Entrepreneurship, ICTE 2020*. https://doi.org/10.1109/ICTE47868.2020.9215538.

Breunig, Markus M., Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. 2000. “LOF: Identifying Density-Based Local Outliers.” *ACM SIGMOD Record* 29 (2): S93–S93. https://doi.org/10.1016/s0020-7292(09)60373-8.

Cariño, J. A., M. Delgado-Prieto, D. Zurita, A. Picot, J. A. Ortega, and R. J. Romero-Troncoso. 2020. “Incremental Novelty Detection and Fault Identification Scheme Applied to a Kinematic Chain under Non-Stationary Operation.” *ISA Transactions* 97: 76–85. https://doi.org/10.1016/j.isatra.2019.07.025.

Cariño, J. A., Miguel Delgado-Prieto, Jose Antonio Iglesias, Araceli Sanchis, Daniel Zurita, Marta Millan, Juan Antonio Ortega Redondo, and Rene Romero-Troncoso. 2018. “Fault Detection and Identification Methodology under an Incremental Learning Framework Applied to Industrial Machinery.” *IEEE Access* 6 (September): 49755–66. https://doi.org/10.1109/ACCESS.2018.2868430.

Chan, Felix T.S., Z. X. Wang, S. Patnaik, M. K. Tiwari, X. P. Wang, and J. H. Ruan. 2020. “Ensemble-Learning Based Neural Networks for Novelty Detection in Multi-Class Systems.” *Applied Soft Computing Journal* 93: 106396. https://doi.org/10.1016/j.asoc.2020.106396.

Datta, A., C. Mavroidis, and M. Hosek. 2007. “A Role of Unsupervised Clustering for Intelligent Fault Diagnosis.” *Industrial Engineering*, 1–9. http://proceedings.asmedigitalcollection.asme.org/proceeding.aspx?articleid=1600207.

Dhiman, Harsh S., Dipankar Deb, S. M. Muyeen, and Innocent Kamwa. 2021. “Wind Turbine Gearbox Anomaly Detection Based on Adaptive Threshold and Twin Support Vector Machines.” *IEEE Transactions on Energy Conversion* 8969 (c): 1–8. https://doi.org/10.1109/TEC.2021.3075897.

Dyer, Karl B., Robert Capo, and Robi Polikar. 2014. “Compose: A Semisupervised Learning Framework for Initially Labeled Nonstationary Streaming Data.” *IEEE Transactions on Neural Networks and Learning Systems* 25 (1): 12–26. https://doi.org/10.1109/TNNLS.2013.2277712.

Ester, Martin, Xiaowei Xu, Hans - peter Kriegel, and Jorg Sander. 1996. “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.” In *KDD 1996*, 226–31.

Faria, Elaine Ribeiro de, André Carlos Ponce de Leon Ferreira Carvalho, and João Gama. 2016. “MINAS: Multiclass Learning Algorithm for Novelty Detection in Data Streams.” *Data Mining and Knowledge Discovery* 30 (3): 640–80. https://doi.org/10.1007/s10618-015-0433-y.

Flynn, P J. 1999. “Data Clustering: A Review.” *ACM Computing Surveys* 31 (3): 264–323. https://doi.org/10.1145/331499.331504.

Garcia, K.D., M. Poel, J.N. Kok, and A.C.P.L.F de Carvalho. 2019. “Online Clustering for Novelty Detection and Concept Drift in Data Streams.” In *Progress in Artificial Intelligence*, 11805:448–59. Springer, Cham. https://doi.org/10.1007/978-3-030-30244-3_58.

Goldstein, Markus, and Seiichi Uchida. 2016. "A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data." *PLoS ONE* 11 (4): 1–31. https://doi.org/10.1371/journal.pone.0152173.

Gouriveau, Rafael, Emmanuel Ramasso, and Noureddine Zerhouni. 2013. "Strategies to Face Imbalanced and Unlabelled Data in PHM Applications." *Chemical Engineering Transactions* 33: 115–20. https://doi.org/10.3303/CET1333020.

Gu, Xiaowei, Plamen P. Angelov, and José C. Príncipe. 2018. "A Method for Autonomous Data Partitioning." *Information Sciences* 460–461 (September): 65–82. https://doi.org/10.1016/j.ins.2018.05.030.

Holmes, C. C., and N. M. Adams. 2002. "A Probabilistic Nearest Neighbour Method for Statistical Pattern Recognition." *Journal of the Royal Statistical Society. Series B: Statistical Methodology* 64 (2): 295–306. https://doi.org/10.1111/1467-9868.00338.

Hu, Yang, Piero Baraldi, Francesco Di Maio, and Enrico Zio. 2016. "A Compacted Object Sample Extraction (COMPOSE)-Based Method for Fault Diagnostics in Evolving Environment." *Proceedings of 2015 Prognostics and System Health Management Conference, PHM 2015*. https://doi.org/10.1109/PHM.2015.7380046.

———. 2017. "A Systematic Semi-Supervised Self-Adaptable Fault Diagnostics Approach in an Evolving Environment." *Mechanical Systems and Signal Processing* 88 (May): 413–27. https://doi.org/10.1016/j.ymssp.2016.11.004.

Inacio, M., A. Lemos, and W. Caminhas. 2015. "Evolving Fuzzy Classifier Based on Clustering Algorithm and Drift Detection for Fault Diagnosis Applications." *Mathematical Problems in Engineering* 2015. https://doi.org/10.1155/2015/368190.

Kasabov, Nikola, and Dimitar Filev. 2006. "Evolving Intelligent Systems: Methods, Learning, & Applications." *International Symposium on Evolving Fuzzy Systems,* 8–18. https://doi.org/10.1109/ISEFS.2006.251185.

Khalastchi, Eliahu, Meir Kalech, Gal A. Kaminka, and Raz Lin. 2015. "Online Data-Driven Anomaly Detection in Autonomous Robots." *Knowledge and Information Systems* 43 (3): 657–88. https://doi.org/10.1007/s10115-014-0754-y.

Khan, Samir, and Takehisa Yairi. 2018. "A Review on the Application of Deep Learning in System Health Management." *Mechanical Systems and Signal Processing*. Academic Press. https://doi.org/10.1016/j.ymssp.2017.11.024.

Larose, Daniel T. 2005. *Discovering Knowledge in Data: An Introduction to Data Mining*. Edited by John Wiley & Sons. https://doi.org/10.1002/0471687545.

Li, Rui, Wim J.C. Verhagen, and Richard Curran. 2020a. "A Systematic Methodology for Prognostic and Health Management System Architecture Definition." *Reliability Engineering and System Safety* 193 (August 2019): 106598. https://doi.org/10.1016/j.ress.2019.106598.

———. 2020b. "Stakeholder-Oriented Systematic Design Methodology for Prognostic and Health Management System: Stakeholder Expectation Definition." *Advanced Engineering Informatics* 43 (February): 101041. https://doi.org/10.1016/j.aei.2020.101041.

Li, Rui, Wim J C Verhagen, and Richard Curran. 2018. "A Functional Architecture of Prognostics and Health Management Using a Systems Engineering Approach." *Proceedings of the European Conference of the Phm Society*, no. Vol 4 No 1 (2018): 1–10.

Li, Xiang, Xu Li, and Hui Ma. 2020. "Deep Representation Clustering-Based Fault Diagnosis Method with Unsupervised Data Applied to Rotating Machinery." *Mechanical Systems and Signal Processing* 143. https://doi.org/10.1016/j.ymssp.2020.106825.

Li, Xiangli, Qiong Han, and Baozhi Qiu. 2017. "A Clustering Algorithm Using Skewness-Based Boundary Detection." *Neurocomputing* 275: 618–26. https://doi.org/10.1016/j.neucom.2017.09.023.

Li, Yuhua, Michael J. Pont, and N. Barrie Jones. 2002. "Improving the Performance of Radial Basis Function Classifiers in Condition Monitoring and Fault Diagnosis Applications Where 'unknown' Faults May Occur." *Pattern Recognition Letters* 23 (5): 569–77. https://doi.org/10.1016/S0167-8655(01)00133-7.

Liu, Fei Tony, Kai Ming Ting, and Zhi Hua Zhou. 2012. "Isolation-Based Anomaly Detection." *ACM Transactions on Knowledge Discovery from Data* 6 (1): 1–44. https://doi.org/10.1145/2133360.2133363.

Liu, Ruonan, Boyuan Yang, Enrico Zio, and Xuefeng Chen. 2018. "Artificial Intelligence for Fault Diagnosis of Rotating Machinery: A Review." *Mechanical Systems and Signal Processing* 108: 33–47. https://doi.org/10.1016/j.ymssp.2018.02.016.

Mirsky, Yisroel, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai. 2018. "Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection." In *Network and Distributed System Security Symposium 2018*. https://doi.org/10.14722/ndss.2018.23204.

Park, Pangun, Piergiuseppe Di Marco, Hyejeon Shin, and Junseong Bang. 2019. "Fault Detection and Diagnosis Using Combined Autoencoder and Long Short-Term Memory Network." *Sensors (Switzerland)* 19 (21): 1–17. https://doi.org/10.3390/s19214612.

Pimentel, Marco A.F., David A. Clifton, Lei Clifton, and Lionel Tarassenko. 2014. "A Review of Novelty Detection." *Signal Processing* 99: 215–49. https://doi.org/10.1016/j.sigpro.2013.12.026.

Prosvirin, Alexander, Bach Phi Duong, and Jong Myon Kim. 2019. "SVM Hyperparameter Optimization Using a Genetic Algorithm for Rub-Impact Fault Diagnosis." In *Advances in Intelligent Systems and Computing*, 924:155–65. Springer Singapore. https://doi.org/10.1007/978-981-13-6861-5_14.

Saucedo-Dorantes, Juan Jose, Miguel Delgado-Prieto, Roque Alfredo Osornio-Rios, and Rene De Jesus Romero-Troncoso. 2020. “Industrial Data-Driven Monitoring Based on Incremental Learning Applied to the Detection of Novel Faults.” *IEEE Transactions on Industrial Informatics* 16 (9): 5985–95. https://doi.org/10.1109/TII.2020.2973731.

Scholkopf, B., J. C Platt, J. Shawe-Taylor, A. J Smola, and R. C Williamson. 2001. “Estimating the Support of a High-Dimensional Distribution.” *Neural Computation* 13 (7): 1443–1471.

Sculley, D. 2010. “Web-Scale K-Means Clustering.” In *Proceedings of the 19th International Conference on World Wide Web, WWW ’10*, 1177–78. Raleigh, NC.

Spinosaa, Eduardo J., André Ponce de Leon F. de Carvalhoa, and João Gamab. 2009. “Novelty Detection with Application to Data Streams.” *Intelligent Data Analysis* 13 (3): 405–22. https://doi.org/10.3233/IDA-2009-0373.

Wang, Li-xin. 1992. “Fuzzy Systems Are Univers Approximators.” In *[1992 Proceedings] IEEE International Conference on Fuzzy Systems*, 1163–70. https://doi.org/10.1109/FUZZY.1992.258721.

Wang, Yi, Yi Ding, Xiangjian He, Xin Fan, Chi Lin, Fengqi Li, Tianzhu Wang, Zhongxuan Luo, and Jiebo Luo. 2021. “Novelty Detection and Online Learning for Chunk Data Streams.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43 (7): 2400–2412. https://doi.org/10.1109/TPAMI.2020.2965531.

Warren Liao, T. 2005. “Clustering of Time Series Data - A Survey.” *Pattern Recognition* 38 (11): 1857–74. https://doi.org/10.1016/j.patcog.2005.01.025.

Widodo, Achmad, and Bo Suk Yang. 2007. “Support Vector Machine in Machine Condition Monitoring and Fault Diagnosis.” *Mechanical Systems and Signal Processing* 21 (6): 2560–74. https://doi.org/10.1016/j.ymssp.2006.12.007.

Xu, Wei, Ling Huang, Armando Fox, David Patterson, and Michael I. Jordan. 2010. “Detecting Large-Scale System Problems by Mining Console Logs.” *ICML 2010 - Proceedings, 27th International Conference on Machine Learning*, 37–44.

Xu, Yan, Yanming Sun, Jiafu Wan, Xiaolong Liu, and Zhiting Song. 2017. “Industrial Big Data for Fault Diagnosis: Taxonomy, Review, and Applications.” *IEEE Access* 5 (July): 17368–80. https://doi.org/10.1109/ACCESS.2017.2731945.

Yadav, Samir S., and Shivajirao M. Jadhav. 2021. “Detection of Common Risk Factors for Diagnosis of Cardiac Arrhythmia Using Machine Learning Algorithm.” *Expert Systems with Applications* 163 (August 2020): 113807. https://doi.org/10.1016/j.eswa.2020.113807.

Yang, Zhe, Dejan Gjorgjevikj, Jianyu Long, Yanyang Zi, Shaohui Zhang, and Chuan Li. 2021. “Sparse Autoencoder-Based Multi-Head Deep Neural Networks for Machinery Fault Diagnostics with Detection of Novelties.” *Chinese Journal of Mechanical Engineering* 34 (1). https://doi.org/10.1186/s10033-021-00569-0.

Yang, Zhe, Jianyu Long, Yanyang Zi, Shaohui Zhang, and Chuan Li. 2021. “Incremental Novelty Identification from Initially One-Class Learning to Unknown Abnormality Classification.” *IEEE Transactions on Industrial Electronics* 0046 (c). https://doi.org/10.1109/TIE.2021.3101001.

Zhang, Jiafan, Qinghua Yan, Yonglin Zhang, and Zhichu Huang. 2006. “Novel Fault Class Detection Based on Novelty Detection Methods.” *Lecture Notes in Control and Information Sciences* 345: 982–87. https://doi.org/10.1007/11816515_124.

Zhu, Yue, Kai Ming Ting, and Zhi Hua Zhou. 2018. “Multi-Label Learning with Emerging New Labels.” *IEEE Transactions on Knowledge and Data Engineering* 30 (10): 1901–14. https://doi.org/10.1109/TKDE.2018.2810872.

5. Concluding Remarks

The primary purpose of this dissertation is to propose an IIoT-based framework for data collection, sharing, storage, and analysis that allows industries to monitor the health condition of machinery installed in their plants or their clients' plants. The health management system has to take advantage of the sensors deployed in the plant and the advances in ICT technologies for a data-driven approach and real-time monitoring and inference. It has to be self-learning, adapt itself to the dynamic nature of industrial environments, and produce interpretable results that support the maintenance decision-making process. The realization of such a system represents a challenging task for maintenance engineers given some technology, domain-specific, and methodological issues. First of all, the new advances in Industry 4.0 technologies and related national and international initiatives push companies to invest in transformation projects of their plants towards creating Smart Factories. The new advances in ICT and, particularly, Industrial AI represent an opportunity for companies to create value from data. However, although the IIoT and Industrial Big Data is seen as a strategic lever to increase the self-awareness and self-predictive abilities of machinery on the shop floor, the inherent characteristics of so-collected data and the distrust of companies to share information and implement data-driven solutions obstacle the extraction of valuable information and knowledge for Intelligent Maintenance. Moreover, the high investments required at the beginning of the project for plant transformations demand a complete, reliable, and generalizable methodology for Predictive Maintenance. However, the unlabelled and unbalanced datasets collected in traditional ways, the evolving environments in which machines operate, and the lack of data referred to all possible health conditions limit the use of Machine Learning. Consequently, engineers must deal with AI solutions that can learn during machine functioning, starting from a few data points.

Based on these statements, the research presented in this dissertation elaborates on two research questions that narrow down the set of potential approaches to the problem of creating a Fault Diagnosis and Detection system that takes advantage of Industry 4.0 technologies and data-driven solutions to extract information about the health condition of machinery in real-time and provide interpretable results to schedule maintenance activities.

The research activity is developed according to the research framework depicted in Figure 2, where the research questions are addressed by research levers, which are explored according to research topics. Each topic is explored according to a specific methodology. However, the overarching methodological approach to the research presented in this dissertation includes three fundamental aspects: the improvement of the quality and integrity of input data, the use of Machine Learning for analyzing the data and provide information and knowledge of the system, and the use of case-study deriving from real-world instances.

The outline of this dissertation is organized as follows. After a first introductive chapter, the second chapter of this thesis draws the state-of-the-art of the current industrial practices on Predictive Maintenance, introducing the most critical challenges for engineers and companies and illustrating the background of this thesis. Moreover, the scientific landscape on Predictive Maintenance in IIoT contexts is explored through a bibliometric analysis of the literature over the last decades.

Chapter 3 and chapter 4 illustrate the research activity. In particular, chapter 3 addresses *RQ 2* by acting on two strategic levers: data collection and sharing and data reduction. The first focuses on the characteristics of an IIoT infrastructure for data collection, based on distributed edge nodes connected to a central cloud, and on the organization of a database gathering all necessary data for fault diagnosis and detection. The second one focuses on data processing techniques to perform at the edge to reduce the amount of data to transfer and extract valuable information on the health condition of machinery.

Chapter 4 addresses *RQ 3* by exploring three strategic levers: acting on offline analysis, acting on streaming analysis, and acting on a logical architecture to join offline and online computing, illustrated in subsections 4.2, 4.3, and 4.4, respectively. The first lever includes Machine Learning algorithms for offline and batch fault diagnosis. The second lever includes Machine Learning algorithms for unsupervised and streaming anomaly detection and the approaches for Novelty Detection in evolving environments. The third level focuses on developing a unique methodology to address specific stakeholders' requirements. The methodology includes data collection, data processing, fault diagnosis, and novelty detection in IIoT environments and supports Original Equipment Manufacturers to implement a Maintenance system as a post-sale service.

Underpinned by the research questions, the explored research topics led to the development of practical and easy approaches, methods, and tools applied on different case studies, from which some general conclusions can be drawn. Figure 70 summarizes these research outcomes, where red lines highlight the interdependencies between the different works. For instance, the organization of a database for integrating multiple data sources, as proposed in section 3.4, supports the easy data analysis for fault classification and (diagnostics) and detection (novelty detection). The definition of the needed data allows building the framework for data collection, data storage, and data sharing. In particular, *RQ2* is addressed by the data collection and processing framework, which uses both edge and cloud computing to reduce the amount of data to share and store relevant information in a structured database. This database is accessible to the machine producer and can contain information on several machinery installed in several clients' plants, which can be organized depending on the objective of the analysis, e.g., scheduling maintenance interventions to one client or organizing the supplying of specific components installed in more machinery. *RQ3* is addressed by the integration of supervised and unsupervised learning for novelty detection in evolving environments and the integration of this activity into a wider methodology that also includes the framework for data collection and processing and other functionalities like RUL prediction, which has not been addressed in this dissertation but represents the fundamental aspect of a Predictive Maintenance system.

The following two subsections illustrate the research outcomes' theoretical, methodological, and practical contributions and the potential research developments.

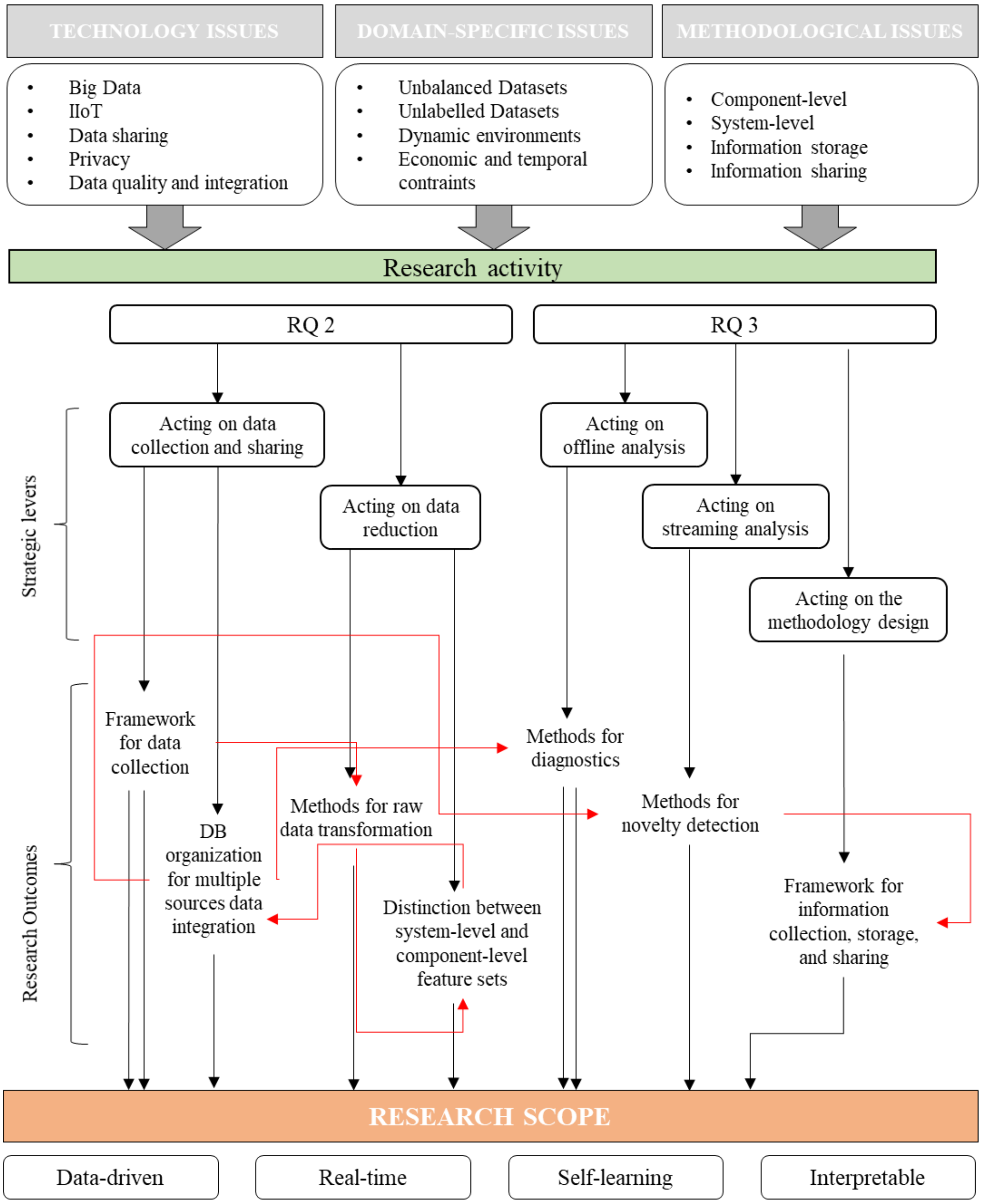

TECHNOLOGY ISSUES
DOMAIN-SPECIFIC ISSUES
METHODOLOGICAL ISSUES
• Big Data
• IIoT
• Data sharing
• Privacy
• Data quality and integration
• Unbalanced Datasets
• Unlabelled Datasets
• Dynamic environments
• Economic and temporal contraints
• Component-level
• System-level
• Information storage
• Information sharing
Research activity
RQ 2
RQ 3
Strategic levers
Acting on data collection and sharing
Acting on data reduction
Acting on offline analysis
Acting on streaming analysis
Acting on the methodology design
Research Outcomes
Framework for data collection
DB organization for multiple sources data integration
Methods for raw data transformation
Distinction between system-level and component-level feature sets
Methods for diagnostics
Methods for novelty detection
Framework for information collection, storage, and sharing
RESEARCH SCOPE
Data-driven
Real-time
Self-learning
Interpretable

Figure 70 Framework of the main contributions presented in this dissertation

5.1. Practical, theoretical, and methodological contributions

The initial bibliographic analysis included in the second chapter highlights the increasing adoption of a wider perspective in approaching the problem of IIoT-based real-time Predictive Maintenance. This analysis also draws the landscape of the current literature intending to identify the main research trends over the last decades, providing both scholars and practitioners with a snapshot of the state-of-the-art. The analysis's most important contribution concerns the methodology since the map of keywords proposed in Figure 9 constitutes a tool to explore the most debated research topics in the field, enabling researchers to identify potential gaps in the literature. The bibliographic analysis outcomes introduce and justify the choice of some of the research topics discussed in this dissertation.

The key elements of the research trends highlighted in Table 1, i.e., diagnostics and prognostics, the Machine Learning for Predictive Maintenance, and Internet of Things for Big Data Collection in the Industry 4.0 era, are further discussed in sections 2.2, 2.3, and 2.4, respectively.

In particular, the Internet of Things for Big Data Collection in Industry 4.0 has been explored since 2018. Although plenty of literature addresses this topic for Predictive Maintenance, its link with strategic levers of data collection, sharing, and reduction makes it essential and fundamental for all other activities, specifically in the case of Original Equipment Manufacturers interested in Maintenance as a post-sale service. The present research confirms the importance of remote monitoring of machinery, the benefits generated by the collection of multi-source data in a unique database, and the reduction of data at the edge through qualitative and quantitative results.

The research proposed in section 3.1 depicts the state-of-the-art of current IIoT infrastructures for Predictive Maintenance and current methods for signal processing and reduction. This work provides helpful insights for practitioners to enhance the effectiveness of data collection tools and data processing models to provide practitioners with general approaches and methods to address the problem of data collection and processing. Moreover, the application of these models to case studies conducted in sections 3.2 and 0 led to defining an edge-cloud architecture for data collection, processing, storage, and transfer. Firstly, a careful assessment of which information has to be extracted, stored, and transferred is fundamental. An initial evaluation of the current industrial practices for data collection and storage may lead to collecting as much condition monitoring data as possible, rather than shrinking the collection to high-quality data only. Condition monitoring data are fundamental for the analysis but cannot provide any information on the system health condition without environmental and event data. Findings from the analysis highlight how collecting condition monitoring data and contextual data to create labeled, balanced, and complete datasets is necessary.

Secondly, the data collection from multiple machinery and plants through an IIoT infrastructure allows leaving the raw signal processing to edge devices and sending only relevant features to the cloud (in the case of already known conditions), solving network issues, and facilitating model training. In this way, more valuable data can be collected from more machinery, increasing the availability of the data in multiple operating conditions. Indeed, the evaluation of the impact of data reduction models on data quantity reduction highlights the benefits of performing data reduction at the edge.

The analysis on the effectiveness of signal processing techniques performed in section 3.3.1 concluded that pre-processing, like signals combination or feature normalization, does not affect the classification accuracy. On the contrary, the specific technique for signal processing may lead to very different results. In particular, when dealing with vibration signals, time-frequency analysis increases the classification accuracy. However, it is not suitable for real-time applications because of its high execution times. Instead, time-domain features require a few milliseconds to be extracted while providing acceptable results. Similarly, the analysis on data reduction performed in sections 3.3.2 and 3.3.3 demonstrates that data reduction does not affect Machine Learning results in terms of classification accuracy, especially when the number of variables in the dataset is low, e.e., ten. However, they strongly affect the training time of classification models because of the high reduction of the dataset's weight.

In conclusion, the choice of the technique to use for signal processing and data reduction strongly depends on the specific case. There are some standard guidelines for some components, e.g., rolling bearings and gearbox. However, a deep investigation into the best models must be performed when dealing with more complex systems, such as sealing groups or extruders. For this reason, infrastructures for data collection and proper organization of the database are essential in industrial contexts. All these considerations led to conclude that creating a database in which all necessary information is stored in a structured and standard way for all available machinery is essential.

The research proposed in section 4.1 depicts the state-of-the-art fault diagnosis and novelty detection approaches. This work provides insights on how to deal with incremental environments in which known and novel behaviors must be recognized. In particular, the approaches have been classified into three categories, based on two dimensions, i.e., the learning paradigm and the necessity of training. Therefore, three different approaches can be distinguished, and several algorithms exist for each approach. Clustering-based approaches may have a training phase in which existing conditions are associated with a cluster and represented by its local parameters. Then, a novel behavior is detected if a point is considered too far or too dissimilar from existing cluster centroids.

However, the main advantage of these approaches is that they can be applied from scratch when no information is available on the belonging fault class. They adopt an incremental learning paradigm and are particularly suitable for streaming applications in both cases. On the contrary, classification-based approaches always require a pre-trained step, in which classification models are built based on the available fault classes. The relevant aspect of this approach is that probability-based classification models must be used because it is the probability class estimation to determine whether a point belongs to a known class or not. In addition, the classification model has to be re-trained each time a novel behavior is detected. Therefore, this approach is not incremental. Instead, an incremental classification-based approach for novelty detection is provided by Deep Learning models, which can automatically include the detected novel behavior.

The research topic of offline and batch fault diagnosis is addressed in section 4.2. It is the basis of any Predictive Maintenance system. This kind of analysis can be done if complete datasets are available. Indeed, it basically consists of training classification models using extracted and selected features associated with the class label as input. Different models have been applied to two case studies, and results are evaluated in terms of training accuracy, training time, and testing accuracy. Although other metrics may be used to evaluate the effectiveness of a model, these three metrics are sufficient to understand how the model works. Indeed, a low training accuracy indicates that the model cannot effectively learn the relationships between the features and the label. This result may depend either on the selected feature set or the selected model. A high training accuracy and a low testing accuracy indicate that the model overfits the data and is not generalizable. Therefore, it cannot apply to new datasets. Finally, high training times indicate that complicated relationships hold among the features and the label and may suggest reducing the dataset dimensionality or choosing another model.

The research topic of online and streaming analysis is addressed in section 4.3, in which both clustering-based and classification-based approaches are applied to the case studies. In the two cases, the considered evaluation metrics are different. In the case of clustering-based approaches, the ability to detect an anomaly and novel behavior, the latency, and the number of observations wrongly assigned to a cluster are considered. The precision and the false positive rate are evaluated in the second case. It emerged that clustering-based approaches require more parameters set by the user, and it is hard to connect data clusters with classes. On the other hand, classification-based approaches depend on the threshold on the PCE. Finally, Deep Learning for novelty detection has been investigated. These models can avoid manual feature extraction, which decreases classification models' dependency on the extracted and selected feature set. In addition, the ability to detect both known and novel behaviors, also with little data, and the low execution times make Deep Learning particularly suitable for streaming applications and evolving environments. The major problem associated with Deep Learning is the impossibility of automatically establishing the most relevant signals. Subsequent analysis can be conducted to understand which signal or signals contribute to novelty detection.

Finally, the research on the methodology design performed in section 4.4 led to the following conclusions that can be explained according to two perspectives. At the methodology level, identifying the stakeholders and the definition of their requirements and expectations is one of this dissertation's main contributions. Moreover, the definition of an edge-cloud-based framework allows the real-time health assessment of industrial machines and the simultaneous collection of high-frequency data, low-frequency data, and event-data from machinery installed in several clients' plants. At the architecture level, the integration of novelty detection into traditional architectures allows discovering unknown or different behaviors of the machines. In addition, the distinction between system-level features and component-level features allows knowing the setting implemented and occurred events for each observation. The system-level feature set reveals the machinery operating condition (i.e., the machine setting parameters implemented by the client during production) and thus are of great interest for the machine producer to know the possible working conditions and collect labeled data. The component-level features set reveal the health status and the degradation trend. Thus, it serves for anomaly detection, fault detection, and RUL prediction.

5.2. Future Developments

The findings of this dissertation prove that many challenges still exist in the field of data-driven Predictive Maintenance in evolving environments. Vast opportunities exist for further research on the presented research topics and unexplored research paths.

Focusing on the research topics included in this dissertation, the initial bibliometric analysis (see section 2.1) could be further expanded, including comparing the results obtained through other search engines (such as Web Of Science), to provide an overarching analysis of the existing literature.

Focusing on research topics included in this dissertations, the main challenges that have to be addressed by future research are related to the following aspects. First, data processing methods illustrated in sections 3.1.2 and 3.1.3 could consider alternative algorithms allowing the extraction of different sets of features each time a novel fault behavior is detected. Automatic and unsupervised feature learning methods, like Deep Learning models, may be directly applied to raw signals, avoiding the phase of signal processing. However, research should focus on making the outputs more interpretable. Second, concerning the supervised learning for offline diagnostics, the research could focus on automatically re-training existing models into the cloud when novel behaviors are detected. This implies reducing the data pre-processing and parameter setting activities, so that data, or information, collected from raw data can be directly used as input of diagnostics models. In addition, further studies may be conducted to define general rules that guide practitioners to choose Machine Learning models or Deep Learning models. Third, further research on streaming novelty detection in industrial contexts should consider reducing the influence of external factors on detection accuracy. Unlike offline and batch analysis, bad quality signals are directly processed from algorithms. However, both clustering-based and classification-based approaches are strongly affected by the quality of input signals, in which noise and outliers may occur because of the simultaneous functioning of several machines or sensor reading errors. For these reasons, more general and less feature-dependent approaches should be investigated. Finally, research on the edge-cloud infrastructure for data collection should consider the implementation of large-scale platforms to solve the issues related to the collection of data illustrated in section 3.2.

Focusing on research topics not included in this dissertation, future research should be conducted in the field of prognostics, which should deal with data availability problems. Indeed, prognostics and, ultimately, RUL prediction depend on run-to-failure trajectories' availability to build predictive models. However, as often pointed out in this dissertation, failure data are not easy to get for several reasons, including the impossibility of conducting fault simulations in the real world for safety issues. For this reason, the realization of digital twins and the application of transfer learning in an industrial context need to be investigated.

LIST OF APPENDED PAPERS

1. Calabrese, F.; Regattieri, A.; Bortolini, M.; Galizia, F.G.; Visentini, L. Feature-based multi-class classification and novelty detection for fault diagnosis of industrial machinery. *Applied Sciences (Switzerland)* 11 (20), 9580 (2021)
2. Del Buono, F.; Calabrese, F.; Baraldi, A.; Paganelli, M.; Regattieri, A. Data-driven predictive maintenance in evolving environments: a comparison between machine learning and deep learning for novelty detection. Smart Innovation, Systems and Technologies, 262 SIST, pp. 109–119, 2022. 8th International Conference on Sustainable Design and Manufacturing, KES-SDM 2021Virtual, Online15 - 17 September 2021
3. Calabrese, F.; Regattieri, A.; Bortolini, M.; Galizia, F.G. Fault diagnosis in industries: how to improve the health assessment of rotating machinery. Smart Innovation, Systems and Technologies, 262 SIST, pp. 257 – 266, 2022. 8th International Conference on Sustainable Design and Manufacturing, KES-SDM 2021Virtual, Online15 - 17 September 2021
4. Calabrese, F., Regattieri, A., Bortolini, M., Gamberi, M., Pilati, F. Predictive Maintenance: A Novel Framework for a Data-Driven, Semi-Supervised, and Partially Online Prognostic Health Management Application in Industries. *Applied Sciences (Switzerland)* 11(8), 3380 (2021)
5. Calabrese, F., Regattieri, A., Botti, L., Mora, C., Galizia, F.G. Unsupervised fault detection and prediction of remaining useful life for online prognostic health management of mechanical systems. 2020, *Applied Sciences (Switzerland)* 10(12), 4120, pp. 4120
6. Calabrese, F., Regattieri, A., Pilati, F., Bortolini, M. Streaming-based Feature Extraction and Clustering for Condition Detection in Dynamic Environments: An Industrial Case. *Proceedings of the 5th European Conference of the Prognostics and Health Management Society 2020*
7. Calabrese, F., Ferrari, E., Lelli G., Regattieri, A. From raw data to information for a continuous supervision of machinery in dynamic industrial environment: a case study. *XXV Summer School "Francesco Turco" – Industrial Systems Engineering, 9-11 September 2020, Bergamo, Italy*

8. Calabrese, F., Regattieri, A., Botti, L., Galizia, F.G. Prognostic health management of production systems. New proposed approach and experimental evidences. 2019, *Procedia Manufacturing* 39, pp. 260-269

9. Calabrese, F., Gamberi, M., Margelli, S., Pilati, F., Regattieri, A. Data-driven prognostics: from an offline and supervised analysis to an innovative, online and unsupervised methodology. *XXIV Summer School "Francesco Turco" – Industrial Systems Engineering, 11-13 September 2019, Brescia, Italy*. Proceedings of the Summer School Francesco Turco 2019, pp. 74-80

10. Calabrese, F., Casto, A., Regattieri, A., Piana, F. Components monitoring and intelligent diagnosis tools for Prognostic Health Management approach. *XXIII Summer School "Francesco Turco" – Industrial Systems Engineering, 12-14 September 2018, Palermo, Italy*. Proceedings of the Summer School Francesco Turco 2018, pp. 142-148