Prediction of the remaining useful life of aircraft components with deep learning methods

Anass Akrim

Université Fédérale

Toulouse Midi-Pyrénées

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)

Présentée et soutenue le 08/12/2022 **par :**

Anass AKRIM

Prédiction de durées de vie restante de composants aéronautiques par des approches de deep learning

JURY

JEAN-MARC BOURINET	Maître de conférence HDR - SIGMA Clermont	Rapporteur
EMMANUEL RAMASSO	Maître de conférence HDR - Institut FEMTO-ST	Rapporteur
BRUNO CASTANIER	Professeur des Universités - Université d'Angers	Président
MIHAELA JUGANARU	Maître Assistant - EMSE	Examinatrice
KAMAL MEDJAHER	Professeur des Universités - ENIT	Examinateur
CHRISTIAN GOGU	Professeur - ISAE-SUPAERO	Directeur de Thèse

École doctorale et spécialité :

EDAA : Mathématiques et Applications

Unité de Recherche :

Institut Clément Ader (UMR 5312) - Modélisation des Systèmes et Microsystèmes Mécaniques (MS2M)

Directeur(s) de Thèse :

Christian GOGU - Directeur de thèse - Professeur (ISAE-SUPAERO),

Rob VINGERHOEDS, Co-directeur de thèse - Professeur (ISAE-SUPAERO), et

Michel SALAÜN, Co-encadrant - Professeur (ISAE-SUPAERO)

Rapporteurs :

Jean-Marc BOURINET et Emmanuel RAMASSO

Acknowledgements

It is with great pleasure and pride that I finish this thesis. This thesis has been a rich experience for me that cannot be completed without thanking the people who have supervised, helped and supported me during the last three years.

First of all, I would like to thank the members of the jury for honouring me with the examination of this thesis and thus judging its scientific content. I would like to thank Dr. Jean-Marc Bourinet and Dr. Emmanuel Ramasso for agreeing to act as rapporteurs; thanks to Pr. Bruno Castanier, Dr. Mihaela Juganaru and Pr. Kamal Medjaher for their participation in my thesis jury as examiners.

My deepest thanks go to my supervisors. To my thesis director Pr. Christian Gogu for having welcomed me in his research team and given me his trust by allowing me to carry out my PhD in a high quality working environment. His countless advices, his scientific rigour and his support, in particular his availability during the Covid period, allowed me to overcome the difficult moments and complete this thesis work. To my thesis co-director Pr. Rob Vingerhoeds for his advice, experience and his rigorous remarks throughout the thesis. I would also like to thank you for your ideas which helped me to progress in my thesis when I faced with scientific obstacle. Your advices has been utterly priceless and genuinely helped me become the researcher I am today. To my supervisor Pr. Michel Salaün who supported and encouraged me a lot during the difficult moments. You have always been able to provide simple and comforting answers to my many questions, helping me many times to find solutions to overcome obstacles and bottlenecks. Once again, thank you to all three of you for pushing me to do my best, for also giving me the opportunity to teach and for giving me the opportunity to attend and present my work at various national and international conferences. It has been a pleasure to work with you.

I would like to thank Paul, Thomas and Brondon for their professionalism and our fruitful collaboration, as well as Pedron Julien for his help in using the ISAE Pando supercomputer and for all the technical and computer tricks he showed me at the beginning of the thesis. I would also like to thank all the administrative staff of the doctoral school, the laboratory of Institu Clément Ader and the ISAE-Supaero engineering school. I particularly thank Maryse Herbillon, Myriam Boyer and Odile Riteau for their warm welcome, their kindness and their professionalism.

Thanks to my colleagues (former and new PhD students, post-docs, engineers) for their help, their support and with whom I spent excellent moments : Aitor, Ali, Adama, Florent, Sébastien, Juan, Sophia, Laetitia, Silvio, Guillermo. My warmest thanks to my colleagues in the office : Tohme, Morgan and Nasrine. Thank you for all our different exchanges and brainstorming sessions on our respective subjects, for the moral support, and for the coffe-breaks. Thank you to my long-time friends who have remained in my entourage despite the different paths they have

taken. To all those who have assisted me in one way or another in the realisation of this work, I would like to say thank you.

And finally, I would like to thank my family, for believing in me and encouraging me, for always willing to go the extra mile for me, to whom I dedicate this work. I was also lucky to have the support of my parents and my sister who, despite the distance that separates us, lived this thesis with me day after day. Thank you for your unfailing support, for your prayers, for your encouragement and for your love: *family is priceless*.

Abstract — The remaining useful life (RUL) of a structure or system is the operating time remaining before it can no longer successfully perform its required functions and must be replaced. Predicting the RUL of a component based on available data (notably sensors data) is the main objective of prognostics tasks (or failure prognostics). As more data becomes available in many engineering domains, including in aerospace, there is a recent surge of interest in using Deep Learning (DL) for prognostics in these fields. Indeed, these approaches have been quite successful in other domains when trained on large datasets. However, one of the major challenges for DL techniques resides in the difficulty of obtaining large amounts of labelled data for their training (i.e. data scarcity). This research aims at proposing possible solutions to address this challenge. In this thesis, fatigue damage prognostics problems are considered. Indeed, for many mechanical structures and notably aeronautic structures, fatigue damage is one of the major modes of failure (e.g. fuselage panels subjected to variable loadings during flight). In the first part of the thesis, a framework and code for synthetically generating large data sets for a realistic fatigue damage prognostics problem are proposed. The objective of this development is to facilitate the comparative evaluation of the latest DL algorithms in the research field, in particular in the aerospace domain. A first step in this direction was then taken by comparing different DL techniques in a variety of settings on this fatigue damage prognostics problem. Furthermore, while labelled data is lacking, the availability of unlabelled data is increasing due to the advancements in sensing technologies. Exploiting unlabelled data during training has therefore become a major goal in ML. Hence, an emerging machine learning technique is investigated in this work: Self-Supervised Learning (SSL). This approach has already proven its worth in fields such as Natural Language Processing or Image Processing. In this research, this approach was developed and implemented on the previously mentioned fatigue damage prognostics problem, on which it showed promising results, notably when large amounts of unlabelled data and only limited labelled data are available.

Keywords: *Prognostics and Health Management (PHM), Remaining Useful Life (RUL), Deep Learning (DL), Data Scarcity, Self-Supervised Learning (SSL)*

Résumé — La durée de vie restante (ou Remaining Useful Life – RUL en anglais) d'une structure ou d'un système est le temps opérationnel restant avant qu'il ne soit plus capable d'accomplir avec succès les fonctionnalités requises et devrait ainsi être remplacé. Prédire la durée de vie restante d'un composant sur la base des données disponibles (notamment des données de capteurs) est l'objectif principal des tâches de pronostic (ou pronostic de défaillance). Comme de plus en plus de données deviennent disponibles dans de nombreux domaines de l'ingénierie, y compris dans le domaine de l'aérospatial, il y a un récent essor en intérêt pour l'utilisation des techniques d'apprentissage profond (ou Deep Learning – DL en anglais) pour le pronostic dans ces domaines. En effet, ces approches ont été souvent couronnées de succès dans d'autres domaines lorsqu'entrainées sur des grandes bases de données. Cependant, l'un des principaux défis de ces techniques réside dans la difficulté d'obtenir de grandes quantités de données étiquetées pour leur entrainement. Les travaux de recherche de cette thèse visent à proposer des solutions potentielles pour répondre à ce défi. Dans le cadre de ces travaux, les problèmes de pronostic d'endommagement par fatigue sont considérés. En effet, pour de nombreuses structures mécaniques et notamment les structures aéronautiques, l'endommagement par fatigue est un mode de défaillance majeur (par exemple pour les panneaux de fuselage soumis à des charges variables pendant le vol). Dans la première partie de la thèse, un cadre algorithmique et un code associé sont proposés, permettant de générer de grandes bases de données synthétiques pour un problème réaliste de pronostic d'endommagement par fatigue. L'objectif de ce travail est de faciliter l'évaluation comparative des derniers algorithmes de DL dans ce domaine de recherche, en particulier dans le secteur aérospatial. Un premier pas dans cette direction a ensuite été fait en comparant différentes approches de DL dans plusieurs configurations sur ce problème de pronostic de l'endommagement par fatigue. Par ailleurs, alors que les données étiquetées sont rares, la disponibilité des données non étiquetées augmente en raison des progrès des technologies de capteurs. L'exploitation des données non étiquetées pendant la phase d'entrainement est donc devenue un objectif majeur de l'apprentissage automatique. Par conséquent, une technique d'apprentissage automatique émergente est étudiée dans cette thèse : l'apprentissage auto-supervisé (ou Self-Supervised Learning – SSL en anglais). Cette approche a déjà fait ses preuves dans des domaines tels que le traitement de texte ou traitement d'images. Dans ces recherches, cette approche a été développée et implémentées dans le cadre du problème de pronostic de l'endommagement par fatigue mentionné précédemment, sur lequel elle a montré des résultats prometteurs, notamment lorsqu'une grande quantité de données non étiquetées et seulement peu de données étiquetées sont disponibles.

Mots-clés: *Pronostic de défaillance, Durée de Vie Restante, Apprentissage Profond, Rareté de données étiquetées, Apprentissage auto-supervisé*

Contents

List of Figures

List of Tables

Table of acronyms

AI	*Artificial Intelligence*
CNN	*Convolutional Neural Network*
DL	*Deep Learning*
DNN	*Deep Neural Networks*
EOL	*End of life*
ERM	*Elman Recurrent Network*
GRU	*Gated Recurrent Unit*
LSTM	*Long short-term memory*
ML	*Machine Learning*
MLP	*Multilayer Perceptron*
NLP	*Natural Language Processing*
PHM	*Prognostics and Health Management*
RNN	*Recurrent Neural Network*
RUL	*Remaining Useful Life*
SSL	*Self-Supervised Learning*
TCN	*Temporal Convolutional Network*
TL	*Transfer Learning*

CHAPTER 1

Introduction

"There's a way to do it better. Find it."

Thomas Edison

Content

This chapter intends to provide a general overview of this thesis. It aims at introducing the general context and the motivations for this research. The scope of the thesis will also be defined, which is developed around two main topics: failure prognostics and Deep Learning (DL). These two topics will be defined in this chapter, followed by the first emerging research questions.

1.1 General Context

Engineering structures or systems are subject to the risk that they are no longer able to perform the desired functions for which they were designed. Equipment that is no longer capable of performing the functions for which it was designed is said to become *defective* and have reached *failure*. Nowadays, with the increasing complexity of industrial systems such as aircraft, there are many instances that can lead to failure, potentially causing considerable damage to property, the environment and people. Beyond its function, it is therefore essential that the design and use of

the system be extended to include health management, which is in the scope of Prognostics and Health Management or PHM domain.

1.1.1 Prognostics and Health Management (PHM)

Prognostics and Health Management (PHM) is a field of research and application, making use of past and present available information of an equipment in order to detect its degradation, diagnose its faults, predict and manage its failures [1], [2]. PHM aims to establish a maintenance policy that is able to prevent structural or system failures and, consequently, minimise the unexpected downtime of complex systems. The French AFNOR standard NFX 13-306 [3] defines the term 'maintenance' as the *process of maintaining or restoring a system to a specified state in which it can perform its required functions*. Nevertheless, the maintenance process must take into account many factors such as quality, safety, the environment, cost, etc. Hence, global performance requirements are leading various industries to optimize their maintenance strategy in order to strengthen their ability to anticipate failures due to degradation. PHM implementation can improve the efficiency of maintenance support [4], optimize the maintenance plan [2], help the industry to balance the safety standard and economic profit [5].

Prognostics and Health Management is described by some authors as a rapidly growing discipline that is interested in studying the failure mechanisms of real systems in order to better manage the use of information on equipment operating conditions [6]. It is a vast topic with two main scopes:

1. **Prognostics** (or failure prognostics), which has as main goal to predict the remaining time until failure based on an estimate of the current state of the system.
2. **Diagnostics** (or failure diagnostics), which corresponds to the localization and identification of the causes of anomalies or failures.

These two terms, prognostics and diagnostics, refer to the maintenance strategy that aims at anticipating failures of equipments based on their condition. The two processes are essential in making maintenance decisions, as they help to determine the operating state of the system and to predict its future state (see Figure 1.1).

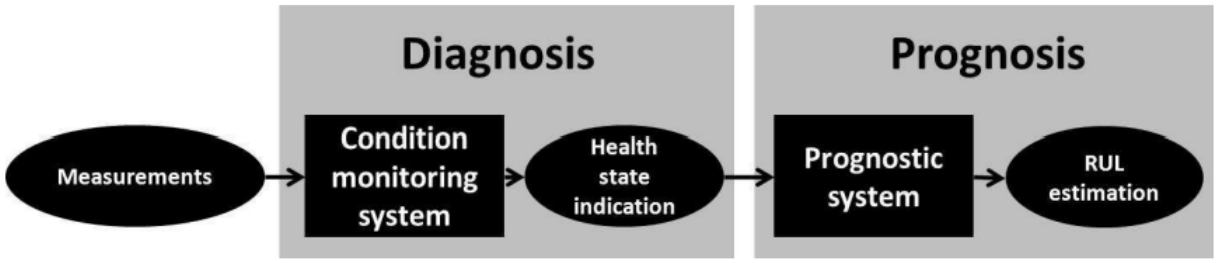


Figure 1.1: A diagnostic-prognostics framework proposed by Gola et al. [7] in order to develop an optimal maintenance scheduling strategy of choke valves undergoing erosion.

The current thesis will focus only on the prognostics aspects.

1.1.2 Failure prognostics

There are several definitions of the term "prognostics" in the literature. According to ISO 13381-1 [8], the prognostics of failure is defined as: *"estimation of time to failure and risk for one or more incipient failure modes"*. From a maintenance point of view, it is necessary to predict the time from which the component or system changes from an operational state (in good working order) to a non-operational state, i.e. in abnormal operation. This operating time before failure is commonly referred to as Remaining Useful Life (RUL) [9].

As mentioned earlier, the main objective of prognostics is to estimate the RUL of a system by projecting the evolution of its health status in the future at an early stage of degradation. In practice, the RUL of a system is the *"remaining time before system health falls below a defined failure threshold"* [8], *i.e.* can no longer successfully perform its required functions and must be repaired or replaced. It is calculated from the difference between the current time t and the time of the predefined failure threshold T_f, as illustrated in Figure 1.2. Such a calculation is generally based on:

1. Several pieces of information, such as an estimate of the current state of the system using a monitoring or diagnostics system, a degradation model, the definition of a degradation threshold considered as failure or end of life, etc;
2. Past condition profile $Z(t)$ up to the current time t, a possible knowledge of the evolution of the future operating conditions: environment, context of use, etc.

Hence, the remaining useful life at time t can be defined as a conditional random variable [10] such that:

$$RUL_t = T_f - t | T_f > t, Z(t)$$

where T_f is the time to failure, t the current time and $Z(t)$ the past condition profile up to the current time.

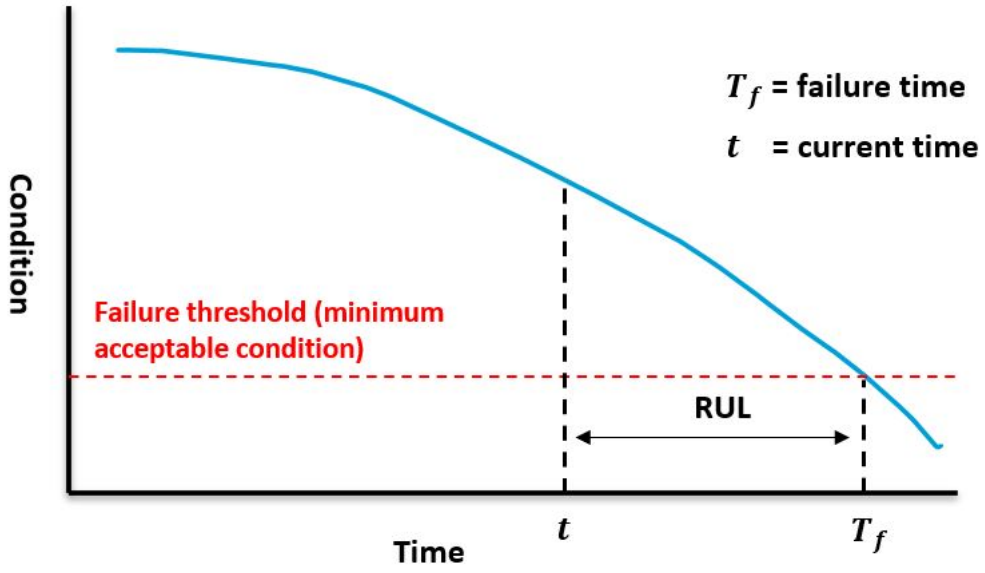


Figure 1.2: Remaining Useful Life illustration. Note that in this figure, the equipment health state has evolved under normal conditions, *i.e.* no anomalies have been identified during its operational lifetime. *Source*: reproduced based on [11].

1.2 Motivations and Objectives of the Research

Failure prognostics is becoming increasingly important in the field of operational safety and is one of the major challenges of the moment in all areas of industry. In this thesis, fatigue damage prognostics problems are considered. Indeed for many mechanical structures and notably aerospace structures, fatigue damage is one of the major modes of failure. Fatigue damage is defined as one of the major life-limiting factors for many structural components subjected to variable loadings in service (e.g. aircrafts during flight). Fatigue cracks are the cause of various failures in domains such as aerospace, develop gradually and progressively grow to a critical crack size, leading to structural or system failure if not early identified (see Fig. 1.3). Therefore, fatigue monitoring and prediction of fatigue life in structures, *i.e.* RUL estimation, represents one of the major challenges to be solved for paving the way towards predictive structural maintenance.

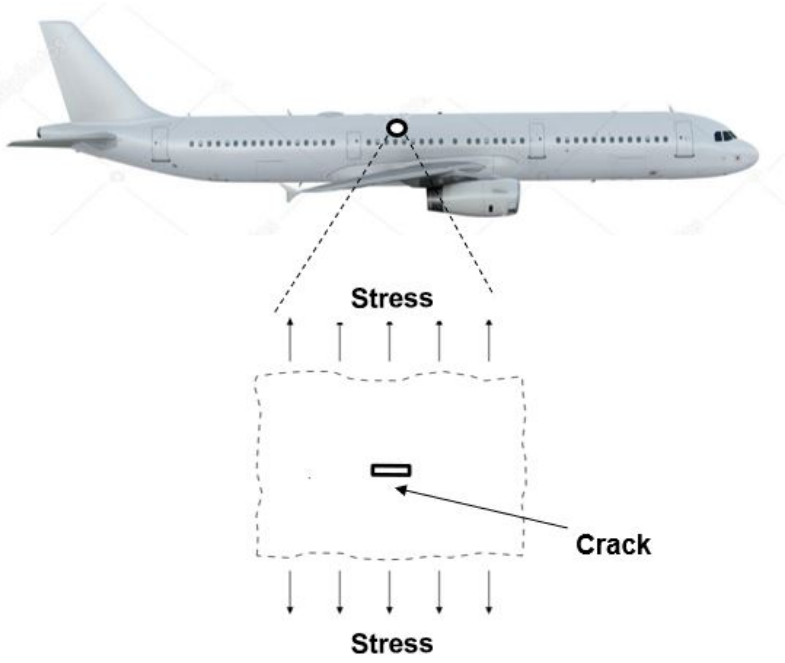


Figure 1.3: An example of a fatigue damage problem in aeronautics.

1.2.1 Failure prognostics methods

Numerous tools and methods for failure prognostics have been proposed over the past decades, and the literature is rich in papers providing a classification of prognostics approaches [12]–[14]. It appears that prognostics methods generally differ by the type of application considered, whereas the specific tools used depend mainly on the nature of the available data (e.g. data collected from sensors, vibrations signals, etc.) and knowledge. Hence, prognostics models can be classified into three major categories, and this classification tends to achieve consensus within the PHM community: Knowledge-based, Physics-Based and Data-Driven models. Among prognostics approaches, Data-Driven models have gained more and more attention in the PHM community, especially the latest Machine Learning (ML) techniques (notably Deep Learning techniques) [15]–[17].

Machine Learning (ML) is a branch of computer science that deals with the development of algorithms and statistical models for building systems that can learn from data and discover useful insights without being programmed to know where to look for them, thus requiring no prior assumptions about the underlying relationships between the variables. It is related to Artificial Intelligence (AI), the broader field of research that aims to create machines that can imitate human cognitive abilities such as natural language understanding, problem-solving, and decision-making. Deep Learning (DL) is a subset of Machine Learning (see Fig. 1.4), which consists of complex architectures of interconnected layers of artificial neurons able to process large amounts of data

and perform specific tasks such as image or speech recognition. It has become a major and rapidly growing research direction, redefining state-of-the-art performances in a wide range of areas in recent years [18].

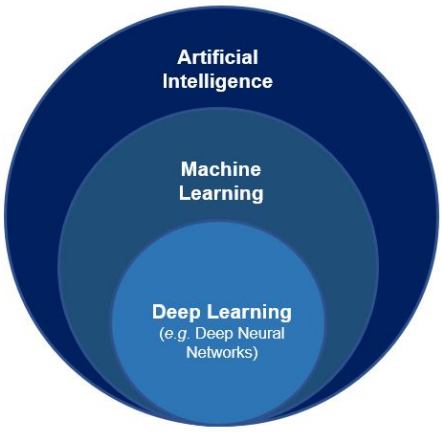
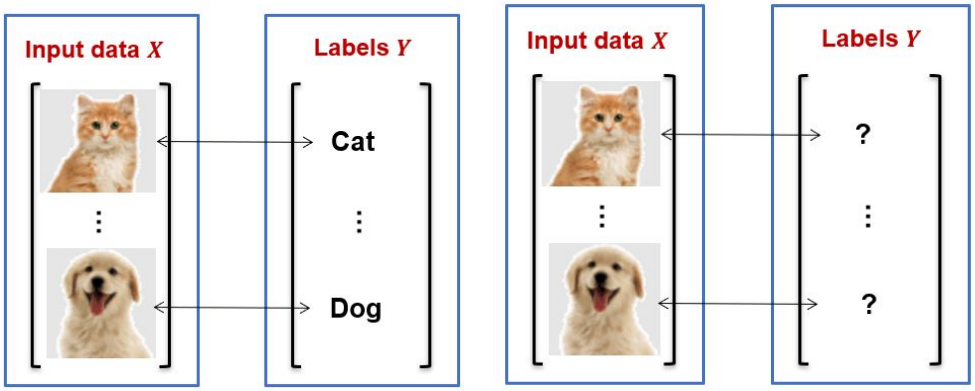


Figure 1.4: Deep Learning (DL), a sub-field of Machine Learning (ML) and Artificial Intelligence (AI).

As illustrated in Fig.1.5, two types of learning process are generally considered in Machine Learning: 1) *Supervised* Learning and 2) *Unsupervised* Learning.



(a) Supervised Learning: labelled data. (b) Unsupervised Learning: unlabelled data.

Figure 1.5: Main types of machine learning processes.

In supervised learning, the data is already labeled, and the model seeks to learn the relationship between the observation and its label. The model is trained to predict the output (*i.e.* label) based on the input data (*i.e.* observations). On the other hand, in unsupervised learning, the data is not labeled and the model is trained to find the hidden structure or pattern in the data without any prior knowledge of the output. The model tries to discover the underlying distribution of the

data. Supervised learning often yields better predictions than unsupervised learning because it has the information about the correct output. However, it requires a large amount of labeled data for training, which may not always be available in some domains such as in predictive maintenance. In contrast, unsupervised learning does not require labeled data but it may be more challenging to extract useful information from the data.

In prognostics tasks, a label can constitute the RUL at each time step of measurements. Fig. 1.6 illustrates the application of the latest ML techniques in prognostics tasks.

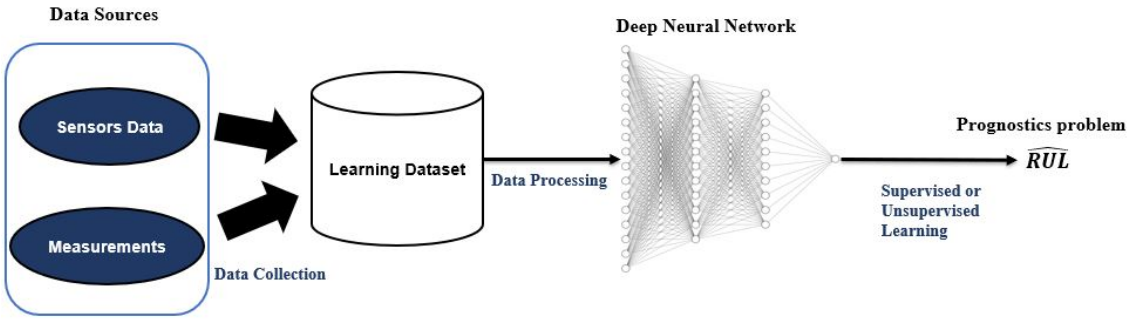
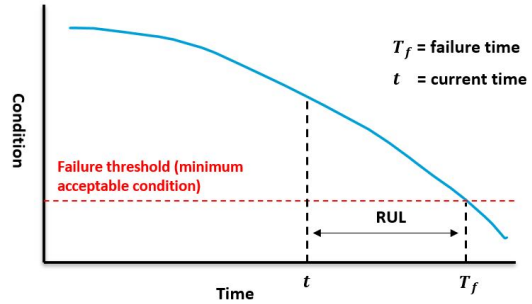
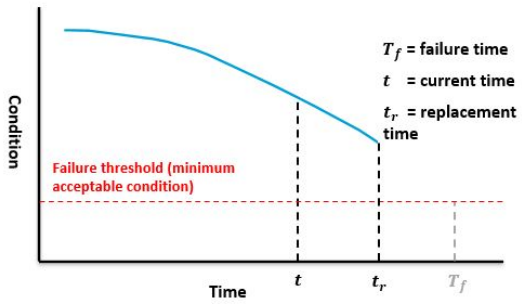


Figure 1.6: Deep Learning in prognostics tasks.

Nevertheless, as structures can be replaced before reaching failure, the true RUL at each timestep is unknown, resulting in unlabelled data. Thus, the lack of available labelled data (*i.e.* data scarcity) is becoming one of the major present challenges in PHM [14], [19] (see Fig. 1.7). Note that in Fig. 1.7, the equipment has functioned under normal conditions, with no anomalies detected during its operational lifetime.



(a) Labelled structure in prognostics tasks.



(b) Unlabelled structure in prognostics tasks.

Figure 1.7: Illustration showing a labelled structure in prognostic tasks (*i.e.* reaching failure and therefore the RUL is known at each time step t) compared to an unlabelled structure (*i.e.* replaced before reaching failure).

1.2.2 Research Statements

As mentioned earlier, there are a large number of approaches to fulfilling the prognostics tasks in a predictive maintenance system, and there are no clear guidelines to help the practitioner select the appropriate model or algorithms that can fulfil these tasks. Exploring the solution space to determine the appropriate technique can then be complex and time consuming. This thesis first aims to provide a comprehensive guide to the reader by enabling a more efficient way of exploring the solution space, particularly for prognostics problems related to fatigue damage. Note that in this thesis, there will be a special focus on the latest Machine Learning techniques in the prognostics field, notably Deep Neural Networks (DNN). Particular attention will be paid to the data scarcity issue in the context of Deep Learning, and another aim of this thesis will be to provide a novel approach to address this issue.

Before delving further into the subject, it is necessary to have an understanding of the scope of prognostics research in predictive maintenance itself and the current trends, as well as the associated challenges. Thus, the following research questions (RQs) are proposed to guide the state-of-the-art study in this topic:

1. What are the current trends in prognostics for predictive maintenance ?
2. Which data-driven Machine Learning techniques are most commonly used to address failure prognostics, and what are their main key strengths and limitations ?
3. What are the existing related challenges and current research areas?

At the end of the state-of-the-art study, the research questions are refined according to the study findings and the initial motivation of this research (see Chapter 2).

1.3 Structure of the thesis

The purpose of this section is to present the structure of the thesis which is organised in five chapters. It also shows the sequential link between the different parts. Figure 1.8 illustrates the outline of this thesis and provides the objective of each chapter.

Chapter 2 provides a state-of-the-art on the prognostics field in predictive maintenance, based on the initial research questions introduced in this chapter. The chapter reviews recent advances of the latest approaches used in the prognostics field, especially Machine Learning techniques. It investigates the application and contributions of these methods to the field, and highlights fundamental research challenges and directions associated with the current Machine Learning

algorithms. The chapter ends by refining the initial research questions according to the study findings.

Chapter 3 seeks to address the limited availability of large public datasets for fatigue damage prognostics problems. Large datasets are indeed a prerequisite for applying some of the latest DL techniques. This chapter proposes a framework and code for synthetically generating arbitrarily large data sets for a realistic fatigue damage prognostics problem. As illustration, pre-cracked Aluminum alloy 7075-T6 plates were considered, which are typical of aeronautic structures, and some of the most commonly used DL models to address failure prognostics were implemented and compared on this dataset.

Chapter 4 addresses the challenge of data scarcity by investigating the learning paradigm of Self-Supervised Learning (SSL) in a fatigue damage prognostics problem, when only limited labelled data is available. SSL is a sub-category of unsupervised learning approaches that have been proposed by several researchers in AI to address data scarcity challenge, showing promising results. The proposed SSL approach will be detailed in this chapter, and the results of this investigation explained and discussed. The investigative research conducted in this chapter resulted in an article submitted for publication to an international journal at the time of writing of this manuscript.

Chapter 5 concludes the thesis with the overview of the contributions, the identified limitations during the research, and identifying the perspectives of future work.

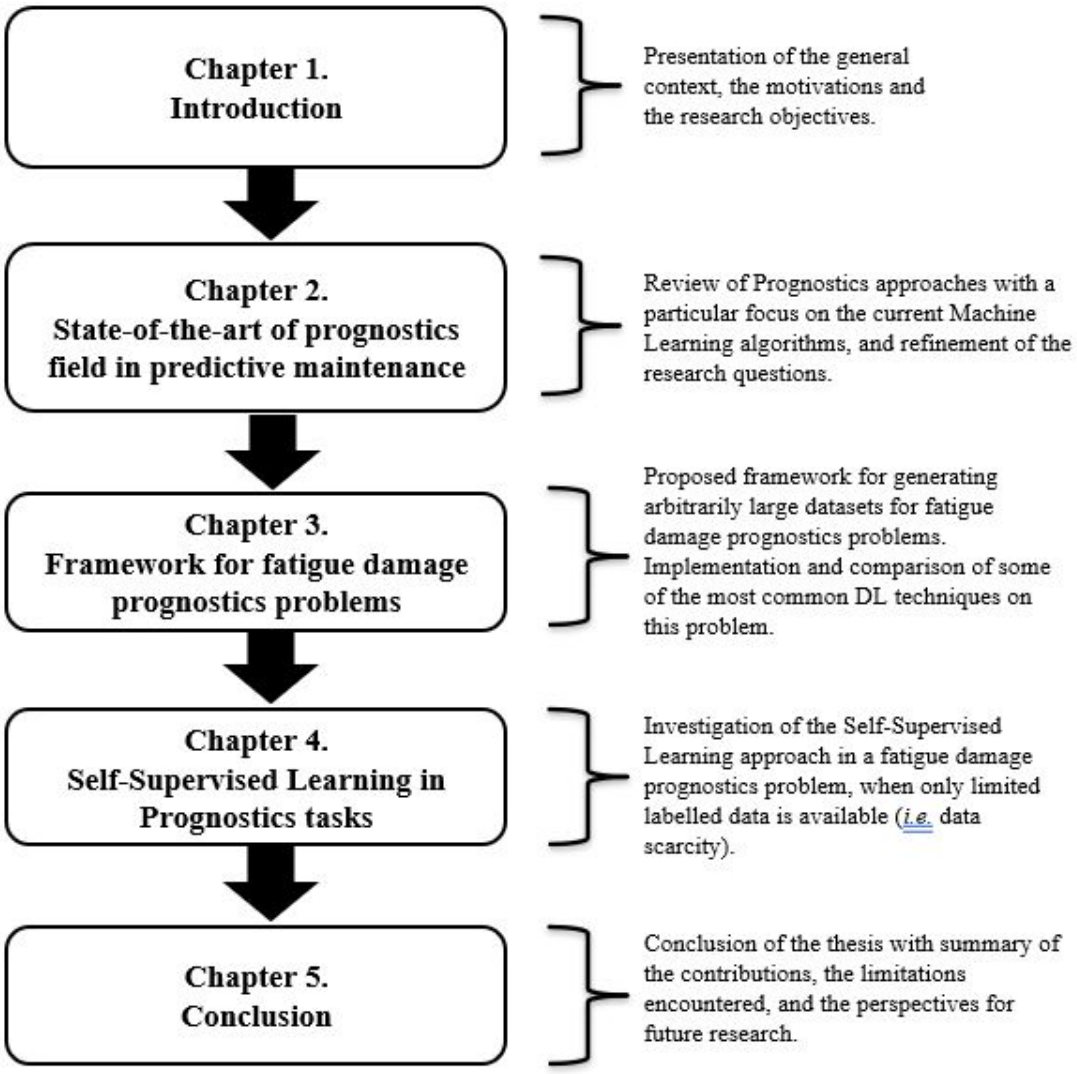


Figure 1.8: Outline of the thesis.

CHAPTER 2

Review of prognostics approaches with a particular focus on the current Machine Learning algorithms

"The secret of change is to focus all of your energy, not on fighting the old, but building on the new."

Socrates

Contribution

This chapter reviews recent advances in Machine Learning techniques used exclusively in prognostics. It investigates the application and contributions of these methods in the field, and highlights fundamental research challenges and directions associated with the current Machine Learning algorithms.

Content

2.1 Introduction

Data-Driven models have gained more and more attention in the PHM community, especially the latest ML techniques (*e.g.* Deep Neural Networks). Let us thus recall the three research questions proposed at the end of Chapter 1:

RQ1 : What are the current trends in prognostics for predictive maintenance ?

RQ2 : Which data-driven Machine Learning techniques are most commonly used to address failure prognostics, and what are their main key strengths and limitations ?

RQ3 : What are the existing related challenges and current research areas ?

In order to address these questions, this chapter provides an overview of the most commonly used prognostics approaches in the engineering field, with a focus on current Data-Driven models, in particular Machine Learning models (*e.g.* Deep Learning). At the end of the chapter, refined research questions will be formulated, motivating the remainder of the manuscript.

Recently, several reviews related to Machine Learning applications in PHM have been published [2], [12]–[14], [20], [21]. Some papers focus on specific approaches in both prognostics and Diagnostic tasks based on Machine Learning models [20], [21] or Deep Learning models [14]. In [13], Montero-Jimenez et al. presented the current trends in prognostics and diagnostics with a focus on multi-model approaches, *i.e.* the combination of two or more predictive algorithms. In [2], [12], the authors provided a summary of the main prognostics approaches with a brief presentation of multi-model approaches (which were called *Hybrid models* in their paper). Although these reviews present a detailed survey, to the best of author's knowledge, little to no research has directly constructed a thorough treatment of the applications and contribution of the latest Machine Learning approaches (notably those based on Deep Neural Networks) in the prognostics field. Moreover, due to the increase in computing power, the number of applications of the latest

ML techniques in PHM has grown exponentially in the last years, and many methods and tools for failure prognostics have been proposed to tackle existing challenges in the field.

As a consequence, the author of this review seek to enrich the material presented in the previous review papers. The current study proposes a review of the latest ML techniques used exclusively in prognostics, as opposed to both diagnostics and prognostics as in previous reviews, to provide a clear starting point for researches in the prognostics field being interested in the latest ML algorithms such as Deep Neural Networks. The survey gives a summary of their characteristics, their key strengths and limitations, illustrated by their applications in prognostics. In addition, special attention is given to emerging techniques combining two or more data-driven models after a better understanding of their strengths and limits: the Multi-Model approaches. These approaches are used to overcome complexity of predictive maintenance tasks and other current challenges. Finally, the existing related challenges of the current ML algorithms in prognostics are pointed out, including *data scarcity*, and *uncertainty*.

The remaining of this chapter is structured as follows. In Section 2.2, an overview of the current trends in the prognostics field is provided. Section 2.3 deals with the latest Machine Learning techniques used in prognostics. These data-driven approaches are discussed, with a particular focus on the Deep Learning algorithms. In Section 2.3.2, the identified challenges of presented Machine Learning techniques in prognostics will be discussed. Some proposed solutions tackling these issues are also discussed. Section 2.5 summarizes and analyzes the research questions presented in this chapter with some research perspectives, followed by a conclusion in Section 2.6.

2.2 Current trends in prognostics for predictive maintenance

Failure prognostics can be performed according to different techniques using different modeling, processing and analysis tools. These approaches can be classified into three main categories: Knowledge-based, Physics-Based and Data-Driven models. The taxonomy of prognostics approaches used in this literature review is illustrated in Figure 2.1.

Knowledge-based models are a type of model that, in PHM, are based on the exploitation of knowledge acquired by expert analysis about the failure or degradation of the component in the past [13]. Knowledge-based algorithms can correspond to models simulating a simplified reasoning mechanisms of human experts (*e.g.* rule-based and fuzzy knowledge-based models), or models whose knowledge is obtained through previously experienced similar cases (*e.g.* case-based mod-

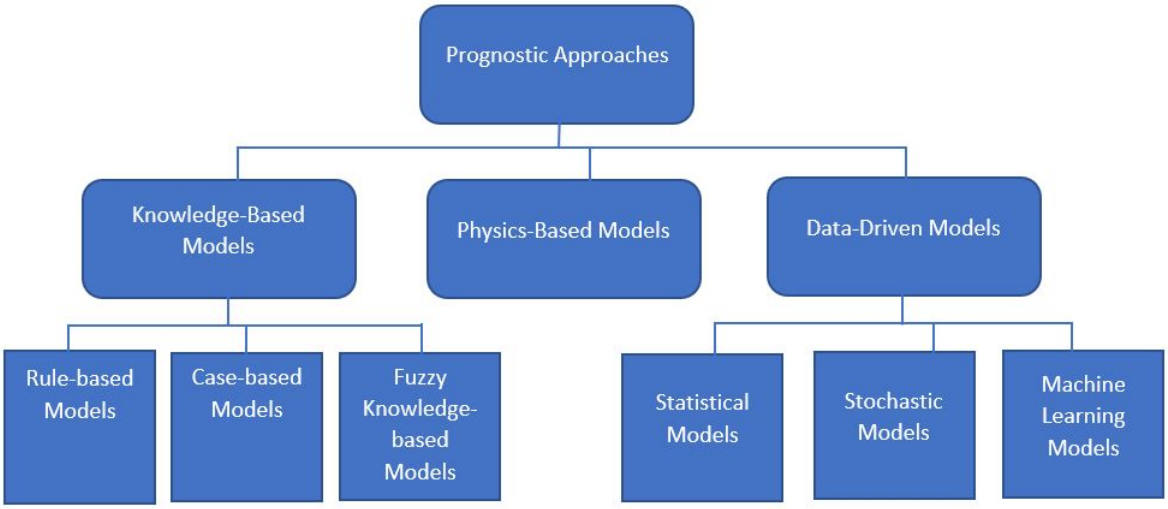


Figure 2.1: Taxonomy of prognostics Approaches considered in this literature review.

els). The reader may refer to the review paper [13] for an in-depth analysis of Knowledge-Based models used in PHM.

Physics-based models are commonly used in prognostics to predict the failure of a component [22]. They are derived from the laws of physics (mechanics, chemistry, electricity, hydraulics, etc.), and incorporate the observation of physical realities (operational or environmental conditions: vibrations, temperature, humidity, etc ...) and of the system to be predicted. Damage and failure are related by a mathematical expression, i.e. a degradation model. Physics-based methods are the most accurate techniques for prognostics if an accurate physical model is available as they are based on a profound knowledge of the system to be predicted [22], [23]. Moreover, physics-based models are able to provide a physical explanation to the results, which might be necessary sometimes in PHM [24]. However, it remains difficult to develop an accurate physics-based technique, due for example to measurement inaccessibility, coarse models, expertise required or prohibitive computational cost for finer models [25].

Data-driven models are a type of model that rely on the exploitation of monitoring data, which are processed to extract characteristics reflecting the system's evolution and/or degradation. These characteristics can then be used to learn predictive models of the future states of the structure or system (*e.g.* RUL estimation). Over the past few years, Data-Driven models have gained more and more attention in the PHM community [15]–[17]. The Data-Driven learning architectures are becoming more mature and with the availability of massive data (*e.g.* sensor data), they can in some cases outperform previous state-of-the-art (SOTA) prognostics approaches, such as

Knowledge-Based and Physics-Based models. Currently, this is certainly the most active category of approaches, and can be decomposed into three sub-categories [13]: 1) Statistical models; 2) Stochastic models; 3) Machine Learning models.

The current Machine Learning techniques, especially Deep Neural Networks, are now the subject of considerable attention from researchers in many fields such as in Natural Language Processing (*e.g.* GPT-3 [26]) or Image Processing [27]. Indeed, data storage is increasingly affordable, and data is becoming more and more numerous and diverse. These approaches are gaining interest in prognostics for predictive maintenance due to their low development cost, flexibility and suitability for complex systems and large amounts of data (*e.g.* sensor data as far as prognostics is concerned). Hence, in the following, the author will focus on Machine learning models, especially Deep Learning models, while the interested reader can refer to [13] for a detailed review of other data-driven models for prognostics.

In this review, the author have selected the most significant research articles in prognostics field published during 2010–2021. Some papers that were reviewed have been found on open-access repositories as preprints, mainly from ArXiv, and were selected due to their relevance to the subject and quality. The main focus was papers from the most reputed publishers such as IEEE, Elsevier, and Springer, excluding non-PHM articles by identifying absence of commonly employed PHM terms such as *prognostics*, *RUL*, *failure*, or *degradation*. The author have reviewed scientific papers on various ML topics. Moreover due to the necessity of a relatively large amount of training data in order to compare and evaluate the latest developments of ML techniques, a large majority of the research has been conducted on state-of-the-art open-sourced datasets, in particular: turbofan engines [9], bearings [28], batteries [29].

Among the machine learning-based models investigated in the prognostics field, the algorithms that appeared to be the most used and that will be considered in this chapter are the following: Support Vector Machines (SVM) [30], Recurrent Neural Networks (RNN) [31] and Convolutional Neural Networks (CNN) [32]. Therefore, the keywords used for search criteria for this review are ((“Support Vector Machines” OR “Neural Networks” OR "Recurrent Neural Networks" OR "Long Short-Term Memory" OR "Gated Recurrent Unit" OR "Convolutional Neural Networks") AND (“Remaining Useful Life” OR "prognostics")). The articles reviewed in this literature review were selected in the following way:

1. references that illustrate the application of the models in prognostics;
2. references that cover their major advantages and theoretical issues;
3. references that propose new methods overcoming these issues or related challenges in the prognostics field, thus guiding the readers in interesting research directions.

Among Deep Neural Networks, only RNNs and CNNs were considered in Section 2.3, which appear to be the most used algorithms in prognostics. Moreover, this literature review shows that, among the latest ML algorithms investigated in the prognostics field, recurrent neural networks are gaining importance in the research community, especially over the last decade. To illustrate this, the author have selected 587 relevant papers published in the last decade (2010-2021) in one of the consulted search sources (ScienceDirect). Fig. 2.2 illustrates the distribution of the published papers per year in the prognostics field, considering the Machine Learning approaches described in this review, used in direct application for prognostics tasks (*e.g.* RUL estimation). The following search terms pattern have been used: (“Support Vector Machines” OR “Neural Networks” OR "Recurrent Neural Networks" OR "Long Short-Term Memory" OR "Gated Recurrent Unit" OR "Convolutional Neural Network") AND (“Remaining Useful Life”). The last search on ScienceDirect was on August 02, 2022; however, the author cannot fully guarantee to have taken into account all the available studies in this research area.

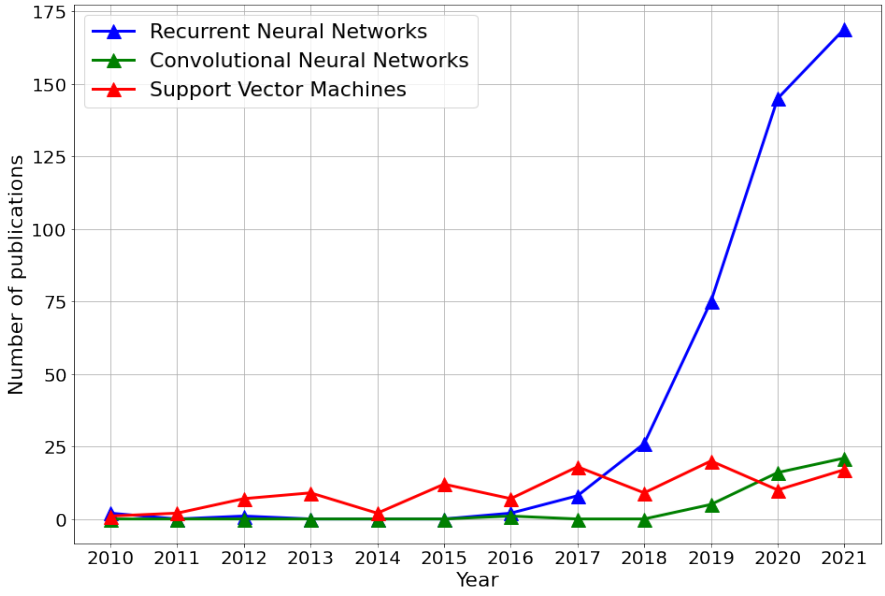


Figure 2.2: Number of publications over the last decade related to the ML approaches described in this review, using the search terms enumerated above in ScienceDirect.

Note that from 2017 to 2021, more than 465 papers addressing the use of Deep Learning techniques (RNN and CNN models) in prognostics were published, which represents approximately 6 times more publications than published papers addressing the use of SVM models (*i.e.* 74 publications). Hence, in this survey, there will be a special focus on the latest Deep Learning techniques in the prognostics field. Furthermore, other DNN algorithms used in multi-model approaches will be presented and discussed in the following section.

2.3 Machine Learning models for prognostics

In this section, the most commonly used Machine Learning techniques used to address failure prognostics are listed with their key strengths and limitations. Note that in this study, the author distinguishes between single-model approaches and multi-model approaches that seek to combine several individual models.

2.3.1 Single-model approaches

Among single-model approaches in prognostics, two widely used Machine Learning models can be listed:

- Support Vector Machines;
- Artificial Neural Networks.

2.3.1.1 Support Vector Machines

Support Vector Machines (SVM) are a method of binary classification by supervised learning [30]. SVM were initially applied on two-class classification problems, as a linear discriminant: this method uses a training data set to learn the model parameters. The idea of SVM is to construct a linear separating hyperplane to separate the training data into two classes, using an optimization problem. The SVM algorithm formulates the problem such that the objective is to maximize the margin between the data points of the two classes, using the parameters of the problem to guide the optimization process. This can be achieved by finding the maximum distance between the data points of the two classes, which allows for an effective separation of the data. Assuming that the two classes (denoted -1 and +1) are separable by a defined hyperplane, the optimization problem can be stated as follows:

$$\min_{w,b} \frac{1}{2}||w||^2 \quad \text{s.t.} \quad y_i(w^T x_i + b) \geq 1 \text{ for } i = 1, \ldots, n \tag{2.1}$$

where w and b are the coefficients of the hyperplane, $x_i \in \mathbb{R}^n$ are the data points and $y_i \in \{-1, +1\}$ the corresponding label.

The objective of this optimization problem is to find the coefficients w and b that maximize the margin between the two classes, while also ensuring that the data points are correctly classified by

the hyperplane. The margin is calculated as $\frac{2}{||w||}$. The constraint $y_i(w^T x_i + b) \geq 1$ ensures that all data points are at least one margin away from the boundary. Fig. 2.3 illustrates the procedure of the SVM algorithm. A main advantage of SVM is that it only requires relatively small amount of training samples if a feature space in which the data separation is linear is available.

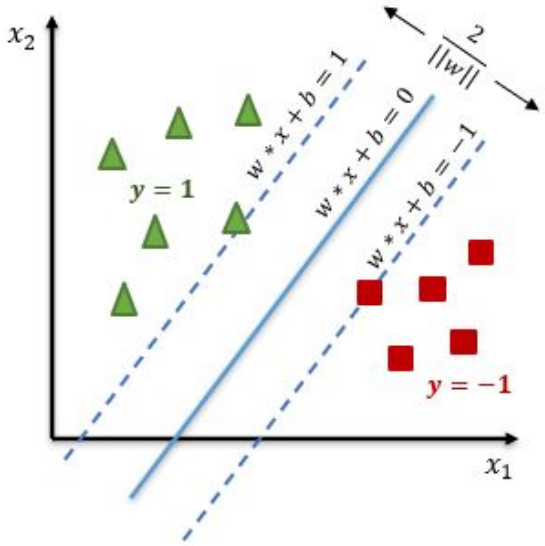


Figure 2.3: Linear SVM separation principles in a 2-dimensional space.

Initially used by the PHM community to solve the fault diagnostics problem [33], the application of this algorithm was later extended to the RUL prediction field [34]–[38]. However, SVMs were initially designed for classification problems (*e.g.* predicting a discrete class label for the RUL), while practitioners in the prognostics field might be more interested in a regression technique (*e.g.* predicting a continuous quantity output for the RUL). Drucker et al. [39] proposed a new regression technique based on the concept of support vectors, Support vector Regression (SVR). SVR can handle regression problems [40] and is applicable to time series processing [41]; this technique has been considered in several works for RUL estimation as a regression technique [42]–[44].

Despite its acknowledged success, SVM's parameters need to be specifically tuned which might be difficult [45] without any knowledge of the test case [46], and they seem to be not very well suitable to deal with complex and large datasets [47]. Note that SVM techniques are not in the focus of this chapter, thus the interested reader may refer to [48] to have a detailed review of the existing applications in prognostics.

2.3.1.2 Artificial Neural Networks

Neural networks, originally developed in the 1940s, seemed promising as they attempted to mimic, in a highly simplified form, the functioning of the brain as it was then understood. Due to their ability of modelling nonlinear processes, neural networks have found application in many fields [49], [50].

With the increasing availability of data, ANNs have become a popular technique also in the prognostics field [51], [52]. As a Data-Driven model, ANNs allow the construction of a mathematical model representing the component from available observational data rather than from physical laws (which presuppose a thorough knowledge of degradation phenomena)[53]. In this thesis, as illustrated in Fig. 2.4, two types of Artificial Neural Networks are considered:

- Shallow Neural Networks (SNN);
- Deep Neural Networks (DNN).

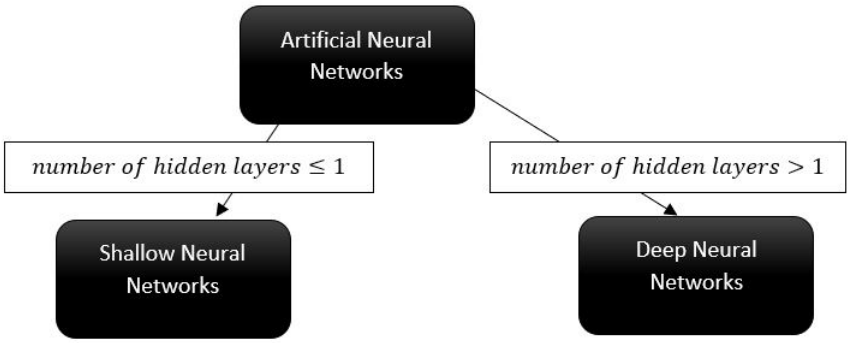


Figure 2.4: Taxonomy of Artificial Neural Networks in this literature review.

Shallow Neural Networks, also known as a single-layer perceptron, is a simplified version of ANNs, first introduced by Rosenblatt in 1958 [54]. The architecture of an SNN typically consists of a single layer of artificial neurons, or perceptrons, that are connected to the input data through a set of weights. The input data is passed through the perceptrons and processed in parallel. Each perceptron applies a linear or non-linear activation function to the weighted sum of the input data, producing a final output. The weights of the perceptrons are adjusted during training using an optimization algorithm, such as gradient descent, to minimize the error between the predicted output and the true output.

The structure of a Shallow Neural Network is illustrated in Fig. 2.5. The artificial neurons (perceptrons) are represented by circles. The process of passing input data through the perceptrons

and adjusting the weights is known as feed-forward propagation.

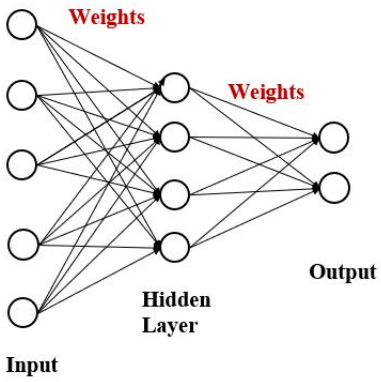


Figure 2.5: Single-layer perceptron architecture, or Shallow Neural Network.

The mathematical formulation of a Shallow Neural Network can be formulated as:

$$y = f(w^T x + b) \tag{2.2}$$

where x is the input data, y the output, w is the weight vector, b is the bias, and $f(.)$ is the activation function. In this equation, the perceptron computes a linear combination of the input data x and weights w and adds the bias term b. The result is then passed through an activation function, such as the sigmoid or rectified linear unit (ReLU) function, to produce the output.

In [55], the authors provided a summary of the applications until 2012 of SNNs in prognostics of rolling bearings. Gebraeel et al. [56] presented a set of neural-network-based models to estimate the bearing failure time from vibration-based degradation signals. Riad et al. [57] compared Neural Networks to a linear regression model in RUL estimation (PHM 2008 Data Challenge dealing with turbofan engine data [9]). Results showed that neural networks were able to outperform the simpler linear regression technique in solving the given tasks. In 2016, Bektas and Jones [58] introduced a nonlinear autoregressive neural network (NARX) prognostics model for RUL estimation of gas turbine engines. Authors presented this approach as a form of dynamic filtering in which past values of a time series are used to predict future values. Haynes et al. [59] compared this technique to Particle Filters (PF) for prognostics of fatigue crack growth in pre-cracked Aluminum coupons. The proposed physics-based approach was applied over the Paris-Erdogan's model, one of the most widely used degradation models in RUL [60], [61]. Results showed that the margin of error of the PF is larger than that of the NARX model, but becomes smaller as the parameter update progresses towards the terminal crack length (*i.e.* the end of fatigue life). According to the authors, it might be interesting to implement multi-model techniques (cf. section 2.3.2) combining the results obtained, such as the NARX approach with PF methods. An et al. [62] compared four widely used algorithms in RUL estimation: Gaussian Process Regression,

Particle Filters, Bayesian Methods and a Neural Networks (a shallow neural network composed of one hidden layer). According to the authors, Neural Networks are appropriate for the case of high noise and complex models with large amounts of training data. Their results also show that NNs are more accurate for long-term predictions when the data set is large. Authors also suggest that adding hidden layers to neural networks might improve results. Moreover, [25] highlights that increasing neural networks' architecture could improve model's accuracy when the dataset is large, *i.e.* Deep Neural Networks.

Deep Neural Networks is a class of Artificial Neural Networks. Indeed, to represent more complex features and to "learn" increasingly complex models for prediction and classification of information that depend on multiple features, the SNNs model presented above need to be scaled up. This is accomplished by increasing the number of hidden layers or the number of neurons per hidden layer. Such neural networks are called Deep Neural Networks; training is then called Deep Learning. More layers and more neurons can represent increasingly complex models but they also come at the cost of increasing time and power-intensive computations. Multi-layer Perceptrons (MLP) are a type of feedforward deep neural network that consists of multiple layers of artificial neurons. The structure of an MLP is typically composed of an input layer, two or more hidden layers, and an output layer. Each layer is fully connected to the next, meaning that each neuron in one layer is connected to all neurons in the next layer. The activation function, which is applied to the output of each neuron, is typically a non-linear function such as the sigmoid or rectified linear unit (ReLU) function. The architecture of an MLP is shown in Fig. 2.6.

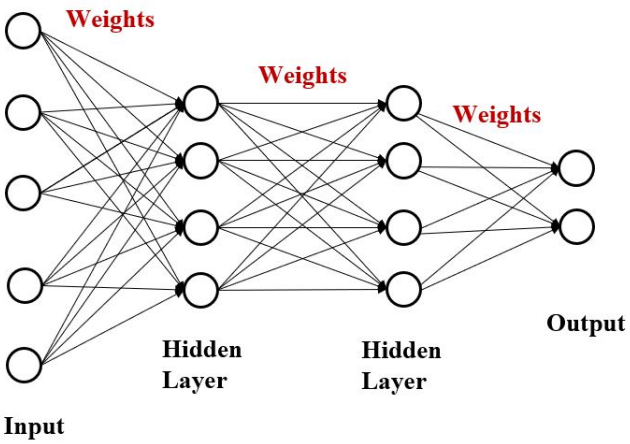


Figure 2.6: MLPs architecture. In this illustration, the MLP is composed of two hidden layers.

Among the deep learning single-model approaches, two widely used algorithms which appear promising in the prognostics field can be enumerated:

- Deep Recurrent Neural Networks;
- Deep Convolutional Neural Networks.

Deep Recurrent Neural Networks Recurrent Neural Networks (RNNs) [31], [63] are a type of neural network that are designed to process sequential data, such as time series or natural language. They are characterized by the use of feedback connections, which allow information to flow through the network across multiple time steps. This allows RNNs to maintain a hidden state that can capture information from previous time steps, allowing them to make predictions or decisions based on the entire sequence of input data.

The basic building block of an RNN is the recurrent unit, which computes the hidden state h_t at time step t as a function of the input x_t at time step t and the hidden state h_{t-1} at time step $t-1$. The hidden state h_t is then used to compute the output o_t at time step t. The parameters of the RNN, such as the weight matrices W and bias vectors b, are learned during training. The formulas that govern the computation happening in an RNN network are as follow:

$$\begin{aligned} h_t &= f(W_{hx}x_t + W_{hh}h_{t-1} + b_h) \\ o_t &= g(W_{ho}h_t + b_o) \end{aligned} \tag{2.3}$$

where x_t is the input sequence at time step t, h_t is a hidden state, o_t is the output, f and g are activation functions (usually non-linear functions), W_{hx}, W_{hh}, W_{ho}, b_h and b_o are the weight matrices and bias vectors which are learned during training. The structure of a simple RNN is shown in Fig. 2.7.

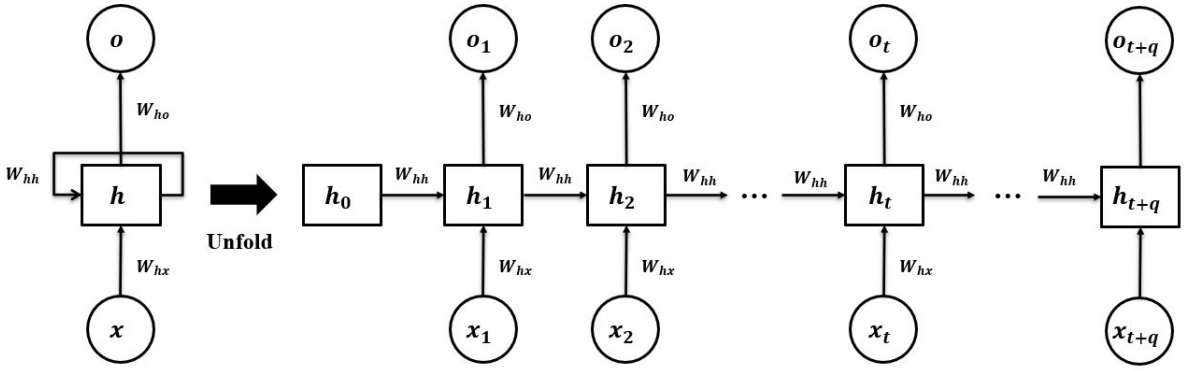


Figure 2.7: RNN architecture and the corresponding unfolded structure. h_0 represents an initialized hidden state, and $t+q$ the prediction horizon. A loop allows information to be passed from one step of the network to the next.

There have been incredible successes applying RNNs to a variety of problems in non-PHM fields: speech recognition, language modeling, image captioning [64]–[66]. Due to their ability to capture time-dependent relationships, RNNs have achieved great interest among PHM researchers as well, especially given the sequential nature of the sensor data in the PHM field.

Many types of RNNs have been developed over the last 40 years. The pioneer recurrent neural networks such as Elman Networks (ERN) [67] or Jordan Networks [68] were able to perform a sequence-prediction that were beyond the power of a typical Deep Neural Network. These two approaches, quite similar in their structure, are also sometimes called Simple Recurrent Networks and have been explored by several researchers among the PHM community [69], [70]. Yan et al. [69] proposed an Elman Recurrent Networks-model for material degradation evaluation and life prediction, applied to blade material's degradation analysis. Kramti et al. [70] used the ERM for high-speed shaft bearing prognostics based on vibration signals. Results presented in both papers showed that the recurrent networks might be an interesting technique for dealing with time-dependent sequential data (such as vibrations, loading data), proposing a 'reliable and robust material fatigue predictor' according to Yan et al. [69].

Other types of RNNs have been explored since, varying the architecture or the topology of the network. Malhi et al. [71] introduced a novel approach to long-term prognostics of machine health status using Recurrent Neural Networks. Aiming at improving the long-term prediction accuracy, the authors first used a Continuous Wavelet Transform (CWT) on vibration signals data from a defect-seeded rolling bearing, then used the data preprocessed as an input to an RNN model for RUL estimation. Peng et al. [72] and Rigamonti et al. [73] proposed a novel RNN, the Echo State Networks [74], to achieve RUL estimation applied on turbofan engine data. The originality of this approach is that the training procedure is based on a simple linear regression and offers a calculation time considerably lower than that of shallow RNNs, while still providing the same generalization capability characteristic. Liu et al. [75] proposed an adaptive recurrent neural network for RUL estimation of Li-ion Batteries. Indeed in a Deep Neural Networks training procedure, weight adjustment with a backpropagation algorithm (*e.g.* gradient descent) may result in the local minima problem. This problem can be avoided by lowering the learning rate and increasing the number of training epochs for example [76], [77]. This can help in achieving better results, getting closer to the global minima, but it requires significant computational time. Hence the approach proposed by Liu et al. [75] is called adaptive because the network weights are adaptively optimized using the embedded recursive Levenberg-Marquardt (RLM) method [78]. Their results, comparing the proposed method with other implemented prognostics methods (including Particle Filters, a Bayesian form of SVM and shallow RNNs), showed that it was able to outperform them on the considered battery RUL prediction problem. Zhi et al. [79] developed a new approach using RNNs and the Extreme Learning Machine (ELM) [80] for modeling crack growth of aluminum alloy. The specificity of their approach is that it embeds the ELM method to train the RNN and randomly assign the input weights (uniformly) in a proper range and globally optimize the output weights, seeking to avoid the local minima issue. Results showed that this approach enhanced the accuracy of the dynamics of crack growth under variable-amplitude loading model and made the training faster.

Theoretically, RNN can make use of the information in arbitrarily long sequences, but in practice is limited to looking back only a few steps due to the vanishing gradient or exploding gradient problem [81], [82]. To avoid the problem of vanishing or exploding gradient, the Long Short-Term Memory (LSTM) network was proposed by Hochreiter and Schmidhuber in 1997 [83].

Long Short-Term Memory (LSTM) networks are a variant of recurrent neural networks that are designed to overcome their limitation in capturing long-term dependencies in sequential data. They have become very popular models in many fields and especially in Natural Language Processing (NLP) [84]. LSTM networks introduce additional gates, such as the forget gate f, input gate i and output gate o, which allow the network to selectively store or discard information from previous time steps in the hidden state. The structure of an LSTM network is shown in Fig. 2.8.

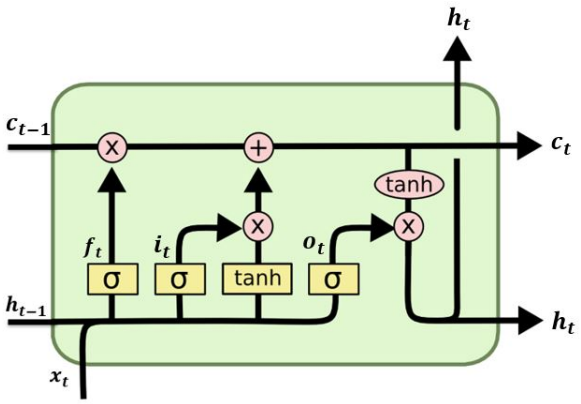


Figure 2.8: LSTM architecture. *Source*: reproduced based on [85].

The basic building block of an LSTM network is the LSTM cell, which computes the hidden state h_t at time step t as a function of the input x_t at time step t, the hidden state h_{t-1} at time step $t-1$, and the cell state c_{t-1} at time step $t-1$. The cell state c_t is used to store information across time steps and is updated by the forget gate f_t and input gate i_t. The parameters of the LSTM network, such as the weight matrices W and bias vectors b, are learned during training. The formulas that govern the computation happening in an LSTM network are as follow:

$$
\begin{aligned}
f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\
c_t &= f_t * c_{t-1} + i_t * \tanh(W_c x_t + U_c h_{t-1} + b_c) \\
h_t &= o_t * tanh(c_t)
\end{aligned}
\tag{2.4}
$$

where x_t is the input sequence at time step t, h_t is a hidden state, c_t is the cell state, f_t, i_t, o_t are the forget gate, input gate, and output gate respectively, $\sigma(.)$ represents the sigmoid activation function and $tanh(.)$ the hyperbolic tangent non-linear function. W and U denote the weight matrices which are learned during training, and b denotes the bias vector. The idea behind the LSTM is that each unit is linked not only to a hidden h state, but also to a cell state c that acts as a memory. The forget gate, input gate and output gate must respectively determine what cell content needs to be forgotten, what cell content can be added and what cell content can be output to the hidden state. More details in [85].

LSTM networks have also grown in popularity and have been used by several researchers in the PHM community. They were considered by Nguyen et al. [86], Yuan et al. [87] and Wu et al. [88] for RUL estimation of turbofan engines; by Ma et al. [89] for Proton exchange membrane fuel cell (PEMFC) degradation prediction. In [88], due to its ability of capturing the long-term spatiotemporal dependency of multiple degradation features, the authors showed that their LSTM model outperformed an RNN model for RUL estimation of turbofan engines. Park et al. [90] confirmed the ability of LSTMs in prognostics to handle multiple measurable data from a battery management system and outperforming RNNs. Cheng et al. [91] presented a novel real-time LSTM-based method using a spectral clustering algorithm for failure time prediction of bearings. Their approach consisted in:

1. feature extraction (time, frequency and time-frequency domains) and degradation feature selection (based on a Euclidean distance algorithm);
2. a kernel spectral clustering algorithm to adaptively identify machine anomaly and detect early failures;
3. an LSTM network model was used for failure time estimation.

Authors showed that this method outperformed other state-of-the-art data-driven techniques (including Relevance Vector Machines, a Bayesian form of SVM models [92]) in the bearing prognostics application considered.

One of the major drawbacks of LSTM are their relatively high computational cost and memory requirement for training (due to multiple memory cells). A slightly more simplified variation of the LSTM is the Gated Recurrent Unit, or GRU, introduced by Cho et al. [93]. It combines the forget and input gates into a single "update gate", and merges the cell state and hidden state. The formulas that govern the computation happening in a GRU network are as follow[93]:

$$
\begin{aligned}
z_t &= \sigma(W_z x_t + U_z h_{t-1}) \\
r_t &= \sigma(W_f x_t + U_f h_{t-1}) \\
\tilde{h}_t &= tanh(W_{\tilde{h}} x_t + U_{\tilde{h}}[h_{t-1} * r_t]) \\
h_t &= z_t * \tilde{h}_t + (1 - z_t) * h_{t-1}
\end{aligned}
\tag{2.5}
$$

where x_t is the input sequence at time step t, h_t a hidden state, z_t the update gate, r_t the reset gate, $\tilde{h}_t$ a cell state, $\sigma(.)$ represents the sigmoid activation function and $tanh(.)$ the hyperbolic tangent non-linear function, W and U denote the weight matrices which are learned during training. The pink circles represent pointwise operations (*e.g.* addition, multiplication). The idea behind the GRU network is that in each unit, the update gate z_t must select whether the hidden state h_t is to be updated with a new hidden state $\tilde{h}_t$; the reset gate r_t must decide whether the previous hidden state h_{t-1} is ignored. More details can be found in [93]. The structure of the GRU network is shown in Fig. 2.9.

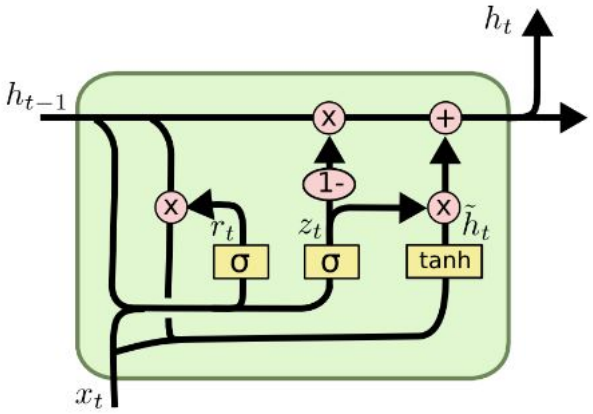


Figure 2.9: GRUs architecture. Source in [85]

This approach has gained in popularity in recent years due to its relative simplicity (*i.e.* lower complexity and faster computation [94]), while the same ability to capture the mapping relationships among time series data [95]. Indeed, Baptista et al. [96] compared three variants of RNN (shallow RNN, LSTM and GRU) with other shallow machine learning models (such as Artificial Neural Networks and Support Vector Regression models), all implemented and applied on two real-world aero-engine datasets for RUL estimation. Their results showed that LSTM and GRU outperformed the other models, and that the GRU model could achieve competitive results with a better training performance than LSTM.

Deep Convolutional Neural Networks Convolutional Neural Networks (CNN) is a specific type of deep neural network inspired by the organization of neurons in the visual cortex. Presented by LeCun et al. in 1998 [32], CNNs have achieved significant success in many research and industry fields including Computer Vision, Natural Language Processing and Speech Recognition [97], [98]. Fig. 2.10 illustrates the architecture of a traditional CNN applied to image classification.

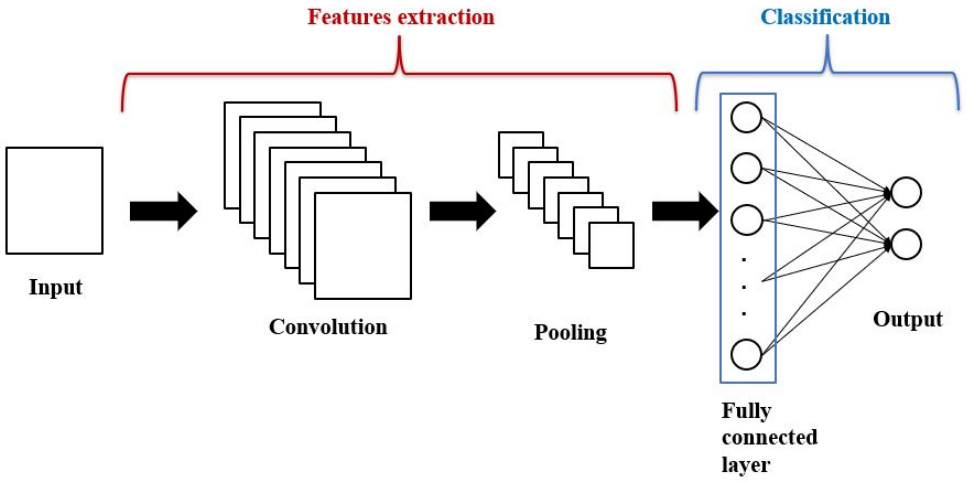


Figure 2.10: Architecture of a traditional CNN applied to image classification.

CNNs typically consist of several layers, including convolutional layers, pooling layers, and fully-connected layers. Convolutional layers apply "filters" to the input data, which analyze the data in small areas and look for specific patterns. Pooling layers reduce the amount of information in the data while maintaining the most important features. In image processing, this layer allows for the reduction of the size of an image while keeping the most important pixels. Fully-connected layers take the output of the previous layers and flatten them into a single vector, which can be used as the output or input for the next step. The reader may refer to [99] and [100] for more details about the Convolutional Neural Networks.

Originally designed for Image Processing, the input data for CNNs are usually 2-dimensional data (*i.e.* height and width pixels for images) in order to learn abstract spatial features. Li et al. [101] investigated how a 2D-DeepCNN model can be used in prognostics and remaining useful life prediction. In their work, experiments were carried out on the popular NASA C-MAPSS dataset [9], and the input data was prepared in 2D format. The first dimension corresponds to the feature number (here 21 features), the second is the time sequence of each feature (time window). The remaining useful life is predicted via a sliding-window time series analysis. Results showed that the proposed Convolutional Neural Networks outperformed other state-of-the-art prognostics approaches the authors also implemented (including RNN and LSTM) on the same turbofan engine dataset. It is worth noting that CNNs can be applied to other types of data such as time series data and text data, by using 1-D convolutional layers (1D-CNN).

1D-CNNs have also been introduced to the analysis of time sequences in RUL estimation. The key difference between 1D-CNN and 2D-CNN is the dimensionality and management of the input data as well as how the feature detector (or filter) slides across the data. The application of the CNN architecture to time series prediction aims to exploit the filters' feature extraction capability demonstrated in image classification. Moreover, CNNs are easier to train than recurrent neural networks due to the implementation of convolutional rather than recursive operation, allowing improved numerical efficiency. Xu et al. [102] proposed a CNN-based model for Fatigue Crack Diagnostics (FCD). First, time-domain and frequency-domain Damage Indexes were extracted from sensor data, which, according to Xu et al., can reflect the fatigue crack growth effectively and provide different perspectives of crack information for the model to learn (for instance: spectrum loss, cross correlation [103]). Then, a CNN model is designed as a classifier: the fatigue cracks predicted by the model are classified into n crack sizes according to its crack length (in their work: 19 crack sizes, 1 crack size per 1 mm). According to the authors, the lowest diagnostic accuracy was 86.84%, with a diagnostic error of 1 mm.

However, CNNs were initially introduced as classifiers [32], thus more suitable for classification problems than regression problems in sequence modeling (*e.g.* RUL estimation problems that have been essentially considered as regression problems so far). Therefore in PHM, CNNs have been mainly used as single-model approaches for fault diagnostic tasks (classification) [20], given their ability to extract local spatial features through the network. They have also been integrated into several multi-model approaches as a discriminant or feature extractor in prognostics tasks which will be described in section 2.3.2.

In 2016, van den Oord et al. introduced a novel CNN architecture for sequence modeling, using dilated causal convolution to preserve the causal order of the input time series: WaveNet, a deep generative model [104]. The originality of WaveNet lies in:

1. being an autoregressive model;
2. causal convolution allowing it to preserve the causal order of the input time series;
3. dilated convolution allowing the network to operate on a larger receptive field than with a normal convolution by skipping input values with a certain step.

In their work, the authors applied WaveNet on audio data for audio generation (Text-To-Speech), showing promising results and performing better than the at-the-time most successful models presented in the literature (Deep RNNs and Hidden Markov Models) [104].

Bai et al. [105] introduced a novel CNN architecture for sequence modeling: Temporal Causal Networks (TCN). The structure of a TCN is similar to that of a CNN, but with an added element called *dilation factor* in the convolutional layers. The *dilation factor* allows to increase the

receptive field of the filter without increasing the number of parameters. This allows to preserve the causal order of the input sequential data and enables the network to efficiently model temporal dependencies at different scales and capture patterns at different time resolutions.

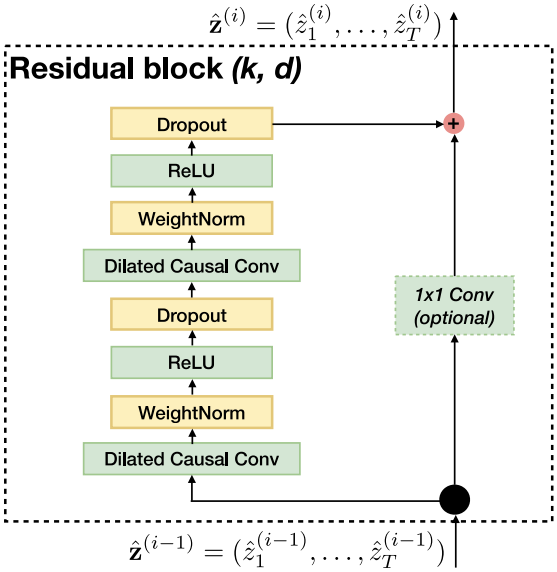


Figure 2.11: Architecture of a residual block of TCNs introduced in [105]. A TCN is a stack of k residual blocks, each composed of two 1D-CNN layers with a dilation factor d followed by a weight normalization layer used for regularization [106].

The main advantage of TCN over traditional RNNs is that it can handle longer sequences with less computational resources as it applies convolutions along the time axis instead of using recurrence. Moreover in [105], results showed that the TCN outperformed the standard recurrent neural networks (RNN, LSTM and GRU) in most Sequence Modeling Tasks (better performances on ten tasks out of eleven). Among PHM researchers, Liu et al. [107] proposed a TCN model for RUL prediction of rolling bearings based on raw vibration data. In their paper, results confirmed that the proposed model is able to outperform generic recurrent architectures such as LSTM and GRU in sequence modeling. Moreover, the authors showed that offline training of the proposed model is almost four times faster than an LSTM network. Indeed, according to Bai et al. [105], the TCN has several advantages that make it superior to the Recurrent Neural Networks (RNN, LSTM, GRU), specifically:

- better control of the model's memory size with a flexible receptive field size (stacking more dilated causal convolutional layers, using larger dilation factors, or increasing the filter size);
- a backpropagation path different from the one used by recurrent neural networks, which allows to avoid the exploding/vanishing problem.

Table 2.1 provides a description of the single-model approaches presented in this section.

Model	Strengths and Limitations	Description
SVM	Strengths: · Only requires small amount of data. Limitations: · Difficult to set the optimal algorithm's parameters.	· SVM classifier: efficient only in binary classification problems. · SVR: can handle regression problems and is applicable for time series processing.
ANN	Strengths: · Suitable for noisy and large datasets. · Able to model complex, multidimensional and nonlinear systems. Limitations: · Difficult to tune the hyperparameters. · Recommended to avoid neural networks for small datasets (prone to overfitting if the model used is too complex with too many parameters). · Interpretability ("black-box").	· SNN: non-complex architectures (zero or one hidden layer). · DNN: complex architectures (more than two hidden layers). - RNN: well suited for time-dependent sequential data (*e.g.* sensors data); relatively high computational cost and memory requirement for training (due to multiple memory cells) - CNN: able to account for spatio-temporal dependencies; easier to train than RNNs, with less complexity and less computational costs; have been shown to work best if problem is formulated as a classification problem.

Table 2.1: Description of the most commonly used Machine Learning techniques in prognostics described in this review.

2.3.2 Multi-model approaches

A multi-model approach is defined as the combination of two or more algorithms, used so that the overall algorithm is better than the individual ones (*i.e.* single-model approaches). These approaches can combine either models from different categories (for instance a data-driven model combined with a knowledge-based model) or multiple models of the same category and benefit from their advantages (*e.g.* applicability, precision, uncertainty management, etc.).

Note that in the literature, multi-model approaches are also commonly known under a different name: 'Hybrid models' [20], [22]. In their review paper, Montero-Jimenez et al. [13] introduced Hybrid models as part of multi-model approaches. The authors suggested that the Hybrid models

can be described as a combination of two or more models to perform a single task involving mutual cooperation among the combined models to obtain their outputs.

According to the taxonomy considered in this chapter, seven types of multi-model approaches are identified: combination of single-model approaches, of the same family or not (Knowledge-Based (KB) models, Physics-Based (PB) models, and Data-Driven (DD) models), see Figure 2.12 below.

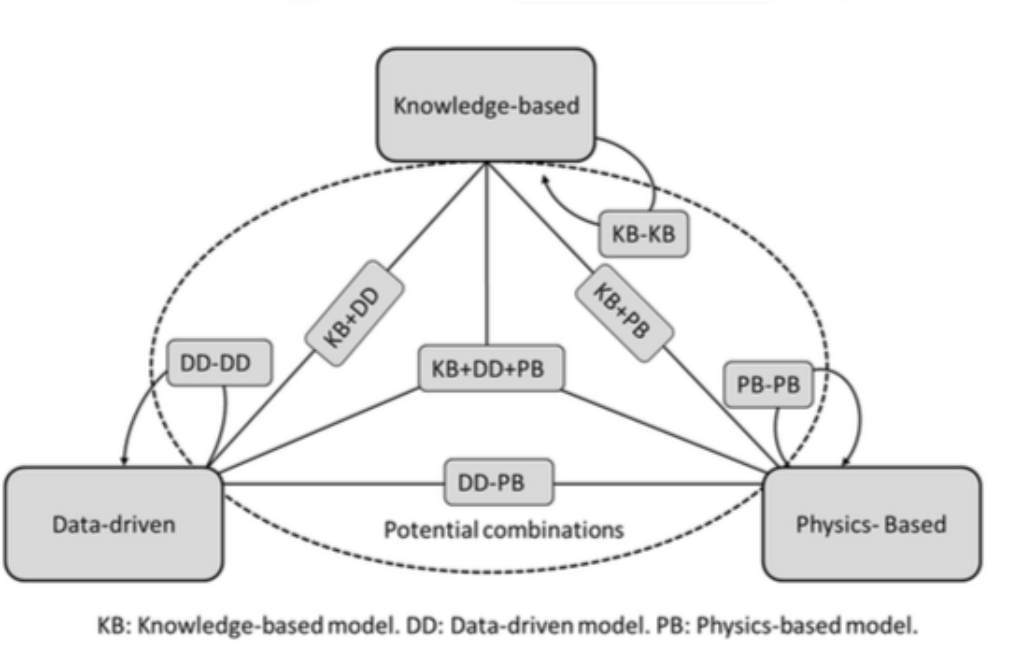


Figure 2.12: Combination of multi-model approaches proposed by Montero-Jimenez et al. [13]

According to [13], the combination of Physics-Based and Data-Driven models is the most common in recent research in PHM, due to their complementarity for degradation modelling and the increasing popularity of Data-Driven models. In their paper, Jouin et al. [108] suggested that the Particle Filters, one of the most commonly used Physics-Based techniques in failure prognostics [22], can be combined with other data-driven models (such as Neural Networks and SVR) to increase the performance of the model in the prognostics tasks. Moreover, [14] states that combining Physics-Based models with Machine Learning models might be a promising approach of inducing interpretability in the Machine Learning models. Among the pioneering works on multi-models combining a physics-based model and a data-driven model, Saha et al. [109] proposed a method combining a Bayesian regression approach (RVM) and a Bayesian estimation approach (Particle Filters) to estimate the RUL of Li-ion batteries. Based on an exponential growth model, the RVM regression was used to estimate the parameters, and the PF used to estimate the RUL. According to the author, combining two Bayesian approaches allowed this multi-model approach to take advantage from their ability to manage uncertainty, and benefit

from both classes which allowed improvements over the corresponding conventional single-model methods of RUL estimation: a purely data-driven method Autoregressive integrated moving average (ARIMA), and a purely Physics-based method Extended Kalman Filter (EKF). However, one of the drawbacks of multi-model approaches involving physics-based models is that for many industrial applications, it turns out to be quite hard to develop an accurate physics-based technique, due for example to measurement inaccessibility, coarse models or prohibitive computational cost for finer models [25]. Following the common thread of this chapter, particular attention will be paid in the following on multi-model approaches involving one or more ML techniques. The reader may refer to Montero-Jimenez et al.'s [13] review and Liao et al.'s article [110] for more informations about the other multi-model approaches and applications.

In most applications, Data-Driven models appear to be the most suitable and efficient approaches when neither a physical model nor expert knowledge are available or sufficiently accurate. Indeed, combining multiple data-driven models can sometimes be much more efficient than using a single-model, benefiting from the advantages of each algorithm. Fohr et al. [111] discussed the application of Deep Neural Networks in Speech Recognition field, and compared a combined Deep Neural Network-Hidden Markov Model DNN-HMM with Gaussian Mixture Model-Hidden Markov Model (GMM-HMM). The GMM-HMM, a variant of Hidden Markov Models [112], has been one of the most widely used techniques in speech recognition field during the previous decade. In [111], results showed that DNN-HMM, a multi-model combining Deep Neural Networks and HMMs, outperformed the state-of-the-art GMM-HMM model.

Inspired by such successes in the NLP field, PHM researchers started considering the combination of multiple data-driven models using one ore more machine learning methods, especially deep neural networks. For example, Zhang et al. [113] proposed a novel RUL estimation method using a deep LSTM-Fusion technique. In this ensembling method, the model used multiple sub-LSTM-network-models, whose outputs were concatenated using a fully connected fusion layer and then weighted using a softmax layer to provide an RUL estimation as output. Experimental results showed that the proposed network outperformed seven state-of-the-art methods implemented for comparison (including MLP, SVR, CNN and shallow LSTM).

In recent years, several works combining data-driven learning models have been proposed to deal with the complexity, heterogeneity and manage noise in available data, in order to improve accuracy in predictive tasks. These approaches focus mainly on the pre-processing of data, including two major intermediate tasks: *feature extraction*, and *denoising*. As the data often comes from multiple sensors, data pre-processing methods are gaining in popularity among the PHM community, especially *features extraction*. Feature extraction, a method of dimensionality reduction, is the transformation of original data to a data set with a reduced number of variables, while still accurately describing the original data set. Feature extraction algorithms are useful to:

1. Reduce calculation, storage and data acquisition costs;
2. Improve learning by building less complex models;
3. Eliminate irrelevant variables that could give false predictions.

Hence, a special focus on feature extraction techniques will be given in this review. Although deep RNNs and deep CNNs are currently the most commonly used ML techniques in direct application for prognostics tasks (*e.g.* RUL estimation), other DNNs are used in multi-model approaches to fulfill pre-processing tasks. Those techniques, including *CNNs*, *Restricted Boltzmann Machines* and *Autoencoders*, will be detailed in the following.

Convolutional Neural Networks used in multi-model approaches As mentioned earlier, CNNs were initially introduced as classifiers, thus have been integrated into several multi-model approaches working in series as a discriminant or feature extractor in prognostics tasks. For example, Bao et al. [114] presented a novel method for RUL estimation applied to aero-engines, combining a Spatio-Temporal LSTM and a Spatio-Temporal CNN (ST-CNN). In their work, the LSTM was used to generate sequences, and the CNN used as a discriminator: if the discriminant result cannot distinguish between LSTM model and the real model, then it is considered as the 'True prediction'; otherwise, the LSTM model will regenerate the model until ST-CNN cannot discriminate between the generated model and the real model. In [115], the authors presented a deep neural network multi-model LSTM-CNN-FNN: CNN and LSTM for features extraction (spatial and temporal features), Feed-Forward neural network (FFNN) for RUL estimation. This method showed better results than the previous one proposed by the same authors [116] applied on turbofan engines.

Restricted Boltzmann Machines used in multi-model approaches Another feature extraction method used in multi-model approaches is based on Restricted Boltzmann Machines (RBMs). Introduced by Ackley et al. [117] in 1985, a Boltzmann machine (also called stochastic Hopfield network with hidden units) is a type of stochastic recurrent neural network. An RBM is a Generative stochastic artificial neural network that can learn a probability distribution from a training dataset. Deutch et al. [118] proposed a multi-model approach combining RBM and a linear regression layer for bearing RUL prediction applied on vibration data. In this study, the RBM was used to capture the "indicator" of the degradation (feature extraction) of the component over time, then included as a new feature for a linear regression technique to predict RUL. Deep Belief Networks (DBN), an unsupervised probabilistic deep learning algorithm, is composed of stacked restricted Boltzmann machines (RBMs) [119]. DBN is also an efficient model to learn powerful hierarchical feature representations from input data, and have been used in RUL estimation, showing competitive results [120], applied on turbofan engine. Ellefsen et al. [116]

proposed a multi-models approach for Remaining useful life prediction applied to aero-engines. In [116], the authors combined unsupervised and supervised learning: unsupervised Restricted Boltzmann Machines (RBM) and supervised LSTM for RUL estimation. Authors compared the proposed method with other selected multi-models published in the literature on the turbofan engine C-MAPSS dataset [9], and their results showed that the multi-model outperformed several purely supervised training methods.

Autoencoders used in multi-model approaches An Autoencoder is an artificial neural network that is often used in learning the discriminating features of a dataset in an unsupervised manner [121]. In recent years, Autoencoders have been successful in the application of machine health monitoring, especially in fault diagnostics and classification, and could be useful for prognostics in terms of noise reduction (denoising) and feature extraction. In 2018, Ren et al. [122] proposed a multi-model approach for bearing RUL prediction using deep autoencoders. The authors 1) first used a feature extraction method on the vibration signal data (deep AE for time domain features, Frequency Spectrum Partition Summation for frequency domain features, and Wavelet transform method for Time–frequency domain feature), then 2) used the deep autoencoders for feature compression, and finally 3) used a deep neural network for RUL estimation. According to the authors, the deep AE model presents two advantages, essential to deep learning:

1. the method can perform automatic feature selection
2. the number of network parameters can be reduced to avoid overfitting

The deep autoencoders have also been used here for time domain feature extraction. Even if this model seems to be complex (considerable calculation cost), results showed that, in this context, the proposed method outperformed three other data-driven methods including single-model DNN and SVM. Vincent et al. [123] introduced an improved version of a shallow AE named Denoising Autoencoder (DAE). The basic idea behind DAE is to reconstruct the original data from corrupted input, which helps to discover the robust representations and prevent it from learning the less important features, typically noise. According to the authors, the DAE are designed to capture more informative hidden patterns and obtain robust and powerful representations from the initial raw noisy data. Furthermore, DAE can generalize well and provide better outputs when it is stacked into a deep neural network. Ma et al. [124] introduced in 2018 a stacked sparse autoencoder (SSAE) for extraction, selection and denoising aeroengines features (RUL estimation for turbofan engines). The proposed SSAE is a multi-layer neural network consisting of autoencoders in each layer. It is composed of multiple layers including a sparse autoencoder (SAE) [125] and a denoising autoencoder (DAE) [123]. The stacked sparse autoencoder was used to automatically extract performance degradation features from multiple sensors on the aircraft engine; a logistic regression was then used to predict the remaining useful life. In 2017, Gugulothu et al. [126] proposed a multi-model based on RNNs for Health Index (HI) and RUL estimation able to handle

noise in sensors data: RNN Encoder-Decoder (RNN-ED), or RNN-Autoencoder [127]. RNN-ED can be considered as a type of multi-models because it is an implementation of a recurrent neural network for sequence data, embedding an Encoder-Decoder architecture (*i.e.* two standard RNNs stacked: an RNN model as an encoder, and another RNN as a decoder). In [126], the proposed model consisted of an RNN Encoder-Decoder used to generate embeddings which captured the trend of multivariate time series data and used the robust embeddings to construct the HI values in an unsupervised manner. The HI curve obtained was then compared with the HI curve of known failed instance to estimate the residual life. Malhotra et al. [128] presented an LSTM Encoder-Decoder for fault diagnostics and prognostics. The authors developed an unsupervised technique to obtain a health index (HI) for a system using multi-sensor time-series data, which allows them to capture the degradation in a system. The LSTM-ED was trained to reconstruct multivariate time-series corresponding to normal behavior of the system. The reconstruction error at a point in a time-series (calculated with the root mean squared error metric) was used to compute the HI at that point, which itself was then used for RUL estimation. Note that RNN-ED have an architecture that allows the model to "be used to both support variable length input sequences and to predict or output variable length output sequences" [129], which is sometimes useful when the length of the inputs or outputs may vary due to the heterogeneity of sensors data. Similarly, Yu et al. [130] proposed an improved RNN-ED method in their paper: the Bidirectional RNN-ED (BiRNN-ED). This approach was introduced by Schuster and Paliwa [131] in 1997 as an extension of RNNs, based on the idea that output at time step t may depend not only on previous elements of the sequence, but also on future elements of the sequence. From an architectural point of view, a BiRNN is simply two RNNs stacked on top of each other: the input sequence is fed in normal time order for one network, and in reverse time order for another. The outputs of the two networks are usually concatenated at each time step. According to Yu et al. [130], the Bi-RNNs can capture the complete and sequential information from a signal in the forward and backward manner, which can improve the reconstruction precision of the RNN-ED.

Table 2.2 summarizes the applications of the corresponding Machine Learning approaches in prognostics identified in this literature review.

2.4 Current challenges of the latest Machine Learning techniques in prognostics

In this section, existing related challenges of these techniques in prognostics are pointed out, including data scarcity and uncertainty. Some proposed solutions tackling these issues are also discussed.

Model	Tasks	Application
Support Vector Machines	RUL estimation	bearings [34]–[37], [42]–[44]; turbine engines [38]; aircraft engine components [46], [96]; li-ion batteries [75].
Shallow Neural Networks	RUL estimation	aircraft engine components [46], [96]; bearings [56]; turbofan engines [57], [96]; gas turbine engines [58]; fatigue crack growth [59], [62].
Recurrent Neural Networks	RUL estimation, features extraction	blade material's degradation [69]; bearings [70], [71], [91]; turbofan engines [72], [73], [86]–[88], [113]–[115]; aircraft engine components [96]; li-ion batteries [75], proton exchange membrane fuel cell [89]; battery management system [90]; fatigue crack growth [79].
Convolutional Neural Networks	RUL estimation, features extraction	turbofan engines [101], [114], [115]; rolling bearings [107]; fatigue crack growth [102].
Restricted Boltzmann Machines	Features extraction	turbofan engines [116], [120]; rolling bearings [118].
Autoencoders	Features extraction, denoising	turbofan engines [124], [126], [128], [130]; rolling bearings [122]; real-world pump dataset [126]; milling machine [128], [130].

Table 2.2: Summary of identified applications of Machine Learning approaches in prognostics.

2.4.1 Data scarcity

According to Fink et al. [14], the potential of the latest Machine Learning algorithms might not yet be fully exploited in PHM due to data scarcity, especially Deep Neural Networks. Indeed, their effectiveness depends on the quantity and quality of available labeled data. As mentioned in Section 2.3, a label can constitute the RUL at each time step of measurements, which is generally difficult to acquire and the corresponding data acquisition campaign often can be a time-consuming and expensive investment. Hence, the lack of available labeled data is becoming one of the most important challenges in PHM [14], [19]. Among the latest advances proposed in PHM to address data scarcity, the most frequently used techniques that have been identified in the literature are the following:

1. Data augmentation;
2. Pre-training techniques.

2.4.1.1 Data augmentation

Data augmentation [132] is a fully-supervised machine learning technique that is used to increase the amount of labelled data by adding newly created synthetic samples from existing labelled data. It can be considered as a multi-model approach combining:

1. a model A used to generate synthetic samples from available labelled samples,
2. a model B used for prediction task (such as RUL estimation in prognostics).

The purpose of the data-driven model A is to create many possible different input samples from the original labelled input samples, thus increasing the size of the labeled dataset in order to improve the performance of the model B. A diagram of the working of data augmentation approach is illustrated in Figure 2.13.

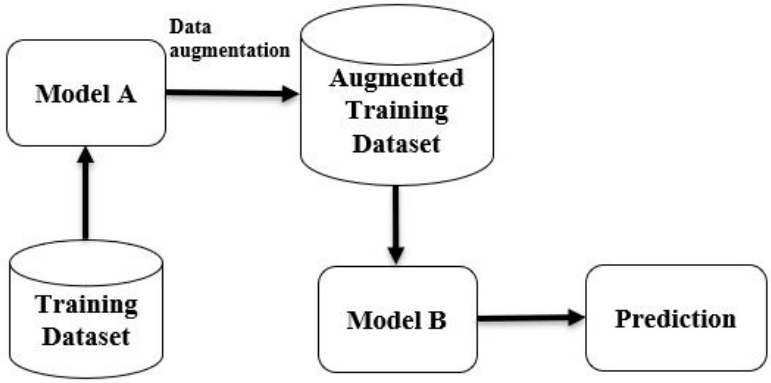


Figure 2.13: Illustration of the Data Augmentation process. The model A is used for data augmentation, and the model B used for prediction task.

Initially developed in the image processing field, data augmentation transforms existing available labelled images using some techniques such as adding Gaussian noise to images or region masking [132]. However, the RUL estimation problem is mainly considered as a Time Series Forecasting problem, and simply applying those data augmentation techniques from image processing may not result in valid synthetic time series data. Hence, effectively generating a large number of synthetic labelled data from time series remains a challenge, and has been investigated by few researches in PHM [133]. Kim et al. [134] used a Dynamic Time Warping (DTW) algorithm (an algorithm for measuring similarity between two temporal sequences [135]) for data augmentation applied on the NASA battery degradation data and on a synthetic dataset. Yuvapoositanon and Intachai [136] proposed a Singular Spectrum Analysis (SSA) [137] method for synthetic dataset generation method combined with an LSTM model for RUL estimation of turbofan engines,

outperforming other implemented generative algorithms such as Empirical Mode Decomposition (EMD) [138], the Fast Fourier Transform (FFT) [139] and the Fourier Decomposition Method (FDM) [140]. Ahn et al. [141] suggested a time-series data-generation method similar to SMOTE [142] in order to avoid overfitting and increase RUL prediction performances, applied on battery and bearing datasets. SMOTE, or (Synthetic Minority Over-sampling Technique, is one of the most widely used over-sampling techniques that can be used for data augmentation. It was initially designed for classification problems to combat data with imbalanced classes (*i.e.* if the categories are not equally represented), and it consists in creating new synthetic samples for the under-represented classes, rendering it impractical for high dimensional regression problems. Indeed, in [141], the time series were first discretized before data augmentation, using a symbolic aggregate approximation (SAX) technique [143]. However, there are existing adaptations of SMOTE for regression problems [144], and other SMOTE regression variants that have been applied on augmenting time series data, such as in [145] for sensors data. Behera et al. [146] proposed a multi-model approach for RUL estimation applied on C-MAPSS dataset, combining a conditional generative adversarial network (CGAN) and a deep GRU network. In their approach, CGAN model is used for augmenting the training dataset in order to improve the prognostics performance of the GRU model. Results showed that the suggested data augmentation technique provided better performances for the GRU model in estimating the RUL of the components, and outperforms other oversampling approaches implemented, including SMOTE and variants. CGAN was also used in [147] to generate real-valued failure data samples for prognostics of air purge valves; Lu et al. [148] used a GAN for data augmentation in order to improve the accuracy of an LSTM model in predicting RUL of bearings. As mentioned above, research for data augmentation in the prognostics field remains limited, since data augmentation techniques were initially developed for image processing field and mainly applied on classification problems. More specialized data augmentation techniques for time series regression problems will need to be developed for the prognostics tasks, some guidelines to tackle this challenge being provided in [149], [150]. Note that this technique may not always be the most suitable approach to address data scarcity in engineering fields such as PHM, as the synthetic data might not represent the events in the real world correctly, making the training data less accurate and causing more error in the failure prediction.

2.4.1.2 Pre-training

In the field of ML, several researchers have sought to address the challenge of data scarcity using an increasingly popular learning paradigm, called *pre-training* [151]–[153]. The *pre-training* technique typically consists in training a model on a source domain data in order to learn a nonlinear transformation of its input (*i.e.* learning abstract features). This learning paradigm has become a standard ML technique, especially in NLP [26], [154] or Image Processing [155]. Thus in the following, two such techniques are discussed: Transfer learning (TL) [156] and

Self-Supervised Learning (SSL) [157].

Transfer Learning Transfer Learning (TL) is a pretraining method [156], [158] that consists of transferring learned knowledge from one domain (*i.e. pre-training* on a source domain data) to another domain (*i.e. fine-tuning* on a target domain data). Transfer learning is mainly used for the purpose of improving the prediction models on the target domain when a small number of labelled samples is available. It can be considered as a DD multi-model approach, since it consists of the combination of a:

1. DL model A pre-trained on a dataset A (*i.e.* features extraction on the source domain)
2. A data-driven model B fine-tuned on a dataset B (*i.e.* target domain) for a related task, using the features extracted by the pre-trained model.

Note that the data-driven model B can sometimes correspond to the same pre-trained model A. Indeed, a DL model can be pre-trained on a source domain, then refined on a target domain if the predictive task remains similar [159]. The working of TL technique can be illustrated in Fig. 2.3.

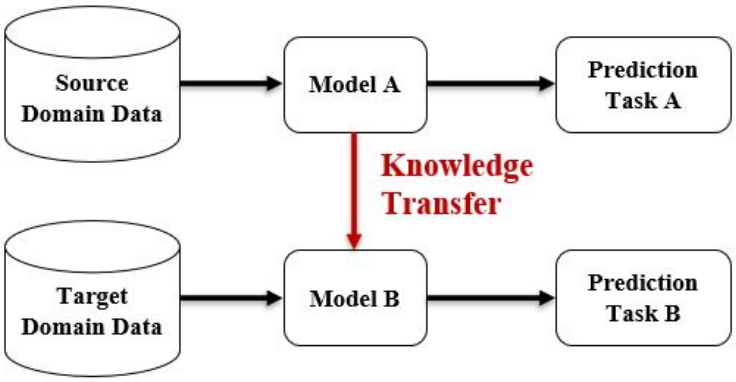


Figure 2.14: Illustration of the TL process. The model A is pre-trained on source domain data for features extraction (*i.e.* prediction task A), and the model B used for prediction task B on target domain data.

There are several categorization criteria of transfer learning. As in [156], [160], TL applications are divided into three categories in this review:

1. *inductive transfer learning*: the source and target domains are the same, while the source and target tasks are different but related. Inductive TL requires the availability of the target data labels.

2. *transductive transfer learning*: both domains data are different but related (*i.e.* different feature spaces or input data distributions). The source and target tasks are the same. Transductive TL requires the availability of the source data labels, while the target data are unlabelled.

3. *unsupervised transfer learning*: the source and target domains are different but related, as well as both predictive tasks. Both source and target data are unlabelled.

Illustrated in Table 2.3, more details about these three TL categories are given in [156]. Note that in unsupervised TL, both domains are not labeled, thus to the best of author's knowledge, there are no or little existing studies that focuses on unsupervised TL approaches for the prognostics field in PHM.

Transfer Learning	Source and Target Domains	Source and Target Tasks	Source Domain Labels	Target Domain Labels
Inductive	Similar	Different but related	Available/Unavailable	Available
Transductive	Different but related	Similar	Available	Unavailable
Unsupervised	Different but related	Different but related	Unavailable	Unavailable

Table 2.3: Summary of Transfer Learning categories.

Zhang et al. [159] proposed an inductive TL approach using a Bi-LSTM model for estimating RUL, applied on the C-MAPSS dataset. The authors trained the Bi-LSTM over the source domain data with a large amount of labelled data, then refined by training it over a smaller amount of target data, showing in most cases better performances than standard Bi-LSTM model without transfer learning. Inductive TL also refers to the situations in which no labeled data in the source domain are available, similar to the *self-taught learning* [161]. Sun et al. [162] pre-trained an SAE model on historical failure tools data in an unsupervised manner to transfer learning to new tools for RUL prediction; in [163], the authors used a denoising autoencoder to extract features during an off-line stage, then combined with an SVM regression model for RUL estimation of bearings during an online stage. Fan et al. [164] proposed a multi-model approach combining a pre-trained Self-Organizing Maps (SOM) and a Random Forest regression model for RUL estimation, applied on the C-MAPSS dataset. The Self-Organizing Maps, or Kohonen Maps [165] is a type of unsupervised DNN and an excellent technique in exploratory phase of data mining. They are used to map a real space, i.e. to study the distribution of data in a large space. In PHM, the SOM has been mainly used in diagnostics tasks but some applications to prognostics also exist. Montero-Jimenez et al. [166] proposed an SOM model to identify the degradation trend on simulated aircraft jet-engines, and suggested that the degradation indexes obtained from the model can be used for the estimation of the Remaining Useful Life in prognostics. Indeed, in

[164], the SOM model was used for features extraction and pre-trained on the source domain.

According to Zhang et al. [159], Transfer Learning might be a promising solution for the lack of labeled data issues, but their results in prognostics showed that the knowledge transfer can have detrimental effect over target task (*i.e.* negative transfer) if the distribution of source data and the distribution of target data are not close enough [167]. Several researchers have investigated a technique in order to overcome this limitation: Domain Adaptation (DA) [168]. DA is an ML technique that aims to reduce the domain shift or the distributional difference between the source and target domains by creating domain-invariant features [168]. This technique has been mainly used in transductive TL problems (*i.e.* both domains data are different while the predictive tasks are similar), to improve the performance of the target task. Some of the most common (unsupervised) DA techniques showing improvements for TL in the prognostics field are the following:

1. Transfer Component Analysis (TCA [169]) is an unsupervised method that aims to reduce the data discrepancy between both domains [163], [170]–[172];
2. CORrelation ALignment (CORAL [173]) is a metric that consists of minimizing the covariance shift between the source and target features [171];
3. Maximum Mean Discrepancy (MMD [174]) is a popular nonparametric distance metric used in DA, used to minimize the distribution discrepancy between source and target domains, and that has been extensively used in fault prognostics [170], [172], [175]–[178].

In [171], Da Costa et al. proposed a DA method based on adversarial training [179]. Adversarial training is a DL technique that consists of training neural network architectures on labeled source data and unlabeled target data in order to discriminate between the two domains and learn domain-invariant features. In their Domain Adversarial Neural Network (DANN) approach [171], an LSTM model was used to extract temporal features from labelled source domain and a DANN to learn domain-invariant features to predict the RUL in the unlabelled target domain, applied on the C-MAPSS dataset. Results showed that 1) adversarial training in DA was able to improve predictive tasks in turbofan engine RUL estimation compared to non-adapted models using TL, 2) TL with DANN was able to obtain performances competitive with SOTA results in the C-MAPSS dataset, and 3) achieved better performance compared to two other DA methods: TCA and CORAL. Domain Adapation with adversarial training appears to be a promising approach for DA in TL, and there have been other studies on Adversarial training for DA in prognostics proposed in recent years [172], [180]–[182].

Self-Supervised Learning A recent alternative to TL and emerging pre-training technique to overcome this lack of labeled data is Self-Supervised Learning (SSL) [157].

Similarly to *self-taught learning* as presented in [161], Self-Supervised Learning (SSL) is a sub-category of unsupervised learning that consists in learning meaningful and general representations from unlabelled data (during the *pre-training phase*) by solving a so-called *pretext task* without requiring the data to be labelled. These representations are applicable to a wide range of related supervised tasks (*i.e. downstream task*) with only few labelled data (*i.e.* "Few-Shots Learning"). This learning paradigm is typically composed of:

1. *a pre-training phase*: a DL model is trained on a raw unlabelled dataset in an unsupervised (or *self-supervised*) manner in order to learn abstract features. The unlabelled dataset is expected to be large enough to learn a general and useful representation for a related tasks.
2. *a fine-tuning phase*: the hidden layers of the pre-trained DL model are coupled to a data-driven model (*e.g.* a linear layer or a predictive model) and then trained on a set of labelled data in a supervised manner.

The working of SSL can be illustrated in Figure 2.15.

Self-Supervised Learning aims to improve state-of-the-art results using unlabelled data, due to its ability to avoid extensive cost of collecting and annotating large-scale datasets [184]. However, to date, there is, to the author's knowledge, only a limited amount of existing research that focuses on the application of SSL to prognostics problems in PHM, for example for RUL estimation on the NASA C-MAPSS dataset [116], [185], [186]. One of the first to explore this learning paradigm in prognostics in order to deal with the problem of lack of labelled data is Yoon et al. [185], using a pre-trained variational autoencoder (VAE [187]) that makes use of available unlabelled data to learn latent space representation in an unsupervised manner; the pre-text task is minimizing reconstruction error. The extracted features by the VAE model are then fed as inputs to an RNN model for RUL estimation, trained in a supervised manner by varying the fraction of labelled engines data down to 1% in order to investigate if the SSL approach can enhance predictive tasks when only a small amount of labelled data is available. Results showed that their proposed method was able to outperform a not pre-trained supervised RNN-model when all the labelled dataset is available as well as in other scenarios when the available labelled data is highly limited with only a small labelled fraction of the training data. The authors in [116] used a Restricted Boltzmann Machine model (RBM) [119] for pre-training on unlabelled dataset with a reconstruction pre-text task, and an LSTM model for RUL prediction. Results showed that this SSL approach could improve the RUL prediction accuracy compared to the purely supervised learning approach (*i.e.* predictive model without the initial pre-training stage), both when the training data is completely labelled and when the labelled training data is reduced. It is worth noting that the methods proposed in [185] and [116] were not presented as SSL approaches, but are considered as such in this review since the proposed methods follow the same procedure as

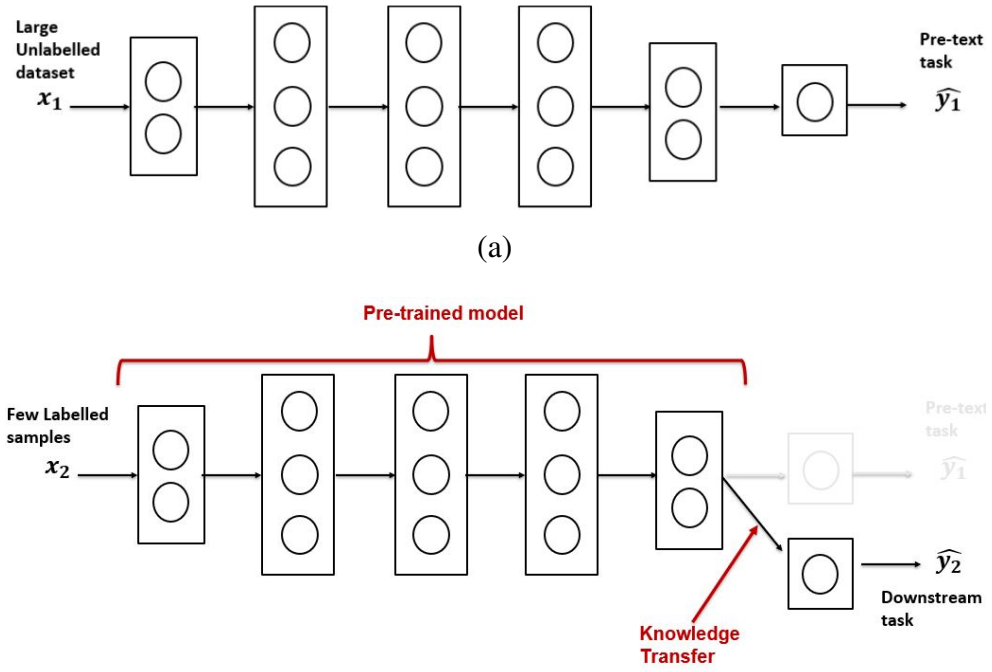


Figure 2.15: A schematic view of the Self-Supervised Learning procedure (a): Pre-training phase in a self-supervised way. The most widely used pre-text tasks are Contrastive learning (learn similar or dissimilar representations from source data [157]) and Autoregressive Predictive Coding (predict the value of the next timesteps [183]), (b): Fine-tuning phase (supervised training on downstream tasks).

described before. However, Krokotsch et al. [186] highlighted two shortcomings of these two previous studies:

1. these approaches were evaluated only on one subset of the C-MAPSS dataset out of four in each study, rendering these investigations limited;
2. pre-training was performed on unlabelled data of engines that contain the point of failure, which should not be the case in real scenarios, since the RUL labels for all the data could be deduced based on the knowledge of the failure time.

To overcome these limitations in [186], the investigation was performed over all subsets of the C-MAPSS data set and the unsupervised pre-training phase was performed over truncated time series, assuming that realistic unlabelled data does not contain features near the time of failure (corresponding to sensors data of structures replaced before reaching failure). Results showed that:

1. the proposed SSL approach can outperform the supervised baseline that used only the labelled data. Both approaches were trained on only few labelled time series for RUL estimation (*i.e.* Few-Shots learning);
2. the proposed pre-training model outperformed two competing pre-training models, including AE and RBM using a reconstruction pre-text task (*i.e.* the output y corresponds to an estimation of the input x).

These results suggest that the choice of the pre-training model (or pre-text task) matters. Unfortunately, there are no clear guidelines for selecting the right pre-text task that learns meaningful representations from unlabelled time series data (*e.g.* sensors data) during the pre-training phase. One of the main challenges for extensive investigations on the potential of SSL in PHM resides in the difficulty of having scalable open-source dataset, similar to those available in Natural Language Processing or Image Processing. Thus, despite demonstrating encouraging results, the domain of SSL is still largely unexplored in the prognostics field and is in contrast with the increasing amount of unlabelled data available, having the potential to enable predictive maintenance.

2.4.2 Uncertainty

Uncertainty remains a challenging task and ubiquitous in real-world applications. In this review, the author distinguishes two different sources of uncertainty: epistemic and aleatoric uncertainty

[188]. Aleatoric uncertainty, also known as *data uncertainty*, arises from the intrinsic stochasticity of data distribution (*e.g.* inherent noise in sensors data or measurement errors). Epistemic uncertainty, also known as *model uncertainty*, comes from a lack of data or a limited understanding of the model built (including uncertainty over the parameters of the model). Both epistemic and aleatoric uncertainties tend to produce variability in the outcome of the predictive models, which can be harmful in risk-sensitive applications like prognostics in predictive maintenance. On the one hand, epistemic uncertainty can be reduced through more comprehensive studies (*e.g.* if the model is refined or enough training samples are given). On the other hand, aleatoric uncertainty is less prone to reduction due to the intrinsic variability in the data (mainly due to the work environment or the limitation of the measurement devices), but can be identified and quantified [189], [190]. Figure 2.16 illustrates the different types of uncertainty sources. Although challenging, it remains crucial to address this challenge in prognostics, by uncertainty quantification (UQ), in order to improve the reliability in making maintenance decisions.

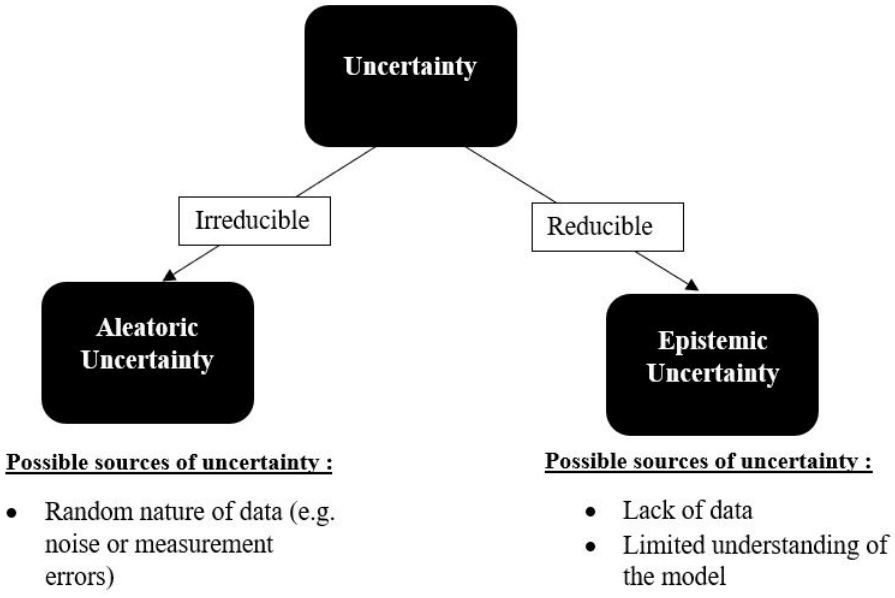


Figure 2.16: Types of uncertainty sources.

The latest ML algorithms (especially DNNs) have achieved SOTA performance in prognostics, but single-model approaches were initially developed with the only aim to improve the prediction accuracy without additional insight that quantifies predictive uncertainty (*e.g.* confidence intervals associated with predictions). According to Abdar et al. [191], several techniques have been proposed over last decade to address this challenge in machine learning, and the most widely used methods for handling uncertainty in machine learning are Bayesian techniques [192] and Ensemble Learning techniques [193].

2.4.2.1 Bayesian methods

Bayesian methods are statistical inference methods applying Bayes' theorem [192], used to estimate the conditional probability of an hypothesis based on the observation of known events. This model class can describe complex stochastic patterns in the data via a distribution over latent input variables, allowing the quantification of the aleatoric uncertainty of the data. An immediate advantage of specifying distributions on model parameters and predictions is that Bayesian inference will find the most suitable explanation for the model parameters given the available data. Bayesian methods are particularly useful when data is scarce (*i.e.* in cases of epistemic uncertainty), as it quantifies uncertainty (which is typically high when the learning dataset is small). These approaches can also be an effective way of preventing neural networks from overfitting [194]. Moreover, for real system as in PHM applications, the true system state can hardly be observed directly. Hence, Bayesian methods has been actively developed and are certainly the most active category of approaches currently in PHM and RUL estimation of machine components.

Liu et al. [195] presented a novel SVR method based on the Bayesian probabilistic paradigm to predict nuclear power plant components RUL. Tipping [92] introduced a Bayesian form of SVM models, called the RVM (Relevance Vector Machine). The RVM, also part of the family of stochastic models, provides an accuracy comparable to SVM learning machines but in addition provide probabilistic output, using Bayesian inference. According to [46], adding a Bayesian framework to SVM models, allows the capability of handling uncertainty in prognostics. They also pointed out the ability of RVMs to detect underlying trends in noisy data with the use of their Bayesian framework, allowing them to take into account aleatoric uncertainties in the field of application. However, as mentioned in Section 2.3.1.1, the application of SVMs remains limited when dealing with complex and heterogeneous data.

Over the last decade, several researchers in the prognostics field have also aimed to embed the Bayesian framework to Deep Neural Networks [196]–[201], due to their ability to deal with large datasets unlike standard ML models such as SVM. Among the latest advances proposed in

PHM to quantify uncertainty, the Bayesian methods that have been identified are the following: 1) Variational Inference, 2) Monte-Carlo Dropout.

Variational Inference So far, one the most commonly used Bayesian method in the prognostics field is Variational Inference (VI) [202]. VI-based methods aim to approximate posterior distributions over the parameters of the model (or *weights* [203]), and outputs two values from a supposed distribution (*e.g.* Gaussian distribution): the mean vector of the distribution μ and the standard deviation vector σ.

Motivated by VI methods, Wang et al. [196] proposed a Bayesian CNN-based method (a deep CNN model using VI) for RUL estimation on the PRONOSTIA dataset with uncertainty quantification. Their results showed that the proposed method could achieve competitive results with three other implemented data-driven methods, including deterministic deep CNN that only provide a pointwise estimation of the RUL without uncertainty quantification. Caceres et al. [198] presented a probabilistic Bayesian RNN for RUL prognostics on the NASA C-MAPSS dataset, using VI in order to address aleatoric uncertainties; Huang et al. [204] combined an LSTM-ED model to extract features and a neural network using VI for RUL estimation on C-MAPSS dataset. Gao et al. [200] used a variational RNN-EncoderDecoder (RNN-ED) for RUL estimation of turbofan engines. Inspired by the Variational Autoencoder (VAE) architecture [187], the proposed RNN-ED model casts learning representations of input data as a VI problem, where the hidden representation is a latent variable supposed to follow a Gaussian distribution. Benker et al. [199] compared Bayesian DL models (MLP and CNN models) with their non-probabilistic counterparts trained for RUL estimation on the NASA C-MAPSS dataset. The Bayesian DL models were implemented and trained with VI and another Bayesian method that has been used to approximate the output distribution: Hamiltonian Monte Carlo (HMC) [205], a variant of a class of algorithms named Markov chain Monte Carlo (MCMC) [191]. The authors demonstrated that adding a Bayesian framework to a singe-model's architecture allows to enhance their predictive accuracy, with in addition an uncertainty quantification of the predicted result. Also, Benker et al. [199] highlight that nowadays integrating a Bayesian framework no longer requires significant additional costs since modern softwares enable to benefit from latest algorithmic advances in Bayesian inference. Indeed, there are more and more flexible and scalable open-source softwares that unify the latest deep learning techniques and Bayesian modeling, such as Pyro [206] and ZhuSuan [207].

Monte-Carlo Dropout In 2016, Gal and Ghahramani proposed a novel Bayesian approximation method and an alternative approach to VI for uncertainty quantification in Deep Learning: Monte Carlo Dropout (MCD) [208]. The MCD is an extension of the regular Dropout approach [194], a learning process which, during the training of the DL model, will zero out some randomly chosen

weights in order to regularize the model and reduce overfitting. The MCD approach consists in applying the dropout at both training and testing steps, hence will change the structure of the network and can generate multiple different predictions as samples from a probability distribution to quantify the uncertainty, using a Monte Carlo sampling method, rendering it equivalent to the VI process. A scheme of MDC approximation procedure is illustrated in Figure 2.17.

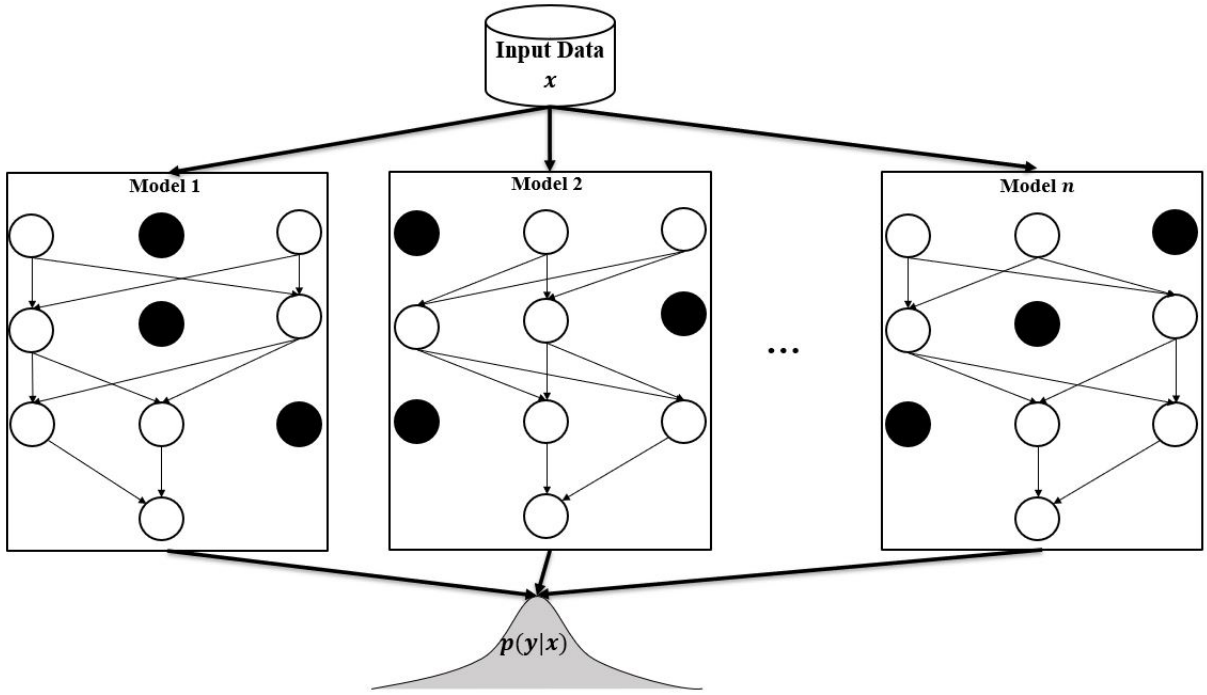


Figure 2.17: A schematic view of the Monte Carlo Dropout approximation procedure. During the prediction phase, the dropout is randomly applied n times to the intial model, producing n models with the same initial architecture but different nodes activated (the nodes in black are those whose weights have been zeroed out), thus generating n different predictions as samples from a probability distribution.

Wei et al. [209] combined MCD and a GRU model for RUL estimation of lithium-ion batteries. Through Monte Carlo sampling, the probability distribution of the prediction results is obtained, providing a 95% confidence interval in order to quantify the uncertainty of the prediction model. Peng et al. [210] integrate a VI-framework and MCD over a Bidirectional LSTM for residual life estimation applied on turbofan engines and ball bearings. Xiao et al. [211] proposed a DNN model using MCD for RUL prediction of Insulated gate bipolar transistors (IGBT); Wang et al. [204] combined a Sparse Autoencoder model to extract features and a DNN model using MCD for RUL estimation of a Proton exchange membrane fuel cells (PEMFC) system. In [208], results showed that MCD was able to outperform other Bayesian methods including VI in most cases, on both accuracy and uncertainty quantification. Biggio et al. [212] used a MCD approach for RUL

estimation with uncertainty quantification on the C-MAPSS dataset, showing competitive performance both in terms of accuracy and UQ with a deterministic FFNN model and variants of Deep Gaussian Processes (DGP) [213], [214]. DGPs are multi-model approaches that combines the benefits of DL models and the Bayesian framework of Gaussian Process (GP) regression, in order to incorporate the ability to quantify the associated uncertainty. DGP appears to be a promising approach in prognostics, and there have been other studies on DGP for quantifying uncertainties in prognostics, such as [215], [216]. Although achieving competitive performances with other Bayesian techniques such as VI, Kraus and Feuerriegel [201] highlight that computational costs for MCD approaches are lower compared to the costs when utilizing VI methods, in particular by using fewer parameters since they do not require modifying existing model architectures and doubling their number of parameters (*e.g.* VI methods that have a mean and a variance as outputs).

2.4.2.2 Ensemble Learning

Ensemble Learning [193] is an alternative to Bayesian methods for UQ in machine learning, that can be considered as a frequentist approach [217] since it only depends on the available data and can be performed when the distribution of the dataset is unknown. It is a statistical multi-model approach in which, instead of trying to optimize a single-model to improve its accuracy, it emphasizes the formation of a large number of single-models (*i.e. meta-learners*) and then combines them in order to obtain an improved precision ensembling model (*i.e. meta-model*). Ensemble learning technique is illustrated in Figure 2.18. Although it was initially used in order to improve the performance of predictive models, Ensembling techniques can also be used to quantify uncertainty (notably epistemic uncertainty), since combining multiple different predictions can also generate distributional estimates of model uncertainty. Note that MCD approach, although was introduced as a Bayesian approximation method, can also be considered as an averaging ensemble method, since it combines the output of several sub-models with shared weights.

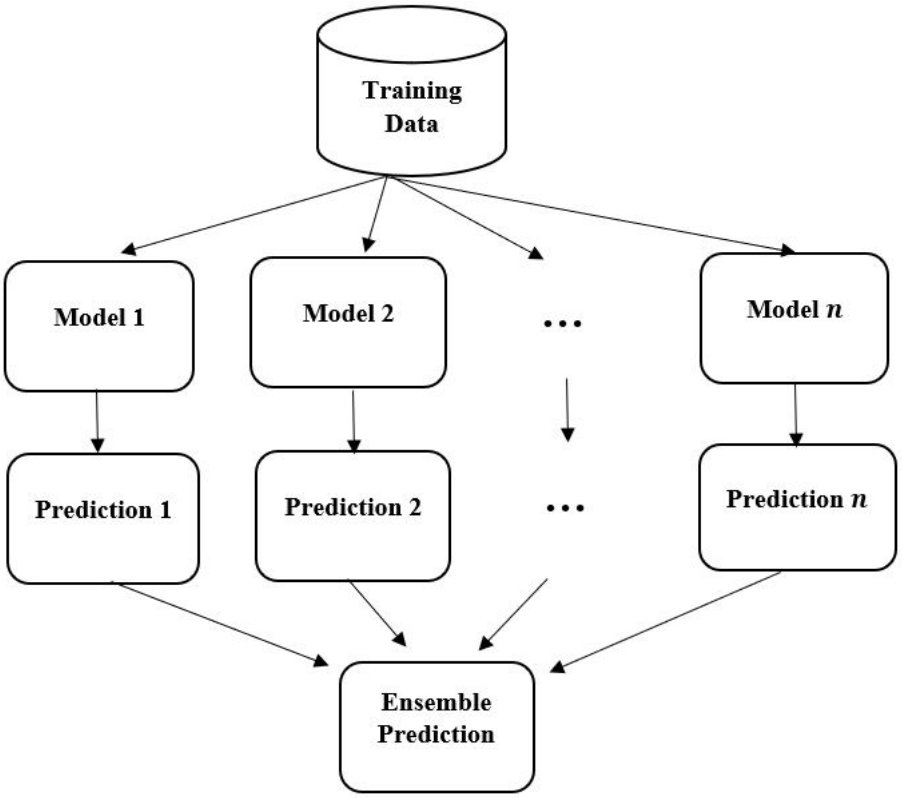


Figure 2.18: Illustration of the ensembling technique process combining n predictive single-models.

Vishnu et al. [218] proposed a multi-model combining different LSTM models as meta-learners for RUL estimation on the NASA C-MAPSS dataset with uncertainty quantification. In their proposed approach, the predictive single-models have the same architecture, but use different initializations of the parameters (weights and biases), and use random shuffling of the samples during the training phase, in order to obtain different trained models in an ensemble. The resulting predictions are then averaged to get their mean value and computed to get their empirical standard deviation, in order to quantify uncertainty in the RUL estimates. This technique is relatively similar to Bagging methods [219], which is an ensembling technique that has been used extensively in many real-world applications for UQ [220]. Also known as bootstrap aggregation, the Bagging method combines Bootstrapping [221] and Aggregation to form one ensemble model. In bagging, the training dataset is divided into a set of training subsets by random sampling with replacement (Bootstrap sampling), where each subset is used to train a separate single-model independently, and the results of each model are aggregated to form an ensemble prediction (Aggregation). The procedure of the Bagging technique is illustrated in Figure 2.19.

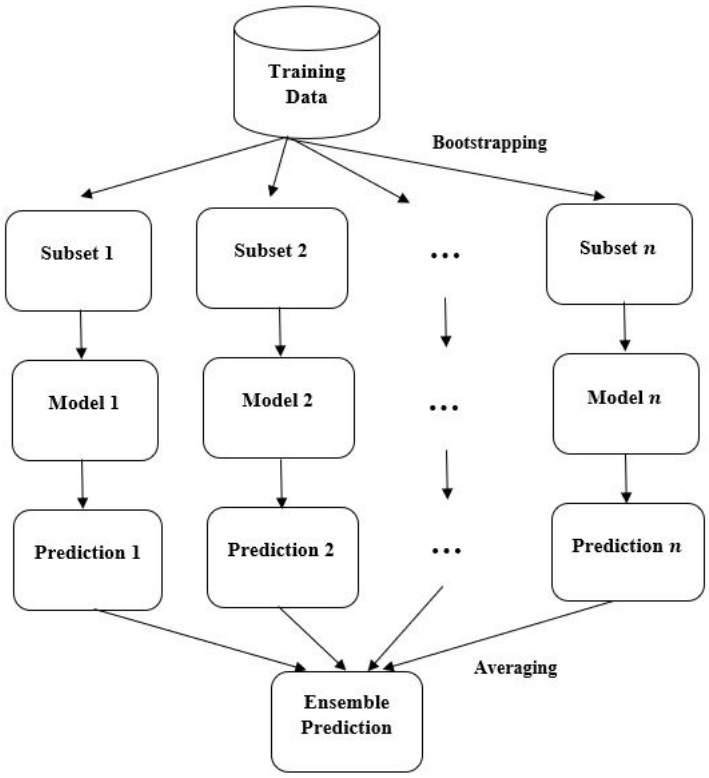


Figure 2.19: Illustration of the Bagging technique process combining n predictive single-models and using n training subsets. The subsets are obtained by random sampling with replacement.

Liao et al. [222] proposed the ensemble of multiple LSTM-FFNN models based on bagging method for RUL estimation applied with UQ on the NASA C-MAPSS dataset. Results showed that the proposed method achieves competitive results with other state-of-the-art data-driven approaches for RUL estimation, including deterministic deep CNN and deep LSTM, but also provides interval estimation for UQ. Cornelius et al. [223] suggest an ensembling method combining a bagging technique using randomly initialized CNN-MLP models and a VI method for RUL estimation with UQ on the same dataset. The authors used the bagging technique to estimate epistemic uncertainty by treating RUL predictions as random variables, and the VI method to quantify aleatoric uncertainty, since each predictive model generate two outputs rather than only predicting RUL: the mean and the variance of RUL estimates, assuming the target RUL probability distribution is Gaussian. Ensemble Learning is a promising technique that has the ability to enhance the performance of predictive models and produce uncertainty estimates, however only few related studies have been conducted in the prognostics field as their main drawback is that they require relatively large computational and memory costs for high-dimensional data sets.

2.5 Summary and Discussion

To summarize the main findings based on this state-of-the-art review, it is necessary to come back to the research questions that initially motivated it:

RQ1 : What are the current trends in prognostics for predictive maintenance ?

RQ2 : Which data-driven Machine Learning techniques are most commonly used to address failure prognostics, and what are their main key strengths and limitations ?

RQ3 : What are the existing related challenges and current research areas ?

2.5.1 RQ1: What are the current trends in prognostics for predictive maintenance ?

This chapter surveyed the current trends in prognostics for Predictive Maintenance. Prognostics approaches were categorized here in terms of: Knowledge-based, Physics-Based and Data-Driven models. The choice of the best suited approach to use in prognostics depends notably on the knowledge about the degradation mechanism and the available data.

As explained in Section 2.3, Data-driven approaches appear promising for complex systems, and especially the emerging Machine Learning models (such as Deep Neural Networks). Data-

driven techniques employ historical data to construct a statistical, stochastic or machine learning based model aimed at implicitly capturing the degradation process and estimating the RUL of a system.

Machine Learning models require only that monitoring data be available on the system, which is almost always the case. Moreover, unlike other prognostics approaches, these approaches do not require an understanding of the physical principles of degradation nor expert knowledge, and can potentially model complex dynamic (non-linear) and multivariate processes. Among Machine Learning techniques, Deep Learning (DL) has become a rapidly growing research direction, redefining state-of-the-art performances in a wide range of areas, especially in prognostics for the following reasons identified in the literature review:

1. they have proven to perform well on a variety of applications;
2. they are able to deal with significant complexity and nonlinearities in data;
3. they are applicable to multiple tasks (including residual life prediction, features extraction, noise reduction).

2.5.2 RQ2: Which data-driven Machine Learning techniques are most commonly used to address failure prognostics, and what are their main key strengths an limitations ?

In Section 2.3, the current Machine Learning techniques in prognostics and their applications have been identified, with their main strengths and limitations. In this chapter, a distinction was made between single-model approaches and multi-model approaches that seek to combine several individual techniques. Among single-model approaches, SVM and ANNs (notably Deep Neural Networks composed of RNNs and CNNs) are the most commonly used algorithms for RUL estimation in prognostics. Fig. 2.20 summarizes the most commonly used machine learning techniques for failure prognostics identified in the literature and presented this chapter.

SVM have been widely investigated in the prognostics field for RUL estimation over the last two decades; the reader can refer to [48] for more details. The main advantage of this technique is that it allows accurate models to be built when there is a lack of failure data. However, the algorithm's parameters need to be specifically tuned which might be difficult without any knowledge of the case study [46]. These models also seem to be not suitable to deal with large datasets [47].

Compared to SVM, the use of artificial neural networks for prognostics has several advantages. The main one is that an ANN makes it possible to model a complex, multidimensional and

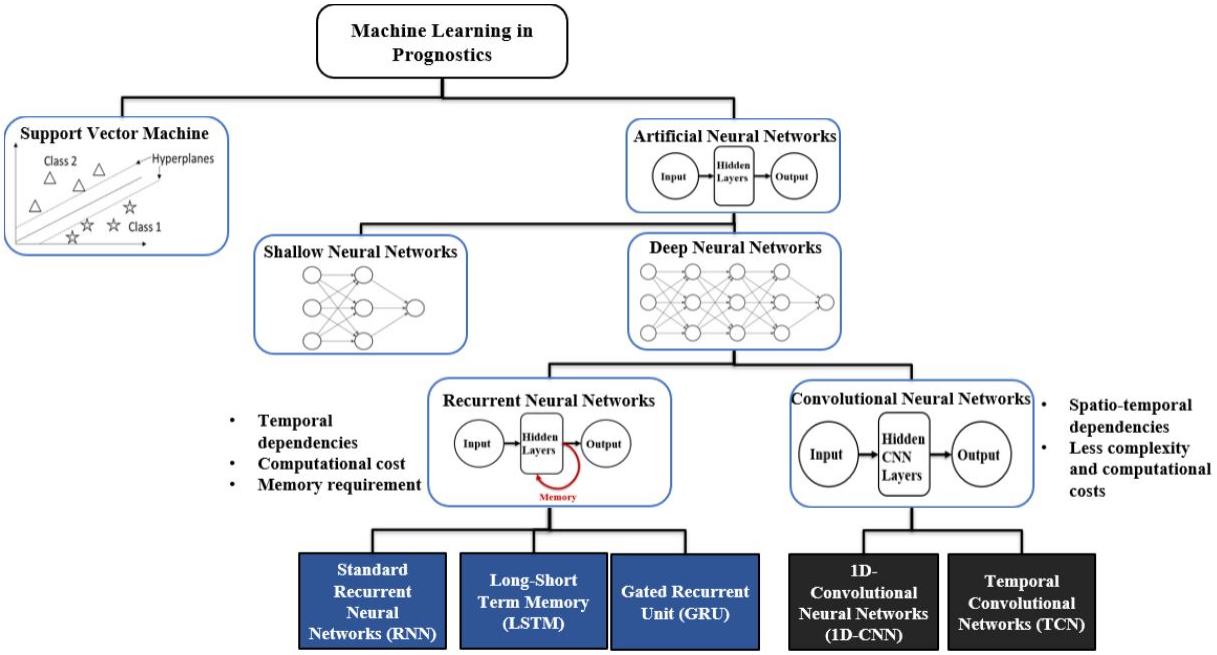


Figure 2.20: The most commonly used machine learning techniques for failure prognostics identified in the literature.

nonlinear system without requiring a physical modeling of its behaviour. According to An et al. [62], Neural Networks are appropriate for the case of high noise and complex models with large amounts of training data. ANNs are also well suited if the physical model is not known or impractical to apply (note that this can also be seen as a disadvantage if interpretability is being sought). Among artificial neural networks, it has been shown that DNNs are more suitable for handling complex and heterogeneous data than SNNs. However, it would be advisable to avoid using neural networks when the size of the available data is small, because a complex model tends to adjust excessively to the training data, which might lead to an overfitting and a lack of generalization. Sikorska et al. [53] states that selecting the best hyperparameters of ANNs remains a challenging and time-consuming task. Note that there are multiple machine learning single-models and variants that have been developed in recent years for prognostics tasks. A further interesting approach identified is to combine two or more single-model algorithms for prognostics, using the strengths of one model to complement the weaknesses of another: multi-model approaches.

As presented in Section 2.3.2, multi-model approaches appear to be a promising way forward, and have been massively investigated during the last years by PHM researches in order to overcome the complexity of predictive maintenance tasks and increase prediction accuracy. Some examples have shown that multi-models combining the prediction of several data-driven models are able to achieve higher prognostics accuracy than single-model approaches. Also, other variants of Deep Learning models, such as CNN, RBM and Autoencoders have been investigated and used

mostly for preprocessing tasks (*e.g.* features extraction mainly and sometimes for denoising tasks). These multi-model approaches that integrate a preprocessing module offer a potential solution to overcome complexity and heterogeneity of sensors data in predictive maintenance systems, thus able to increase prognostics accuracy.

Overall, several presented examples show that the multi-model approaches can provide more accurate prognostics prediction than state-of-the-art single-model approaches in the field. However, the most accurate model remains sometimes too expensive to run on a computer with limited resources. It results in a clear trade-off between the performance of the latest ML algorithms and their computational complexity [224]. Indeed, in the reviewed articles in this chapter, ML models were mainly compared using performance metrics such as accuracy for classification problems and Mean Squared Error (MSE) for regression problems. However, other metrics that are important for real-world applications, such as time and memory complexity (hence environmental impact), were rarely taken into account in the comparison. In this regard, it would be interesting to discuss the environmental impact of the latest ML techniques and the trade-off between the performance gain and the computational resources required. This highlights the need to consider a broader set of metrics when evaluating ML models, especially in the context of real-world applications where the computational resources and environmental impact can play a crucial role in the selection of models.

2.5.3 RQ3: What are the existing related challenges and current research areas ?

With the increasing availability of data for PHM, DL techniques are now the subject of considerable attention for this application, often achieving more accurate RUL predictions. However, their performance can sometimes be limited. Thus, Section 2.4 aimed to identify some of the current challenges of these techniques in prognostics as well as the existing opportunities, and can be summarized as follows:

- *Data scarcity*: One of the major existing challenges for the application of the latest Machine Learning algorithms in the prognostics field is that their effectiveness depends on the quality and quantity of available labelled data. This requires a monitoring system as well as learning time, which is an additional investment, both financial and temporal. Combining data-driven models with Machine Learning techniques such as Data Augmentation, or pre-training techniques seem to be promising to address this challenge. On the other hand, due to the advancements in sensing technologies in engineering fields such as predictive maintenance, the availability of unlabelled data is increasing (*e.g.* raw sensors data of structures replaced before the end of their service life). Self-Supervised Learning offers a potential solution to overcome data scarcity and would be able to benefit from large unlabelled data. Despite

demonstrating encouraging results as described in Section 2.4, the domain of SSL is still largely unexplored in the prognostics field and requires more research given the increasing amount of unlabelled data in PHM.

- *Uncertainty*: Due to several unavoidable factors such as random nature of data or limited source of data, uncertainty remains a challenging task and ubiquitous in real-world applications. Uncertainties tend to produce variability in the outcome of the predictive models, which can be harmful in risk-sensitive applications like prognostics in predictive maintenance. Thus it remains crucial to address this challenge in prognostics, by uncertainty quantification (UQ), in order to improve the reliability in making maintenance decisions. Multi-model approaches such as Ensemble learning or integrating a Bayesian framework to Deep Neural Networks may be a promising solution to tackle this challenge. Moreover according to [225], adding a Bayesian framework to Neural Networks allows the ability to handle uncertainty and also overcome the problem of overfitting [226]. These approaches are commonly called Bayesian Neural Networks (BNNs). Furthermore, nowadays there are more and more flexible and scalable open-source softwares that unify the latest deep learning techniques and Bayesian modeling, such as Pyro [206] and ZhuSuan [207].

Note that these two challenges are more or less related: epistemic uncertainty is a reducible error and arises in areas where there is a lack of available labelled data (*i.e.* data scarcity), or if the used model is too complex to understand by knowledge experts. Thus uncertainty is prone to reduction by reducing data scarcity. Data augmentation, which is a technique used to address data scarcity, can also be used to reduce epistemic uncertainty by increasing the training data size, but also to estimate the aleatoric uncertainty of deep neural networks by obtaining a predictive distribution [227]. Moreover, although SSL is now considered as an increasingly popular technique to tackle data scarcity, it is also considered as a promising new direction to tackle uncertainty according to Yann LeCun [228].

2.6 Conclusion

This chapter presented an overview and comparative analysis of current state-of-the-art machine learning methods in prognostics. Moreover, several existing related challenges were reviewed, and some future research directions in the field were presented. The author believes that this review focusing on the application of the latest ML techniques in prognostics will benefit researchers and may be considered as a possible guide to the reader in navigating this fast-growing research field.

As mentioned in this chapter, the lack of available labelled data (*i.e.* data scarcity) is becoming a major challenge in the application of machine learning to prognostics. According to the study findings related to this challenge, two main limitations in the prognostics field have been identified:

1. On one hand, the effectiveness of the latest ML algorithms depends on the quantity and quality of available labeled data. Hence, due to data scarcity in engineering fields such as in predictive maintenance, their potential might be not fully exploited, especially for Deep Neural Networks. Large, open-source datasets are limited and do not cover all potential engineering domains. This makes it difficult to evaluate and compare the latest DL models in some of these domains (for example in fatigue damage prognostics).

2. On the other hand, while labelled data is lacking, the availability of unlabelled data is increasing due to advances advancements in sensing technologies (e.g. raw sensors of structures replaced before failure). Hence in order to address data scarcity, exploiting unlabeled data during training has become a major goal in ML.

A limited amount of existing research in the prognostics field has sought to overcome these limitations, especially for fatigue damage prognostics problems. Therefore, in what follows, particular attention will be paid to the challenge of data scarcity. This motivated a refinement of the research questions that will be further addressed in this thesis:

1. How to facilitate the comparative evaluation of the latest ML models (notably DNNs) in fatigue damage prognosis problems?

2. Is it possible to learn meaningful representations from unlabeled data and use it to enhance related supervised predictive tasks on a fatigue damage prognostics problem ?

Chapter 3 will address the first refined research question, while the second one will be addressed in Chapter 4.

CHAPTER 3

Comparative investigations on some of the most common Machine Learning methods on a new, open-source, synthetic dataset for Fatigue Damage Prognostics

---

> "You can't just ask customers what they want and then try to give that to them. By the time you get it built, they'll want something new."
>
> Steve Jobs

Content

3.1 Motivation

As mentioned in the previous chapter, Deep Learning has become a major and rapidly growing research direction, redefining state-of-the-art performances in Prognostics in recent years. However, their effectiveness depends on the quantity and quality of available labeled data, which is generally difficult to acquire and often can be a time-consuming and expensive investment. Hence, data scarcity is actually one of the most important challenges in PHM, rendering it difficult to evaluate and compare the latest DL models in the research field.

In order to address this limitation and provide an answer to the first refined question, the current research proposes a framework and associated code to generate arbitrarily large data sets for a realistic fatigue damage prognostics problem, representative of fatigue cracks propagation in aeronautical fuselage panels. The objective of this research work is to propose a framework that will facilitate the comparative evaluation of the latest ML algorithms for fatigue damage prognostics applications, particularly in the aerospace field. An illustrative case study will also be carried out, involving several of the most commonly used DL models to address failure prognostics (*e.g.* recurrent and convolutional neural networks).

3.2 A Framework for Generating Large Data Sets for Fatigue Damage Prognostic Problems (Article 1)

The content in this section corresponds to a published work in the 2022 IEEE International Conference on Prognostics and Health Management (ICPHM) held in Detroit (Michigan), United States. *Anass Akrim, Christian Gogu, Thomas Guillebot de Nerville, Paul Strahle, Brondon Waffa Pagou, Rob Vingerhoeds and Michel Salaün*. "A Framework for Generating Large Data Sets for Fatigue Damage Prognostic Problems." In: 2022 IEEE International Conference on Prognostics and Health Management (ICPHM), Detroit (Michigan), United States [229]. DOI: 10.1109/ICPHM53196.2022.9815692. This article is referred to as Article 1 in the current manuscript.

2022 IEEE International Conference on Prognostics and Health Management (ICPHM)

A Framework for Generating Large Data Sets for Fatigue Damage Prognostic Problems

Anass Akrim[†‡], Christian Gogu[‡], Thomas Guillebot de Nerville[†], Paul Strähle[†],
Brondon Waffa Pagou[‡], Michel Salaün[†], and Rob Vingerhoeds[†]
[†]ISAE-SUPAERO, Université de Toulouse,
10 Avenue Edouard Belin, 31400 Toulouse, France
Email: {anass.akrim, michel.salaun, rob.vingerhoeds}@isae-supaero.fr
[‡]Institut Clément Ader (UMR CNRS 5312) INSA/UPS/ISAE/Mines Albi,
Université de Toulouse, 3 rue Caroline Aigle, 31400 Toulouse, France
Email: {anass.akrim, christian.gogu}@gmail.com

***Abstract*—Prognostics and Health Management (PHM) relies on the availability of large amounts of data for a given system and allows to analyse this data and to draw conclusions as to the health state of the system, the identification of faults and failures, as well as the calculation of the remaining useful life time. Often, it is expected that this data is labelled, *i.e.* that the data has been pre-analysed and that for each data point an exact information is available as to what it is about, when it was measured, etc. In reality, this is not always easy and this labeled data is not always available. For example, on aerospace structures, complete labeled data until the end of their lifetime are not usually available. This may hamper for example the use of Deep Learning (DL) techniques for Predictive Maintenance, as they rely on the availability of large amounts of labeled sensor data. In this paper a framework and associated code[1] is proposed to generate high dimensional data sets for a realistic fatigue damage prognostics problem, representative of fatigue cracks propagation in aeronautical fuselage panels. With this data, DL techniques can be trained, and we will illustrate this with a case study involving several of the most commonly used DL models to address failure prognostics.**

***Index Terms*—Prognostics and Health Management (PHM), Remaining Useful Life (RUL), Fatigue Damage Prognostics Problem (FDPP), Synthetic Dataset, Deep Learning.**

I. Introduction

Prognostics and Health Management (PHM) is a field of research and application, making use of past and present available information of an equipment in order to detect its degradation, diagnose its faults, predict and manage its failures [1]. Fatigue damage is defined as one of the major life-limiting factors for many structural components subjected to variable loadings in service (e.g. aircrafts during flight) [2]. Fatigue is the cause of various failure modes in aerospace, develops gradually and progressively grows to a critical size a_{crit}, leading to structural or system failure. The operating time before failure is commonly referred to as Remaining Useful Life (RUL) [3]. Fatigue monitoring and potential early identification of critical cracks approaching the critical size a_{crit} is a major challenge, with great potential in terms of improving the operational efficiency through the development of predictive maintenance strategies. Hence, prediction of fatigue life in structures is necessary, becoming one of the main issues in the field of operational safety.

Among Prognostics approaches, Data-Driven models have gained more and more attention in the PHM community, especially the latest Machine Learning (ML) techniques (notably Deep Learning (DL) techniques) [4]–[6]. DL has become a major and rapidly growing research direction, redefining state-of-the-art performances in a wide range of areas in recent years [7]. As more data becomes available in the engineering domain, there is a recent surge of interest in using Deep Learning in Prognostics and Health Management [8]. However, their effectiveness depends on the quantity and quality of available labeled data, which is generally difficult to acquire and often can be a time-consuming and expensive investment. Indeed in PHM, faults are rare and structures can be replaced before reaching failure; in Prognostics tasks, a label can constitute the RUL at each time step of measurements. Hence, data scarcity is becoming one of the most important challenges in PHM [9], [10], rendering it difficult to evaluate and compare the latest DL models in the research field.

Several PHM researchers have already attempted to address this challenge, proposing realistic synthetic data sets that have been collected and made publicly available by NASA's Prognostic Center of Excellence, such that for turbofan engines [3], bearings [11], batteries [12], etc. The reader may refer to [13] for more details on commonly used synthetic data sets in PHM. Although these open source data sets have been widely successful among the PHM community, to the best of the authors' knowledge, little to no research has directly aimed at proposing and making available an open source framework for generating large data sets specific to fatigue damage prognostic problems. Virkler et al. [14] proposed a fatigue crack growth dataset for fatigue damage prognostic problems, allowing several PHM researchers to evaluate data-driven approaches on it [15], [16]. However, the proposed datasets consist only of time series of structural crack length data and, according to [13], indirect sensory

[1]See https://github.com/ansak95/FrameworkFDPP

978-1-6654-6615-8/22/$31.00 ©2022 IEEE

measurements (e.g. vibration, acoustic emissions, strains, etc.) are not provided, making it difficult to transfer to real life case applications where there is no direct access to crack length data but increasing amounts of indirect sensor data may instead be available.

In order to stimulate research on the applicability of the latest deep learning models to fatigue damage prognostics, notably in the aerospace domain, while awaiting real in service data, our paper proposes a framework and software code for synthetically generating large training data sets for a realistic fatigue damage prognostics problem (FDPP), simulating crack propagation in a fuselage panel, where the indirect sensory measurements correspond to mechanical strains. The crack growth is simulated based on Paris-Erdogan's crack growth model [17] and we consider strain data at various position (i.e. synthetic strain gauges) as output that will be used as sensor data. Due to the synthetic nature of this data set, it is possible to significantly vary the size of the training data sets, which may be particularly relevant for some deep learning approaches. The authors believe that the proposed framework and software code can help facilitate the development of deep learning algorithms for prognostic applications, and make them more easily transferable to real world applications. As illustration, some of the most common deep learning models have been applied to the generated data sets, including Recurrent Neural Network (RNN), Long short-term memory (LSTM), Gated Recurrent Unit (GRU), 1D-Convolutional Neural Network (CNN), and Temporal Convolutional Network (TCN), and we notably investigated the variation of the methods' performance with increasing labeled training data. All codes, both for the generation of the synthetic datasets and for the training of all the models are publicly available on https://github.com/ansak95/FrameworkFDPP.

The remainder of the paper is organized as follows. Section II describes the chosen approach to generate a synthetic data set simulating the crack growth in the considered structures. The degradation model used to generate the data set is presented in this section as well. Section III illustrates a realistic case study in order to investigate the applicability of DL methods in a prognostics problem based on a generated synthetic data set. Section III-A describes the data set generated. Section III-B presents the deep learning-based models investigated to illustrate this study and details the proposed RUL estimation strategy. The proposed metric for performance evaluation, used to rank the models investigated, are presented in this section as well. Section III-C presents and analyzes the results of the investigated deep learning-based approaches in this study. Finally, Section IV concludes the paper and identifies some research perspectives.

II. Framework

The proposed framework seeks to generate synthetic data sets of mechanical strain data (i.e. virtual strain gauges), simulating the crack growths in structures based on the Paris-Erdogan model. These synthetic data sets will be composed of labeled data, i.e. measurement sequences of structures until failure, where the label is the RUL of the structure at each time step of the strain measurements.

A. Simulation of the crack growth : a theoretical approach

1) Crack growth model: Fatigue crack propagation is modeled by the Paris-Erdogan's law [18]

$$\frac{da}{dk} = C(\Delta\sigma\sqrt{\pi a})^m \tag{1}$$

where a is the half crack size, k is the number of loading cycles, C and m are the empirical parameters of the Paris-Erdogan's law, and $\Delta\sigma$ is the difference between the minimum σ_{min} and maximum σ_{max} far field stress.

This law will be used to simulate the crack growth in the current setting to give the crack length at a given number of cycles, which will in turn be used to generate the deformation data on which the data-driven models will be trained. To get the crack length after k cycles, the analytical solution to Paris-Erdogan's law from [19] provided in Eq. 2 will be used.

$$a_k = \left[kC\left(1-\frac{m}{2}\right)\left(\Delta\sigma\sqrt{\pi}\right)^m + a_0^{1-\frac{m}{2}}\right]^{\frac{2}{2-m}} \tag{2}$$

where a_0 is the initial crack length.

The critical crack size a_{crit} that causes structural failure can be calculated by the empirical formula in Eq. (3).

$$a_{crit} = \left(\frac{K_I}{\Delta\sigma\sqrt{\pi}}\right)^2 \tag{3}$$

where K_I is a conservative estimate of the fracture toughness in Mode I crack loading [20].

2) Strain fields around the fatigue crack: In order to obtain the strain field around the crack, we work under the assumption of a finite crack in an infinite plate under Mode I crack loading [20], assuming $\Delta\sigma = \sigma_{max} - \sigma_{min}$ with $\sigma_{min} = 0\ MPa$. In practice, the infinite plate assumption implies that the crack size must be much smaller than the size of the plate, which is typically verified for structures such as fuselage panels. A figure of a crack of length $2a$ and a loading of σ_{max} is given in Figure 1.

To calculate the displacements around the crack, a complex variable formulation along with a Westergaard approach is used [18], [20], [21]. In a Cartesian coordinate system (x, y), a complex variable z is introduced :

$$z = x + iy \tag{4}$$

The Airy stress function ϕ used in this approach is defined such that :

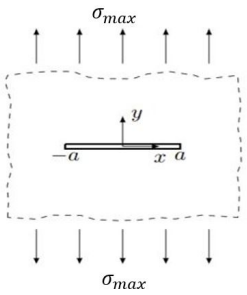


Fig. 1. Crack of length $2a$ in a plate under Mode I loading

$$\phi = \sigma_{max}\sqrt{z^2-a^2} - \sigma_{max}z + \text{const.} \quad (5)$$
$$\phi' = \frac{\sigma_{max}z}{\sqrt{z^2-a^2}} - \sigma_{max} \quad (6)$$
$$\phi'' = \frac{\sigma_{max}}{\sqrt{z^2-a^2}} - \frac{\sigma_{max}z^2}{(z^2-a^2)^{\frac{3}{2}}} \quad (7)$$

The stresses in a plane stress state, which is assumed here, can then be expressed as

$$\sigma_{11} = \Re(\phi') - y\Im(\phi'') \quad (8)$$
$$\sigma_{22} = \Re(\phi') + y\Im(\phi'') + \sigma_{max} \quad (9)$$
$$\sigma_{12} = -y\Re(\phi'') \quad (10)$$

where $\Re(.)$ and $\Im(.)$ are respectively the real part and the imaginary part of a complex number.

The corresponding strain state for this stress state are given by Hooke's law [22], which can be written under the plane stress assumption as :

$$\varepsilon_{11} = \frac{1}{E}(\sigma_{11} - \nu\sigma_{22}) \quad (11)$$
$$\varepsilon_{22} = \frac{1}{E}(-\nu\sigma_{11} + \sigma_{22}) \quad (12)$$
$$\varepsilon_{12} = \frac{1+\nu}{E}\sigma_{12} \quad (13)$$
$$\varepsilon_{33} = \frac{-\nu}{E}(\sigma_{11} + \sigma_{22}) \quad (14)$$

where E and ν are the elastic parameters, respectively Young's modulus and Poisson's ratio. A typical normal strains state is shown in Fig. 2.

Finally, the strain measurements ε of a strain gauge placed at the position (x, y) with an angle θ between the x-axis and the strain gauge measurements direction are given by Eq. 15.

$$\varepsilon = \varepsilon_{11}\cos(\theta)^2 + \varepsilon_{22}\sin(\theta)^2 + 2\varepsilon_{12}\cos(\theta)\sin(\theta) \quad (15)$$

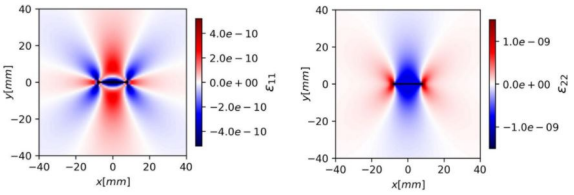


Fig. 2. Example normal strains state for a crack of length $2a$

B. Generation Algorithm

All data sets composed of strain measurements are generated using the previously defined crack growth model and, for initialization, each structure n is defined by three parameters that vary from structure to structure within the data set : the initial crack length $a_{0,n}$, and the two material parameters C_n and m_n generating the crack growth. Note that in this paper, we consider that C is distributed following a log-normal law, and $\log C$ and m are assumed to follow a multivariate distribution with a linear correlation coefficient ρ, based on the literature [23], [24]. The steps of the numerical implementation of the data set creation are summarized below, illustrated in Fig. 3:

1) For the n^{th} structure $(n = 1, ..., N)$, draw the samples of initial crack size and the Paris-Erdogan's law parameters respectively : $a_{0,n} \sim \mathcal{N}(\mu_{a_0}, \sigma_{a_0})$ and $(m_n, \log C_n) \sim \mathcal{N}(\mu_m, \sigma_m, \mu_{\log C}, \sigma_{\log C}, \rho)$.
2) Using the crack growth model described in Section II-A1, propagate the crack size of the n^{th} structure until it reaches the critical crack size a_{crit}. Every Δk cycles until failure, compute and collect the strain measurements at the n_g strain gauge positions according to Eq. 15.

In order to train prognostics models based on this strains dataset we also need the RUL at each cycle. For the n^{th} structure at cycle k, the remaining useful life $RUL_{n,k}$ can be computed such that $RUL_{n,k} = k_{crit} - k$ (where k_{crit} is the number of cycles for the crack to reach the critical size, such that $a_{k_{crit}} = a_{crit}$). An illustration of three generated sequences (i.e. three strain gauges) for a single structure until failure is given in Fig. 4.

III. Illustration

In this section, an illustration of the framework, the generated data sets and the use of Deep Learning (DL) techniques for the estimation of the Remaining Useful Lifetime (RUL) is provided.

A. Data Set

Starting from the analytical setup described in the previous section, a synthetic data set is generated containing the variations of the strains at $n_g = 3$ positions in the plate as a function of the number of cycles. This setup can

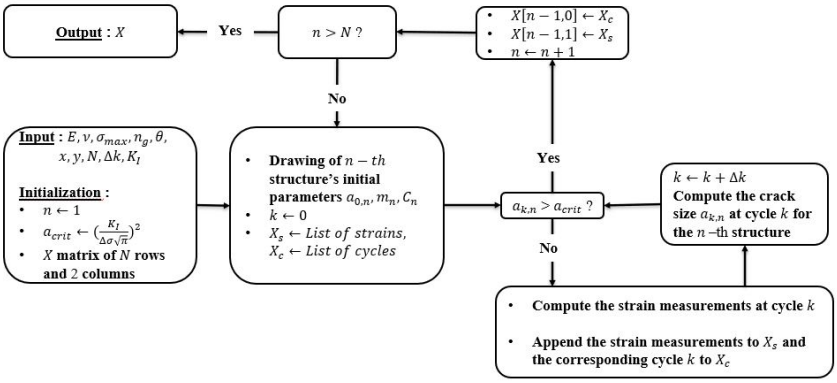


Fig. 3. Flow chart of the strain gauge sequence generation.

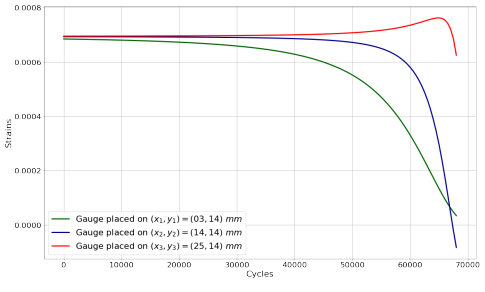


Fig. 4. Strain values time series corresponding to a sensor sample generated. In this illustration, three strain gauges are placed at the following positions (x,y) : (3,14), (14,14) and (25,14) mm.

be seen representative of real experiments under fatigue loading where the strain state is monitored at three strain gauge positions. The strain data, or measurement sequences, are obtained until the critical crack size a_{crit} is reached. In this experiment, an Aluminum alloy 7075-T6 plate was considered, which is typical of aeronautic structures. The elastic parameters considered are Young's modulus $E = 71.7GPa$ and Poisson's ratio $\nu = 0.33$ (assumed to be constant at their nominal values). The critical crack size a_{crit} that causes structural failure can be calculated by the empirical formula in Eq. (3), in which $K_I = 19,7\ MPa\sqrt{m}$ and $\Delta\sigma = \sigma_{max} - \sigma_{min} = 78,6\ MPa$ with $\sigma_{min} = 0\ MPa$ in this work.

In this series of numerical experiments, it is assumed that the initial crack position is at the origin of the (x, y) reference frame and that the crack is along the x direction. Furthermore, it is assumed that the strain gauges are placed at an angle θ = 45° with x-axis, so they are sensitive to both ϵ_{11} and ϵ_{22} strains. The $n_g = 3$ strain gauges are placed at the following positions (x,y) : (3,14), (14,14) and (25,14) mm. Considering that the evolution of the changes from one cycle to another are small, it was decided to collect the data every $\Delta k = 500$ cycles. Fig. 4 illustrates an instance of sequences generated for a single structure until failure.

In the following illustration, artificial experiments are set up in order to acquire a training, validation and testing data set. The validation set is used to monitor and optimize the models hyperparameters during the training phase; the testing set is used to evaluate the performance of the trained models as a held-out data set that has not been used prior, either for training the model or tuning the model hyperparameters. For training, data sets of various sizes $N_{T_1} = 100$, $N_{T_2} = 500$, $N_{T_3} = 1000$ structures are generated, while for validation a set of $N_V = 100$ structures is generated, and for testing, a set of $N_{Test} = 100$ structures is generated. Each structure n is defined by three parameters that vary from structure to structure within the data set (i.e. the initial crack length $a_{0,n}$, C_n and the exponent m_n), such that :

1) a_0 varies following a Gaussian law with a mean of $1 \cdot 10^{-3}[m]$ and a coefficient of variation (C_v) of 0.125, such that $C_v(a_0) = \frac{\sigma_{a_0}}{\mu_{a_0}}$, where the Gaussian is truncated to avoid negative values.
2) C is distributed following a log-normal law with a geometric mean equal to $1 \cdot 10^{-10}$ and a geometric standard deviation such that the exponential of the upper and lower bounds of the 95% confidence interval for $\log C$ have a ratio of $8 \cdot 10^3$, which is consistent with the data from [25].
3) Finally, m is distributed following a Gaussian law such that the bounds of the 95% confidence interval are 2 and 4, as suggests [18] and is again roughly consistent with [25]. Again, the distribution is truncated at the lower end at 0 so that m cannot become negative.

Parameter	Denotation	Type	Value	Unit
Elastic parameters				
Young's modulus	E	Deterministic	71.7	GPa
Poisson's ratio	ν	Deterministic	0.33	-
Strain field parameters				
Maximum stress intensity	σ_{max}	Deterministic	$78, 6.10^6$	Pa
Fracture toughness	K_I	Deterministic	$19, 7.10^6$	$Pa\sqrt{m}$
Strain gauges				
Number of gauges placed	n_g	Deterministic	3	-
Position of the gauges placed	$(x_i, y_i)_{i=1,..,n_g}$	Deterministic	$(3, 14), (14, 14), (25, 14)$	mm
Angle of the gauges placed	θ	Deterministic	45	deg
Initialization parameters				
Initial crack size	a_0	Gaussian distribution	$\mathcal{N}(\mu_{a_0}, \sigma_{a_0})$	m
Mean of a_0	μ_{a_0}	Deterministic	1.10^{-3}	m
Standard deviation of a_0	σ_{a_0}	Deterministic	$0, 125.10^{-3}$	m
Paris-Erdogan's law parameters	$(m, \log C)$	Multivariate Gaussian distribution	$\mathcal{N}(\mu_m, \sigma_m, \mu_{\log C}, \sigma_{\log C}, \rho)$	-
Mean of m	μ_m	Deterministic	$3, 5$	-
Standard deviation of m	σ_m	Deterministic	$0, 125$	-
Mean of C	μ_C	Deterministic	1.10^{-10}	-
Standard deviation of C	σ_C	Deterministic	5.10^{-11}	-
Correlation coefficient of m and $\log C$	ρ	Deterministic	-0.996	-
Generated data set				
Number of training structures	$(N_{T_1}, N_{T_2}, N_{T_3})$	Deterministic	$(100, 500, 1000)$	-
Number of validation structures	N_V	Deterministic	100	-
Number of testing structures	N_{test}	Deterministic	100	-
Data collection interval	Δk	Deterministic	500	-

TABLE I
PARAMETERS FOR NUMERICAL STUDY.

For the material studied in this paper (i.e. alluminum alloys), m and $\log C$ are assumed to follow a multivariate distribution with a negative correlation coefficient of $\rho = -0.996$ [26]. The parameters of this numerical case study are summarized in Table I.

B. Methodology

In this section, an overview of some Deep Learning methods commonly used in prognostics is provided, followed by a description of the problem considered in this paper and the way these methods were implemented on it.

1) Deep Learning in Prognostics: Deep Learning has shown multiple times to provide good results when applied to RUL prediction [9], [27]. Among the deep learning techniques, we can enumerate two widely used algorithms in the prognostics field :

- Recurrent Neural Networks
- Convolutional Neural Networks

Both will be briefly introduced below.

a) Recurrent Neural Networks: The most common type of Deep Learning model for Time Series Forecasting are Recurrent Neural Networks (RNN) [28]. The algorithm remembers its input due to an internal memory, which makes it well suited for problems that involve sequentially evolving data. Good results have been obtained by applying RNNs to a variety of problems in non-PHM fields, such as speech recognition or language modeling [29]–[31]. Due to their ability to capture time-dependent relationships, RNNs have achieved interest among PHM researchers as well, especially given the sequential nature of the sensor data in the prognostics field (e.g. sensors data).

However, standard RNNs are limited to looking back only a few steps due to the vanishing gradient or exploding gradient problem (see for example [32], [33]). To address this issue, Long Short-Term Memory (LSTM) networks were proposed, and have established themselves as one of the most used Deep Learning model types in many fields and especially in Natural Language Processing (NLP) [34]. More recently, LSTM networks have also grown in popularity and have been used by several researchers in the PHM community [9]. One of the major drawbacks of LSTM concerns their relatively high computational cost and memory requirement for training [35]. A slightly simplified variation of the LSTM is the Gated Recurrent Unit (GRU), introduced by Cho et al. [36], gaining in popularity in recent years due to its relative simplicity [37]. In [38], results showed GRU model could achieve competitive results with a better training performance than LSTM for RUL estimation.

b) Convolutional Neural Networks: Convolutional Neural Networks (CNN) concern a specific type of deep neural network inspired by the organization of neurons in the visual cortex. Presented by LeCun et al. in 1998 [39], CNNs achieved significant success in many research and industry fields including computer vision, natural language processing and speech recognition [40], [41]. Input data for CNNs are usually 2-dimensional (e.g. height and width pixels for images) so to learn abstract spatial features. Li et al. [42] investigated how a 2D-DeepCNN model can be used in prognostics and remaining useful life prediction. 1D-CNNs have also been introduced to the analysis of time

sequences in RUL estimation. The key difference between 1D-CNN, 2D-CNN and 3D-CNN is the dimensionality and management of the input data and how the feature detector (or filter) slides across the data. In this paper, only 1D-CNNs will be considered due to the format of our data sets (Time Series). Indeed, the application of the CNN architecture to time series prediction aims to exploit the filters' feature extraction capability demonstrated in image classification, and CNNs are easier to train than recurrent neural networks due to the implementation of convolutional rather than recursive operations, allowing improved numerical efficiency. However, note that CNNs were initially introduced as a classifiers [39], thus more suitable for classification problems than regression problems in sequence modeling (i.e. for RUL estimation problems that have been essentially considered as regression problems so far).

According to [43], sequence modeling was synonymous with recurrent networks for most deep learning practitioners until 2018. Bai et al. [43] introduced a novel CNN architecture for sequence modeling, using dilated causal convolution to preserve the causal order of the input time series : Temporal Causal Networks (TCN). Their results showed that the TCN outperformed the standard recurrent neural networks (RNN, LSTM and GRU) in most Sequence Modeling Tasks (better performances on ten tasks out of eleven). Among PHM researchers, Liu et al. [44] proposed a TCN model for RUL prediction of rolling bearings based on raw vibration data. In their paper, results confirmed that the proposed model is able to outperform generic recurrent architectures such as LSTM and GRU in sequence modeling. Moreover, the authors showed that offline training of the proposed model is almost four times faster than an LSTM network. Indeed, according to Bai et al. [43], the TCN has several advantages that make it superior to the Recurrent Neural Networks (RNN, LSTM, GRU), specifically :

- better control of the model's memory size with a flexible receptive field size (stacking more dilated causal convolutional layers, using larger dilation factors, or increasing the filter size);
- a backpropagation path different from the one used by recurrent neural networks, which allows to avoid the exploding/vanishing problem.

2) Problem statement: In this paper, the RUL prediction problem is considered as a multivariate time-series-related regression problem and, as illustration, the five most commonly used deep learning-based algorithms in Time Series Forecasting (RNN, LSTM, GRU, 1D-CNN and TCN) have been applied to the generated data set.

In the training phase, a sliding window approach is used. At each time-step t, the input of the predictive model correspond to the current and past measures, such that $X_t := (x_{t-n_w+1}, \ldots, x_t) \in \mathbb{R}^{n_g \text{ x } n_w}$ where n_g is the number of strain gauges and thus of time series and n_w is the length of the sliding window; the output of the predictive model is a point-wise estimation of the RUL of the structure at time t, denoted $\hat{RUL}_t \in \mathbb{R}$. Note that each training structure is composed of S samples (sliding windows of size n_w) with $S = n_T - n_w$, n_T the number of timesteps in the sequence.

After training, the models are evaluated on the testing set composed of N_{test} different structures, and for each structure a unique RUL estimation is performed at a time t_n^*. For each structure $n \in \{1, \ldots, N_{test}\}$, the parameter t_n^* is randomly drawn such that $t_n^* \sim k_{crit}^n \times \mathcal{U}([0,33; 0,9])$, where k_{crit}^n is the cycle of failure for the n-th structure. In plain words, this means that we carry out a test prediction for the RUL at a time t_n^* which is drawn uniformly between 33% and 90% of the sequence's length. Hence, the input data for the model is $X_{t_n^*} = (x_{t_n^*-n_w+1}, \ldots, x_{t_n^*}) \in \mathbb{R}^{n_g \text{ x } n_w}$ and the output of the model is $\hat{RUL}_{t_n^*} \in \mathbb{R}$.

3) Training and Evaluation: For the recurrent neural networks, the input data is pre-processed using a min-max scaler before being fed into the models. Fig. 5 shows an example of how the time series data of the strain gauges from a single structure are processed, and gives an illustration of recurrent neural networks architecture used in this work.

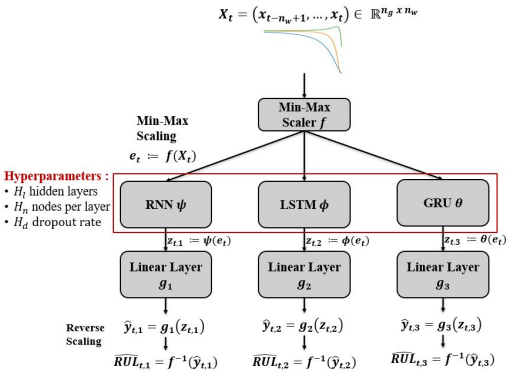


Fig. 5. Brief illustration of recurrent neural networks (RNN, LSTM and GRU) architectures. n_g corresponds the number of gauges placed (i.e. number of time series), and n_w to the window size.

The first layer of the 1D-CNN's architecture is a normalization layer. The architecture of the 1D-CNN models is illustrated in Fig. 6.

The input data is not pre-processed for the TCN model and we have kept the same architecture as presented in [45], illustrated in Fig. 7.

Given that the RUL estimation problem is considered as a regression problem, we aim to minimize a mean squared error loss L_{MSE} during the training phase, and the mean absolute percentage error (*MAPE*) metric is used to evaluate the performance of the investigated models such that :

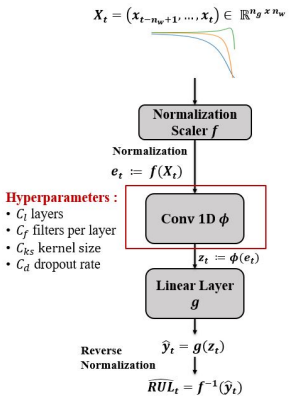


Fig. 6. Brief illustration of convolutions neural networks (1D-CNN) architecture. n_g corresponds the number of gauges placed (i.e. number of time series), and n_w to the window size.

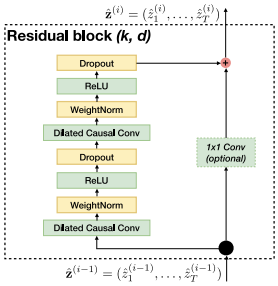


Fig. 7. Architecture of a residual block of TCNs introduced in [45]. A TCN is a stack of k residual blocks, each composed of two 1D-CNN layers (same hyperparameters as illustrated in Fig. 6) with a dilation factor d followed by a weight normalization layer used for regularization [46].

$$L_{MSE} = \frac{1}{S}\sum_{i=1}^{S}(RUL_i - \hat{RUL}_i)^2 \quad (16)$$

$$MAPE = \frac{1}{S}\sum_{i=1}^{S}\left|\frac{RUL_i - \hat{RUL}_i}{RUL_i}\right| * 100 \quad (17)$$

where S is the number of samples with $\hat{RUL}_.$ being the prediction and $RUL_.$ the target value.

In the training phase, our strategy consists of a hyperparameters optimization stage using a Grid Search algorithm [47], followed by a Fine-tuning stage. During the Fine-tuning stage, we use the Adam optimizer with default parameters and we decrease the learning rate incrementally : we sequentially try the learning rates $10^{-3}, 10^{-4}, 10^{-5}$ for a predefined number of epochs, saving the model weights each time the validation MAPE decreases; the weights of the best model are loaded each time the learning rate is lowered. By using this approach, the model in the early stages of training with a high learning rate is less likely to get stuck in a local minimum and explores a wider range of possible configurations. As the training comes closer to an optimum, the decaying learning rate helps with convergence and avoiding oscillations around local minima [48].

C. Experiments and Results

Due to the time series nature of the data and the sliding window procedure (with $n_w = 30$ after some preliminary experiments), the training sets with $N_{T_1} = 100$, $N_{T_2} = 500$ and $N_{T_3} = 1000$ structures correspond to respectively 10956, 54365 and 108452 training samples. The validation set is processed in the same way : N_V contains 100 validation structures thus 10937 samples. For testing, a set of $N_{Test} = 100$ structures is used, hence composed of 100 samples since only one RUL estimation is performed per structure.

For the recurrent neural networks, the optimization strategy was applied on 100, 500 and 1000 structures. The optimal hyperparameters found are summarized in Table II.

Training structures	100			500			1000		
Model	RNN	LSTM	GRU	RNN	LSTM	GRU	RNN	LSTM	GRU
Hidden layers	3	2	3	3	2	2	3	2	2
Nodes per layer	32	64	256	32	128	256	32	128	256
Dropout rate	0	0	0	0	0	0	0	0	0

TABLE II
BEST MODELS OF THE RNN, LSTM AND GRU HYPERPARAMETER OPTIMIZATION FOR 100, 500 AND 1000 TRAINING STRUCTURES

For the 1D-CNN and TCN model the hyperparameter optimization were again performed on 100, 500 and 1000 training structures, and the optimal hyperparameters found are summarized in Table III. Note that the dilation factor is set to $d = 6$, so that the receptive field is of size $2^{d-1} = 2^5 = 32$, thus able to capture relationships over a time sequence of size $n_w = 30$ in this work, as described in [45].

Training structures	100	500	1000
Convolutional layers	6	2	4
Filters per layer	20	40	25
Kernel size	6	12	9
Dropout rate	0.0	0.0	0.0

(a) Best models of the 1D-CNN.

Training structures	100	500	1000
TCN residual blocks	8	8	8
Filters per layer	30	25	35
Kernel size	2	2	2
Dropout rate	0.0	0.0	0.0

(b) Best models of the TCN.

TABLE III
HYPERPARAMETER OPTIMIZATION FOR 100, 500 AND 1000 TRAINING STRUCTURES.

As the hyperparameters are optimized, the resulting models are fine-tuned on the same data set with an adaptative learning rate to optimize their performance, and then evaluated on 100 structures of the testing data set; for each structure we only have one sample to evaluate, thus the models are evaluated on 100 samples. Table IV shows an overview of the achieved accuracy for all recurrent neural networks (RNN, LSTM and GRU) and convolutional neural networks (1D-CNN and TCN).

Training structures	100		500		1000	
$MAPE(\%)$	Val	Test	Val	Test	Val	Test
RNN	3.5	2.95	2.22	1.54	0.91	0.67
LSTM	**0.55**	**0.65**	0.25	1.15	1.50	1.24
GRU	1.72	1.22	**0.05**	**0.02**	**0.08**	**0.04**
1D-CNN	4.19	3.13	1.06	0.82	0.58	0.54
TCN	2.57	2.35	0.76	0.66	0.77	0.69

TABLE IV
REGRESSION TASK : MAPE (%) OF ALL FINE TUNED RNN (RNN, LSTM AND GRU) AND CNN MODELS (1D-CNN AND TCN) ON THE VALIDATION AND TESTING DATA SETS

LSTMs outperformed the other algorithms when trained on 100 structures (0.65% testing MAPE score), while the best performance in general was achieved by GRU models (0.02% testing MAPE score when trained on 500 structures, and 0.04% testing MAPE score when trained on 1000 structures). Indeed, recurrent neural networks (especially GRU) appear to be the best suited for the regression task in this RUL estimation problem. Nevertheless, we can see that more available training data leads to a better performance of the models and a better identification of the most suitable models for the problem.

IV. CONCLUSION

In this paper, a framework and code for synthetically generating high dimensional data sets for a realistic fatigue damage prognostics problem has been presented. The proposed framework generates multivariate run-to-failure time series data for structures subject to fatigue loading, consisting of synthetic mechanical strain data sets (i.e. synthetic strain gauges) and associated RUL based on the Paris-Erdogan crack growth model. The authors believe that the proposed framework will help facilitate the benchmarking of latest ML algorithms for fatigue damage prognostics applications, in particular in the aerospace domain (e.g. fuselage panels).

As illustration, pre-cracked Aluminum alloy 7075-T6 plates were considered, which are typical of aeronautic structures, and the applicability of some of the most commonly used DL models to address failure prognostics (including RNN, LSTM, GRU, 1D-CNN, and TCN) have been studied. Without the "flaws" of real world data, good results were expected and obtained. Indeed, there is no noise in the data which would not be the case in reality : a gaussian noise can be added to time series in order to mimic the effect of aleatoric uncertainties that occur in real world applications. In next steps, building upon the possibility to "play" with the initialized parameters, this illustration case study can be further varied in order to complexify the data set and make it as realistic as possible.

Furthermore, the authors believe that the use of synthetic data can improve the prognostic performance of data-driven models in real-world cases in the absence of training data or with very limited labeled data, using knowledge transfer when a model is pre-trained on a large set of related synthetic data (e.g. Transfer Learning [49], [50]), and the presented framework might be useful in this aspect as well.

ACKNOWLEDGMENT

This work was partially funded by the French "Occitanie Region" under the Predict project. This funding is gratefully acknowledged. This work has been carried out on the supercomputers PANDO (ISAE-SUPAERO, Toulouse) and Olympe (CALMIP, Toulouse, project n°21042). Authors are grateful to ISAE-SUPAERO and CALMIP for the hours allocated to this project. The authors would like to thank Christian Thomas Nitschke for his initial input on modelling the crack growth problem.

REFERENCES

[1] E. Zio, "Prognostics and health management of industrial equipment," in *Diagnostics and prognostics of engineering systems: methods and techniques*. IGI Global, 2013, pp. 333–356.
[2] A. Vasudevan, K. Sadananda, and N. Iyyer, "Fatigue damage analysis: Issues and challenges," *International Journal of Fatigue*, vol. 82, pp. 120–133, 2016.
[3] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *2008 international conference on prognostics and health management*. IEEE, 2008, pp. 1–9.
[4] K. Javed, "A robust & reliable data-driven prognostics approach based on extreme learning machine and fuzzy clustering." Ph.D. dissertation, 2014.
[5] K. L. Tsui, N. Chen, Q. Zhou, Y. Hai, and W. Wang, "Prognostics and health management: A review on data driven approaches," *Mathematical Problems in Engineering*, vol. 2015, 2015.
[6] X.-S. Si, W. Wang, C.-H. Hu, and D.-H. Zhou, "Remaining useful life estimation–a review on the statistical data driven approaches," *European journal of operational research*, vol. 213, no. 1, pp. 1–14, 2011.
[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," vol. 521, no. 7553, pp. 436–444.
[8] A. Voulodimos, N. Doulamis, G. Bebis, and T. Stathaki, "Recent developments in deep learning for engineering applications," vol. 2018, p. 8141259.
[9] O. Fink, Q. Wang, M. Svensén, P. Dersin, W.-J. Lee, and M. Ducoffe, "Potential, challenges and future directions for deep learning in prognostics and health management applications," *Engineering Applications of Artificial Intelligence*, vol. 92, p. 103678, 2020.
[10] A. Theissler, J. Pérez-Velázquez, M. Kettelgerdes, and G. Elger, "Predictive maintenance enabled by machine learning: Use cases and challenges in the automotive industry," *Reliability engineering & system safety*, vol. 215, p. 107864, 2021.
[11] P. Nectoux, R. Gouriveau, K. Medjaher, E. Ramasso, B. Chebel-Morello, N. Zerhouni, and C. Varnier, "Pronostia: An experimental platform for bearings accelerated degradation tests." in *IEEE International Conference on Prognostics and Health Management, PHM'12*. IEEE Catalog Number: CPF12PHM-CDR, 2012, pp. 1–8.
[12] B. Saha and K. Goebel, "Battery data set," *NASA AMES prognostics data repository*, 2007.

[13] O. F. Eker, F. Camci, and I. K. Jennions, "Major challenges in prognostics: Study on benchmarking prognostics datasets," in *PHM Society European Conference*, vol. 1, no. 1, 2012.

[14] D. A. Virkler, B. Hillberry, and P. Goel, "The statistical nature of fatigue crack propagation," 1979.

[15] A. Ray and S. Tangirala, "Stochastic modeling of fatigue crack dynamics for on-line failure prognostics," *IEEE Transactions on Control Systems Technology*, vol. 4, no. 4, pp. 443–451, 1996.

[16] A. Abbasi, F. Nazari, and C. Nataraj, "Application of long short-term memory neural network to crack propagation prognostics," in *2020 IEEE International Conference on Prognostics and Health Management (ICPHM)*. IEEE, 2020, pp. 1–6.

[17] P. Paris and F. Erdogan, "A critical analysis of crack propagation laws," vol. 85, no. 4, pp. 528–533.

[18] A. T. Zehnder, *Fracture mechanics*, ser. Lecture notes in applied and computational mechanics. London ; New York: Springer Science+Business Media, 2012, no. 62, oCLC: ocn755698387.

[19] Y. Wang, C. Gogu, N. H. Kim, R. T. Haftka, N. Binaud, and C. Bes, "Noise-dependent ranking of prognostics algorithms based on discrepancy without true damage information," *Reliability Engineering & System Safety*, Oct. 2017.

[20] C.-T. Sun and Z. Jin, "The elastic stress field around a crack tip," *Fracture Mechanics*, pp. 25–75, 2012.

[21] A. T. Zehnder and M. J. Viz, "Fracture mechanics of thin plates and shells under combined membrane, bending, and twisting loads," *Appl. Mech. Rev.*, vol. 58, no. 1, pp. 37–48, 2005.

[22] A. F. Bower, *Applied mechanics of solids*. Boca Raton: CRC Press, 2010, oCLC: ocn277196164. [Online]. Available: solidmechanics.org

[23] J. Benson and D. Edmonds, "Relationship between the parameters c and m of paris' law for fatigue crack growth in a low-alloy steel," *Scr. Metall.;(United States)*, vol. 12, no. 7, 1978.

[24] M. Cortie and G. Garrett, "On the correlation between the c and m in the paris equation for fatigue crack propagation," *Engineering fracture mechanics*, vol. 30, no. 1, pp. 49–58, 1988.

[25] G. Sinclair and R. Pieri, "On obtaining fatigue crack growth parameters from the literature," *International Journal of Fatigue*, vol. 12, no. 1, pp. 57–62, Jan. 1990. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/014211239090343D

[26] E. H. Niccolls, "A correlation for fatigue crack growth rate," *Scripta Metallurgica*, vol. 10, no. 4, pp. 295–298, Apr. 1976. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/003697487690079X

[27] J. J. Montero Jimenez, S. Schwartz, R. Vingerhoeds, B. Grabot, and M. Salaün, "Towards multi-model approaches to predictive maintenance: A systematic literature survey on diagnostics and prognostics," vol. 56, pp. 539–557. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0278612520301187

[28] H. Hewamalage, C. Bergmeir, and K. Bandara, "Recurrent neural networks for time series forecasting: Current status and future directions," *International Journal of Forecasting*, vol. 37, no. 1, pp. 388–427, 2021.

[29] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," 2014.

[30] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

[31] K. Yao, G. Zweig, M.-Y. Hwang, Y. Shi, and D. Yu, "Recurrent neural networks for language understanding," in *INTERSPEECH*, 2013.

[32] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[33] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, 2013, pp. 1310–1318.

[34] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Łukasz Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean, "Google's neural machine translation system: Bridging the gap between human and machine translation."

[35] T. Masuko, "Computational cost reduction of long short-term memory based on simultaneous compression of input and hidden state," in *2017 IEEE automatic speech recognition and understanding workshop (ASRU)*. IEEE, 2017, pp. 126–133.

[36] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[37] R. Rana, "Gated recurrent unit (gru) for emotion classification from noisy speech," *arXiv preprint arXiv:1612.07778*, 2016.

[38] M. Baptista, H. Prendinger, and E. Henriques, "Prognostics in aeronautics with deep recurrent neural networks," in *PHM Society European Conference*, vol. 5, no. 1, 2020, pp. 11–11.

[39] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[40] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *arXiv preprint arXiv:1901.06032*, 2019.

[41] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1d convolutional neural networks and applications: A survey," *arXiv preprint arXiv:1905.03554*, 2019.

[42] X. Li, Q. Ding, and J.-Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Reliability Engineering & System Safety*, vol. 172, pp. 1–11, 2018.

[43] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[44] C. Liu, L. Zhang, and C. Wu, "Direct remaining useful life prediction for rolling bearing using temporal convolutional networks," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2019, pp. 2965–2971.

[45] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," vol. abs/1803.01271.

[46] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," *Advances in neural information processing systems*, vol. 29, 2016.

[47] M. Feurer and F. Hutter, "Hyperparameter optimization," in *Automated Machine Learning*. Springer, Cham, pp. 3–33.

[48] K. You, M. Long, J. Wang, and M. I. Jordan, "How does learning rate decay help modern neural networks?"

[49] J. C. Paetzold, O. Schoppe, R. Al-Maskari, G. Tetteh, V. Efremov, M. I. Todorov, R. Cai, H. Mai, Z. Rong, A. Ertuerk *et al.*, "Transfer learning from synthetic data reduces need for labels to segment brain vasculature and neural pathways in 3d," in *International Conference on Medical Imaging with Deep Learning–Extended Abstract Track*, 2019.

[50] C. Douarre, R. Schielein, C. Frindel, S. Gerth, and D. Rousseau, "Transfer learning from synthetic data applied to soil–root segmentation in x-ray tomography images," *Journal of Imaging*, vol. 4, no. 5, p. 65, 2018.

3.3 Generation algorithm

As extention to the article, the steps of the numerical implementation of the dataset creation are detailed in the following algorithm:

Algorithm 1: Implementation of function to generate sequences.

Function *generate_sequences*$(E, \nu, \sigma_{max}, n_g, \theta, x, y, N, \Delta k)$
  Input : E, ν elastic parameters
  Input : σ_{max} maximum far field stress
  Input : n_g number of gauges
  Input : θ angle of gauges placement
  Input : (x, y) position of the gauges
  Input : N number of structures
  Input : Δk data acquisition interval
  $X \leftarrow$ matrix of N rows and 2 columns `// Input data of the training set`
  for $n \leftarrow 1$ **to** N **do**
    $X_s \leftarrow$ List
    $X_c \leftarrow$ List
    Drawing of $a_{0,n}$, m_n and C_n
    $z_j \leftarrow x_j + iy_j$ **for** $j \in \{1, ..., n_g\}$
    $k \leftarrow 0$
    while $a_{k,n} \leq a_{crit}$ **do**
      for $j \leftarrow 1$ **to** n_g **do**
        `// Airy stress function by Westergaard approach at cycle` k
        $\phi_{n,k,j} \leftarrow \sigma_{max}\sqrt{z_j^2 - a_{k,n}^2} - \sigma_{max} z_j$
        $\phi'_{n,k,j} \leftarrow \frac{\sigma_{max} z_j}{\sqrt{z_j^2 - a_{k,n}^2}} - \sigma_{max}$
        $\phi''_{n,k,j} \leftarrow \frac{\sigma_{max}}{\sqrt{z_j^2 - a_{k,n}^2}} - \frac{\sigma_{max} z_j^2}{(z_j^2 - a_{k,n}^2)^{\frac{3}{2}}}$
        `// Stress state at cycle` k
        $\sigma_{xx,n,k,j} \leftarrow \Re(\phi'_{n,k,j}) - y_j \Im(\phi''_{n,k,j})$
        $\sigma_{yy,n,k,j} \leftarrow \Re(\phi'_{n,k,j}) + y_j \Im(\phi''_{n,k,j}) + \sigma_{max}$
        $\sigma_{xy,n,k,j} \leftarrow -y_j \Re(\phi''_{n,k,j})$
        `// Strain state at cycle` k `by Hooke's law`
        $\epsilon_{xx,n,k,j} \leftarrow \frac{1}{E}(\sigma_{xx,n,k,j} - \nu\sigma_{yy,n,k,j})$
        $\epsilon_{yy,n,k,j} \leftarrow \frac{1}{E}(-\nu\sigma_{xx,n,k,j} + \nu\sigma_{yy,n,k,j})$
        $\epsilon_{xy,n,k,j} \leftarrow \frac{1+\nu}{E}\sigma_{xy,n,k,j}$
        $\epsilon_{zz,n,k,j} \leftarrow \frac{-\nu}{E}(\sigma_{xx,n,k,j} + \sigma_{yy,n,k,j})$
        `// Strain measurement by gauge` j
        $\epsilon_{n,k,j} \leftarrow \epsilon_{xx,n,k,j}\cos(\theta)^2 + \epsilon_{yy,n,k,j}\sin(\theta)^2 + 2\epsilon_{xy,n,k,j}\cos(\theta)\sin(\theta)$
      `// Collect the data every` Δk `cycles`
      Append $(\epsilon_{n,k,j})_{j=1,...,n_g}$ to X_s
      Append k to X_c
      $k \leftarrow k + \Delta k$
      `// Analytical formula for the crack size of` n`-th structure at cycle` k
      $a_{k,n} \leftarrow [kC_n(1 - \frac{m_n}{2})(\sigma_{max}\sqrt{\pi})^{m_n} + a_{0,n}^{1-\frac{m_n}{2}}]^{\frac{2}{2-m_n}}$
    $X[n-1, 0] \leftarrow X_c$
    $X[n-1, 1] \leftarrow X_s$
  return X

29

3.4 Further investigations

In the illustration case study, the RUL prediction was considered as a time-series-related regression problem, but other approaches such as classification based are possible as well. In this section, a different formulation of the RUL estimation problem will be considered through a classification problem formulation.

3.4.1 Context

In the previous illustration, some of the most common DL models have been applied to the generated data sets and compared, including RNN, LSTM, GRU, 1D-CNN and TCN. The prediction of the RUL was a time series regression problem, as the objective was to predict a real-valued RUL based on the sensor data stream. Indeed in most articles reviewed in Chapter 2, RUL prediction has been essentially considered as a time-series-related regression problem, but other approaches such as classification based are possible. This approach has already shown promising results in time series forecasting [230].

Therefore, this study aims to investigate the effect of *output binning* (*i.e.* converting real-valued outputs into categorical ones) on the performance of the considered forecasting architectures. The problem of prognostics is framed as a classification task, in which the goal is, instead of trying to predict an exact point-wise RUL value at each prediction step, to predict the correct RUL class with a lower and upper bound.

3.4.2 Methodology

The models used in this classification problem are the same as those investigated previously in the regression case study (*i.e.* RNN, LSTM, GRU, 1D-CNN and TCN), with the same architecture and same parameters, only the output layer changes as the output changes. In this section, a description of the classification problem considered is provided as well as the way in which DL methods are implemented on it.

3.4.2.1 Problem Statement

Meaning that the problem of prognostics is framed as a classification problem, instead of trying to predict an exact point-wise RUL value at each prediction step, the goal is to predict the correct

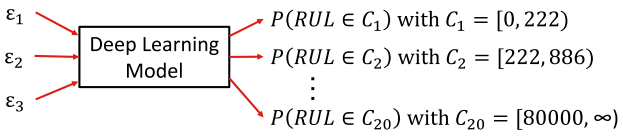
RUL class with a lower and upper bound for the RUL.

As a classification problem is considered, an intermediate task would be to generate RUL classes (or intervals) that can be used as prediction outputs. The structures in the generated dataset have a maximum total useful life value denoted T_{max} loading cycles. Therefore the chosen strategy is to create n_c intervals, where $(n_c - 1)$ of them follow a parabolic equation between 0 and T_{max} cycles. A last class is added for all RUL values greater than T_{max} cycles. The parabolic equation for class generation and the partitioning are provided in the following equation:

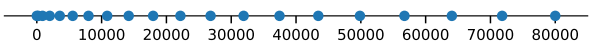
$$\begin{cases} C_{n_c-1} = [T_{max}\ ,\ +\infty[\\ C_j = [u_j\ ,\ u_{j+1}] & \forall j \in [|0\ ,\ n_c-2|] \\ u_0 = 0\ \ , u_{n_c-2} = T_{max}\ \text{ and } u_{j+1} = (u_j + \frac{u_{n_c-2}^{1-w} - u_0}{n_c - 1})^{\frac{1}{1-w}} \end{cases}$$

where C_j is the j^{th} interval, for $j = 0, ..., n_c - 1$, and w the parameter used to monitor the variation of the parabolic function. Note that T_{max} is assumed to be equal to 80000 in the generated dataset, and $w = 0.5$ in this work after some preliminary experiments.

The reason to use the parabolic equation for class generation is to generate more classes near an RUL value close to 0 as it is more critical to generate accurate predictions near the point of failure of the structures. Fig. 3.1 shows the principal output of the neural networks as well as the boundaries between the RUL classes in this work.



(a) Neural network model output for RUL classification



(b) Boundaries between the RUL classes (the dots represent the boundaries).

Figure 3.1: In this illustration, $T_{max} = 80000$, $w = 0,5$ and $n_c = 20$.

As in the previous illustration case study, a sliding window approach is used (see Section III-B

in Article 1). The classification models have the same inputs as the regression models used in the previous work, while the output is a class estimation of the RUL of the structure at time t.

3.4.2.2 Training and Evaluation

The input data is pre-processed similarly as in the previous illustration case study:

1. For the recurrent neural networks, the input data is pre-processed using a min-max scaler before being fed into the models.
2. The first layer of the 1D-CNN's architecture is a normalization layer.
3. The input data is not pre-processed for the TCN model and the same architecture as presented in [231] is kept.

Given that this problem is a multi-class classification problem, the aim is to minimize a categorical cross entropy (CE) loss L_{CE} during the training phase. The accuracy metric is used to evaluate the performance of the investigated models such that:

$$L_{CE} = -\sum_{j=0}^{n_c-1} y_{o,j} log(p_{o,j}) \tag{3.1}$$

$$\text{Accuracy} = \frac{1}{S}\sum_{i=1}^{S} 1_{RUL_{bin,i}=\hat{RUL}_{bin,i}} \tag{3.2}$$

where S is the number of samples with $\hat{RUL}_{bin,.}$ being the prediction and $RUL_{bin,.}$ the target value. For the classification-task loss function, $y_{o,j}$ is a binary indicator (0 or 1) if the sample o belongs to the class j, o_j the model output logit score that the observation o belongs to the class j, and $p_{o,j}$ the Softmax estimated probability that the observation o belongs to the class j such that $p_{o,j} = \frac{e^{o_j}}{\sum_{p=0}^{n_c-1} e^{o_p}}$.

In order to compare the classification models with the regression models, the resulting accuracy of the models is converted into an *MAPE* score such that:

$$MAPE_{Acc} = \frac{1}{S}\sum_{i=1}^{S} \left|\frac{y_i - H(\hat{y}_i)}{y_i}\right| * 100 \tag{3.3}$$

where y_i is the target RUL and $\hat{y}_i$ the estimated class, while the operator $H(.)$ is defined by $H(\hat{y}_i) = \frac{upper(\hat{y}_i) - lower(\hat{y}_i)}{2}$ which corresponds to the mean between the upper and the lower bound of the interval of the estimated class $\hat{y}_i$. Note that this conversion boils down to converting the class prediction to a point wise prediction which is equal to the center of the class's interval. With this procedure an equivalent *MAPE* can thus also be calculated for the classification models.

3.4.3 Experiments and Results

In this study, the same training strategy is used as previously (*i.e.* hyperparameters optimization followed by a fine-tuning phase). The training sets, validation set and testing set remain the same. The classification models keep the same optimal hyperparameters as the regression models (*e.g.* number of neurons or layers), only the output layer and the batch size during training change. Indeed in classification, the batch size was chosen to be as large as possible in order to speed up calculations, here $2^{12} = 4096$ depending on the available memory of the used GPUs, and not too large in order to avoid numerical instability; in regression, the batch size is set to 32: experiments have shown that in regression a high batch size quickly led to numerical instability during training. As a reminder, the optimal hyperparameters found for the recurrent neural network are summarized in Table 3.1, and those for CNN models in Table 3.2.

Training structures	100			500			1000		
Model	RNN	LSTM	GRU	RNN	LSTM	GRU	RNN	LSTM	GRU
Hidden layers	3	2	3	3	2	2	3	2	2
Nodes per layer	32	64	256	32	128	256	32	128	256
Dropout rate	0	0	0	0	0	0	0	0	0

Table 3.1: Best models of the RNN, LSTM and GRU hyperparameter optimization for 100, 500 and 1000 training structures

Training structures	100	500	1000
Convolutional layers	6	2	4
Filters per layer	20	40	25
Kernel size	6	12	9
Dropout rate	0.0	0.0	0.0

(a) Best models of the 1D-CNN.

Training structures	100	500	1000
TCN residual blocks	8	8	8
Filters per layer	30	25	35
Kernel size	2	2	2
Dropout rate	0.0	0.0	0.0

(b) Best models of the TCN.

Table 3.2: Hyperparameter optimization for 100, 500 and 1000 training structures.

3.4.3.1 Comparison in Classification Task

Table 3.3 shows an overview of the achieved accuracy for all recurrent neural networks (RNN, LSTM and GRU) and convolutional neural networks (1D-CNN and TCN).

Training structures	100		500		1000	
Accuracy (%)	Val	Test	Val	Test	Val	Test
Reccurent Networks						
RNN	**0.90**	**0.84**	0.91	0.90	0.89	0.78
LSTM	0.85	0.73	0.93	0.83	0.93	0.72
GRU	0.78	0.70	0.96	0.77	0.97	0.75
Convolutional Networks						
1D-CNN	0.78	0.82	0.84	0.84	0.82	0.84
TCN	0.81	0.82	**0.98**	**0.98**	**0.99**	**0.99**

Table 3.3: Accuracy of all fine tuned RNN (RNN, LSTM and GRU) and CNN models (1D-CNN and TCN) on the validation and testing datasets

RNNs outperformed the other algorithms when trained on 100 structures (84% testing accuracy), while the best performance in general was achieved by TCN models (98% testing accuracy when trained on 500 structures, and 99% testing accuracy when trained on 1000 structures).

The recurrent networks seem to have difficulties to generalize the patterns learned to new sets of data. They achieved good performances on the validation dataset, while when evaluated on

the testing dataset there is a significant drop in performance, especially when trained on 1000 structures. This may result from an overfitting on the validation set during the fine-tuning phase, thus leading to a lack of generalization from recurrent networks. Moreover, the overfitting is more noticeable when the number of network parameters (neurons) increases: greater difference for the GRU model (256 neurons, 3 layers for 100 structures and 2 layers for 500 and 1000 structures) than for the RNN model (32 neurons, 3 layers). Note that no dropout was used on recurrent networks, which may be a contributing factors to this overfitting.

For the 1D-CNN and TCN models there is a steady increase in the achieved accuracy on the testing dataset if the amount of structures for training is increased. The increase in testing accuracy corresponds largely to the increase in validation accuracy. The testing accuracy is in the same order of magnitude as the validation accuracy indicating that the networks generalize well to new sets of data. In general TCN work better than 1D-CNN, especially on the larger datasets of 500 and 1000 structures where the TCN achieve close to 100% accuracy.

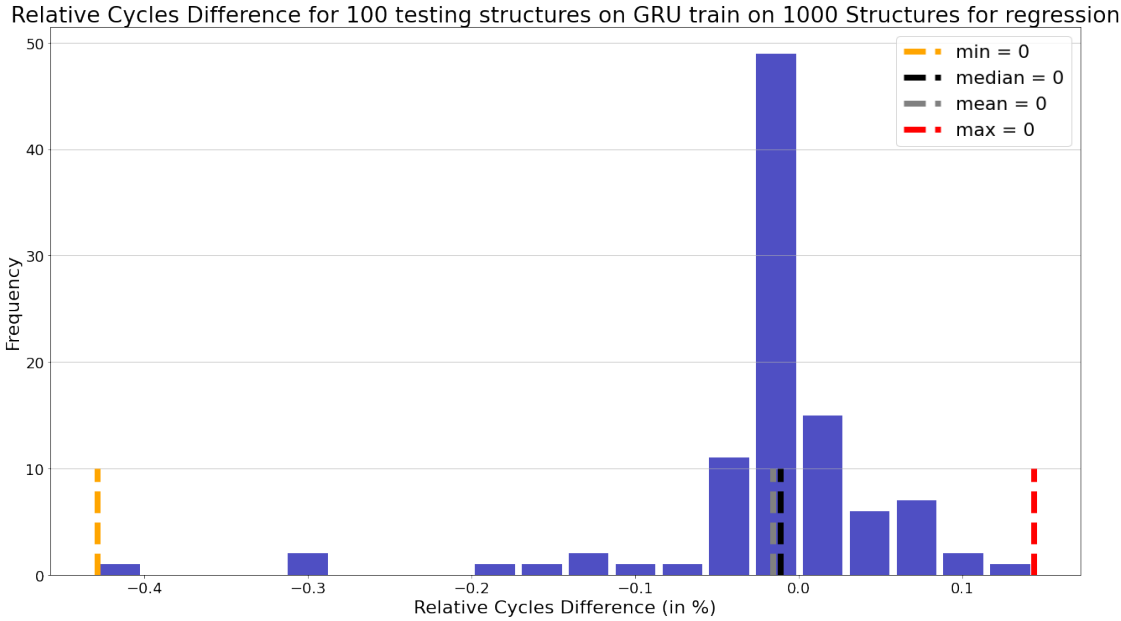
3.4.3.2 Comparison of the approaches

The performance of all the investigated models on the testing set for both approaches are then compared: classification and regression. As described in Section 3.4.2.2, the *MAPE* metric is used. The results are illustrated in Table 3.4.

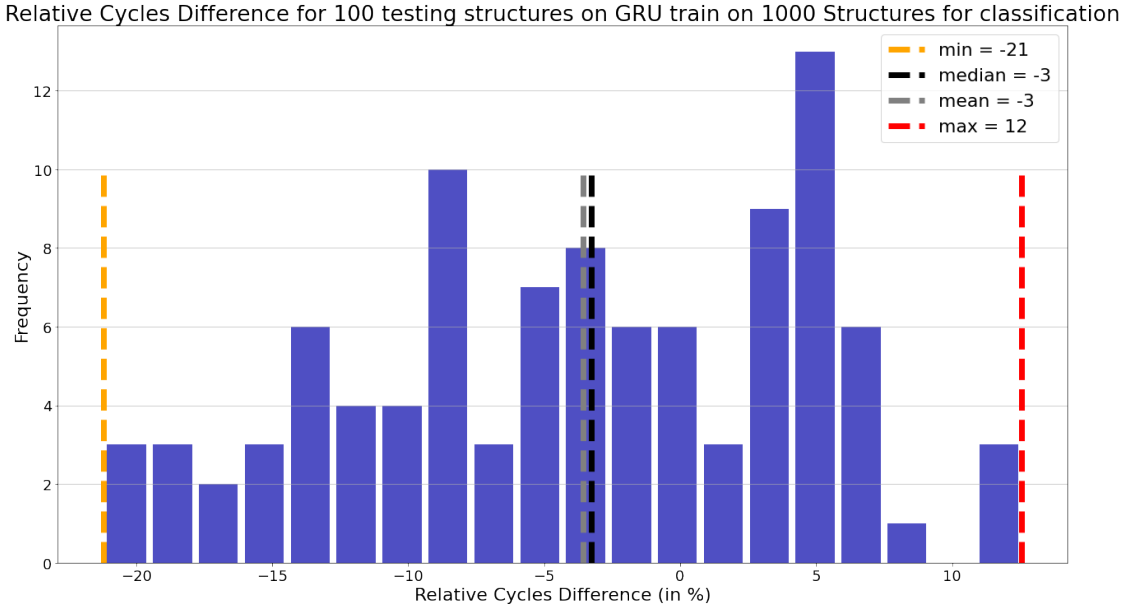
Training structures	100		500		1000	
Models	MAPE (%)	Acc	MAPE (%)	Acc	MAPE (%)	Acc
Classification models						
RNN	9.24	**0.84**	5.84	0.90	6.95	0.78
LSTM	7.32	0.73	6.51	0.83	7.06	0.72
GRU	7.83	0.70	6.92	0.77	7.39	0.75
1D-CNN	6.65	0.82	6.33	0.84	6.36	0.84
TCN	**6.62**	0.82	**5.55**	**0.98**	**5.56**	**0.99**
Regression models						
RNN	2.95	-	1.54	-	0.67	-
LSTM	**0.65**	-	1.15	-	1.24	-
GRU	1.22	-	**0.02**	-	**0.04**	-
1D-CNN	3.13	-	0.82	-	0.54	-
TCN	2.35	-	0.66	-	0.69	-

Table 3.4: RUL estimation performance of all fine tuned RNN (RNN, LSTM and GRU) and CNN models (1D-CNN and TCN) on the testing dataset.

The results in Table 3.3 show that even if the TCN model can achieve a performance close to 100% accuracy when trained on 1000 structures, the corresponding score in *MAPE* remains relatively high compared to the *MAPE* of the corresponding TCN-regression model. Overall, regression approaches seem to be more suited for the considered fatigue damage RUL estimation than classification approaches, according the *MAPE* metric used. So far, the GRU model seems to be the best performing model in pointwise RUL estimation when trained on more than 500 structures, based on the MAPE metric. Figure 3.2 illustrates the performance of the classification GRU trained on 1000 structures and the corresponding regression GRU.



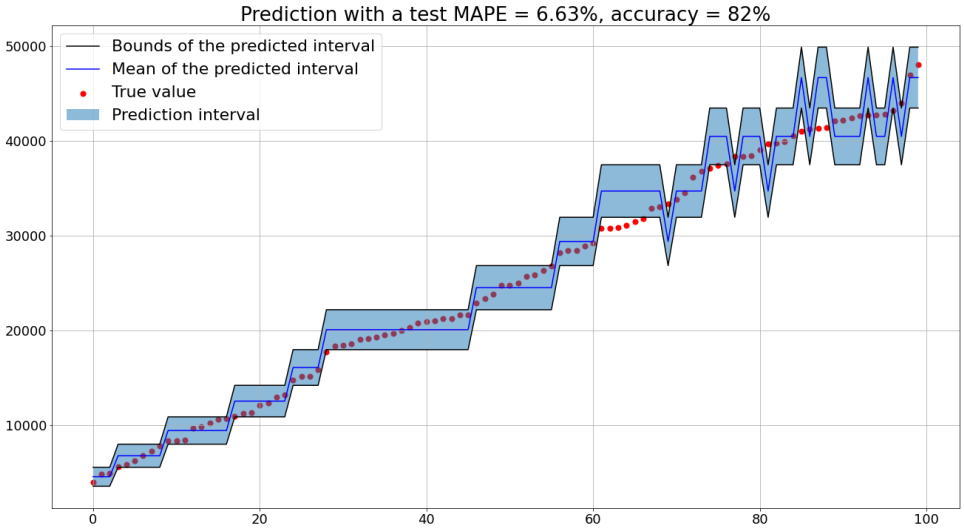
(a) GRU model in a regression task obtaining an MAPE score of 0.04%



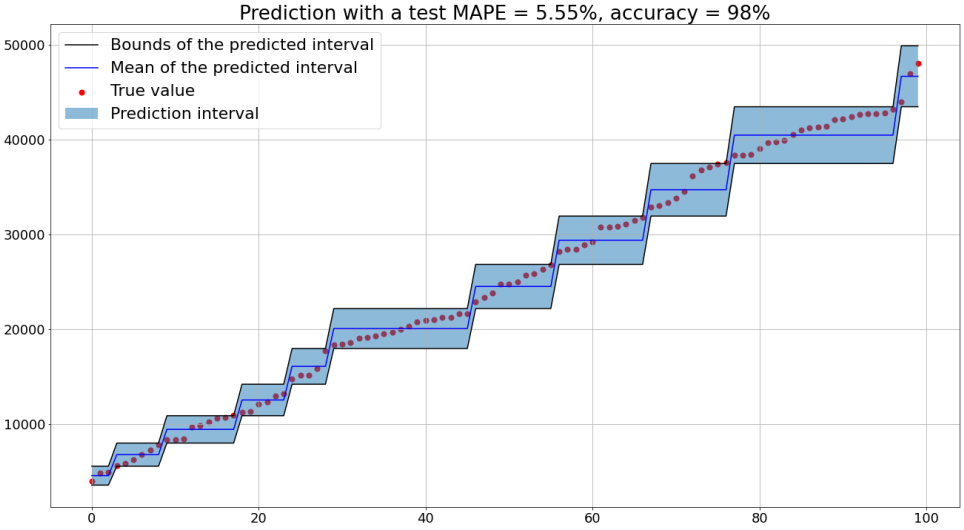
(b) GRU model in a classification task obtaining an MAPE score of 7.39%

Figure 3.2: Histograms depicting the relative difference between the actual RUL value and the estimation by the GRU model. The models were trained on 1000 structures and evaluated on 100 testing structures.

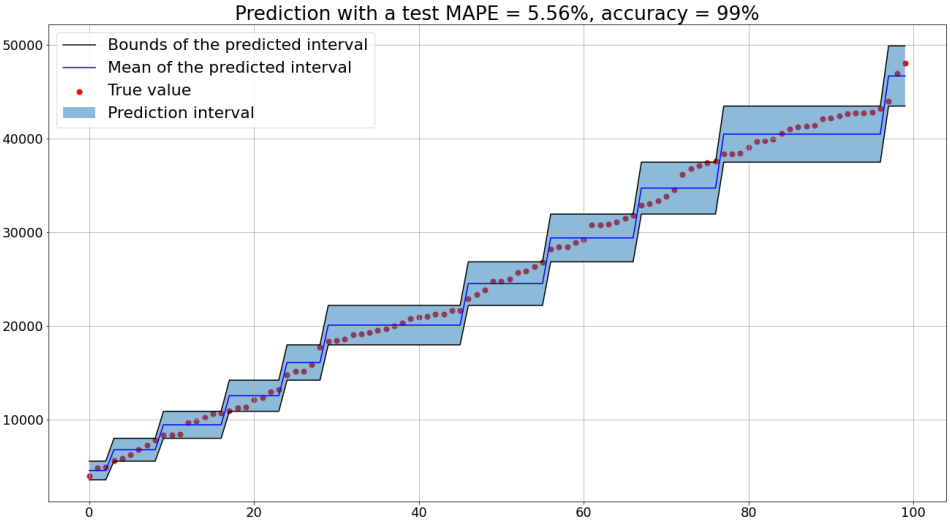
Note that the number of classes was fixed at a rather small number of classes for this first study (*i.e.* $n_c = 20$), which might explain the significant difference between the excellent performance of the TCN model in terms of classification accuracy and its relatively poor performance in terms of regression *MAPE* metrics. Fig. 3.3 illustrated the performances of TCN model in classification task.



(a) TCN classification model trained on 100 structures.



(b) TCN classification model trained on 500 structures.



(c) TCN classification model trained on 1000 structures.

Figure 3.3: Illustration of the performance of the TCN classification model in RUL estimation, using $n_c = 20$ classes. The blue area represents the prediction interval.

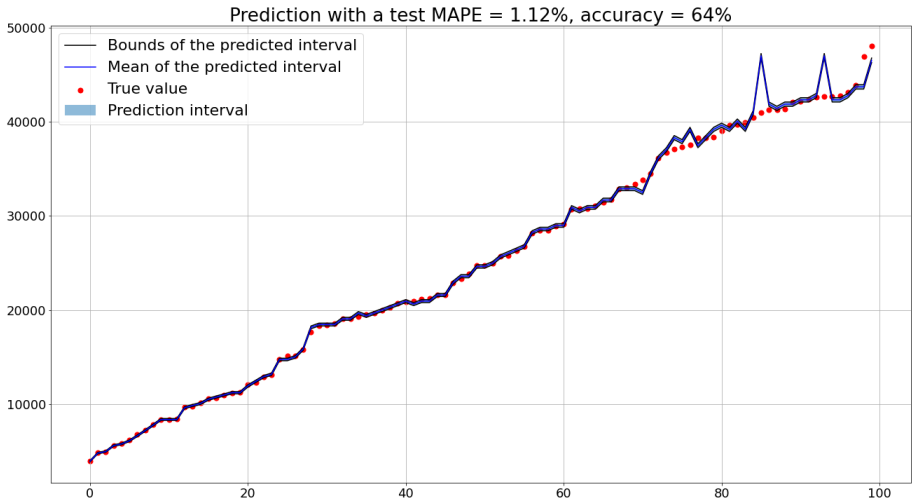
Therefore in order to investigate the effect of the number of classes on the results, the number of classes was varied and set to $n_c = 20, 64, 128, 256$ and 512. Given its very good results in classification, only the TCN model is considered in the following. The results are shown in Table 3.5.

Training structures	100		500		1000	
Models	MAPE (%)	Acc	MAPE (%)	Acc	MAPE (%)	Acc
Classification models						
RNN ($n_c = 20$)	9.24	**0.84**	5.84	0.90	6.95	0.78
LSTM ($n_c = 20$)	7.32	0.73	6.51	0.83	7.06	0.72
GRU ($n_c = 20$)	7.83	0.70	6.92	0.77	7.39	0.75
1D-CNN ($n_c = 20$)	6.65	0.82	6.33	0.84	6.36	0.84
TCN ($n_c = 20$)	6.62	0.82	5.55	0.98	5.56	**0.99**
TCN ($n_c = 64$)	2.00	0.83	1.57	**0.99**	1.58	**0.99**
TCN ($n_c = 128$)	**1.06**	0.73	0.83	0.93	0.82	0.98
TCN ($n_c = 256$)	1.12	0.64	0.39	0.93	0.38	0.97
TCN ($n_c = 512$)	1.26	0.26	**0.26**	0.76	**0.22**	0.85
Regression models						
TCN	**2.35**	-	**0.66**	-	**0.69**	-

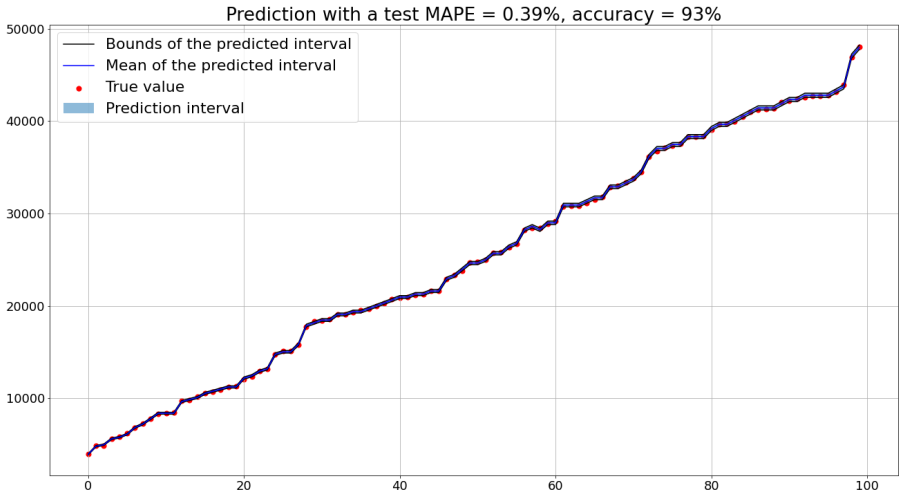
Table 3.5: RUL estimation performance of all fine tuned RNN (RNN, LSTM and GRU) and CNN models (1D-CNN and TCN) on the testing dataset.

Results showed that the *MAPE* steadily decreases when the number of intervals increases, reaching its best *MAPE* score when using 512 classes and trained on more than 500 structures.

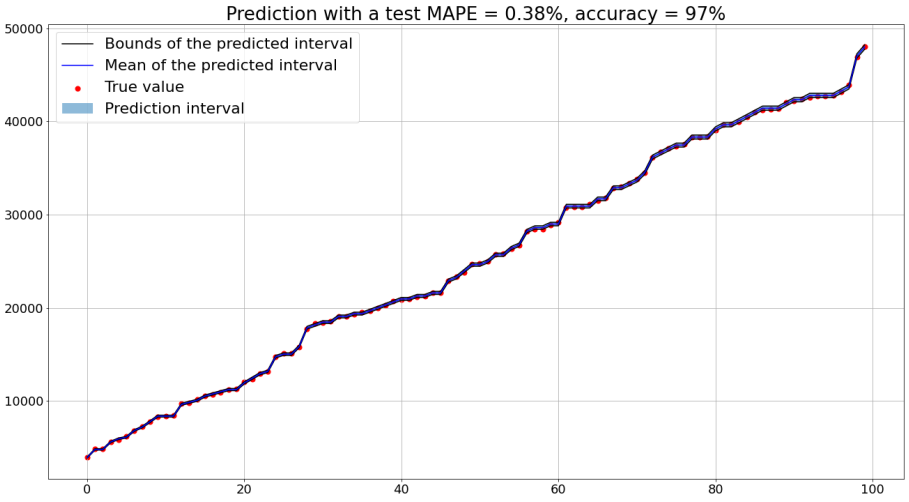
The TCN model reaches its best *MAPE* score when using 512 classes and trained on more than 500 structures, outperforming the best *MAPE* score obtained so far among both classification and regression approaches. A possible explanation for these results is that the estimation intervals become smaller as the number of classes increases, so the corresponding mean used for the MAPE metric becomes closer to the true RUL value. However, on the contrary, the model reaches its lowest performance in terms of accuracy metric using 512 classes. Indeed, results show that the best accuracy is reached for 64 classes (99%) and then drops significantly when the number of classes increases to 512 classes. Finally, the results, illustrated in Fig. 3.4, suggest that setting the number of intervals to $n_c = 256$ seems to be a fair trade-off between accuracy and point-wise RUL estimation.



(a) TCN classification model trained on 100 structures.



(b) TCN classification model trained on 500 structures.



(c) TCN classification model trained on 1000 structures.

Figure 3.4: Illustration of the performance of the TCN classification model in RUL estimation, using $n_c = 256$ classes. The blue area represents the prediction interval.

Furthermore, the author sought to understand why the accuracy is relatively low when the TCN model is trained on a high number of classes (*e.g.* 512), while the performance in *MAPE* is improving. Therefore, the gap between the true class and the estimated class by the TCN model is investigated, and illustrated in Fig. 3.5.

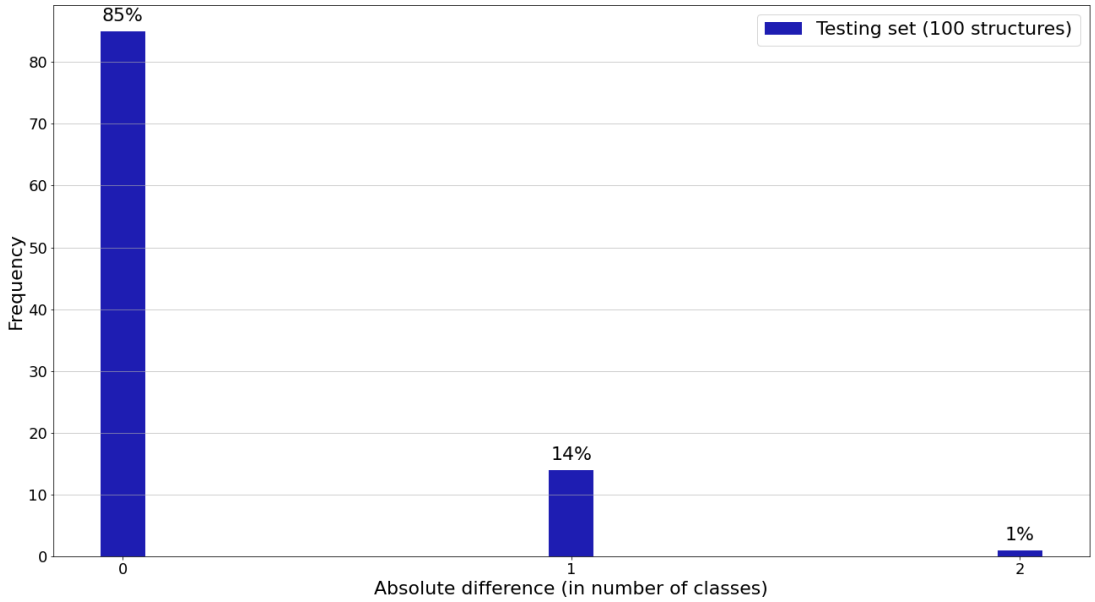


Figure 3.5: Histogram illustrating the gap (absolute difference in number of classes) between the true class and the estimated class by the TCN model trained on 100 structures. The model was evaluated on 100 testing structures.

Results in Fig. 3.5 show that most of the time, the predictions falls in the classes close to the target class (2 classes of difference at most). Results are similar when the model is trained on 500 or 1000 structures. Moreover, as the intervals of the classes are getting tighter when the number of classes increases (see Fig. 3.5), the accuracy of the model increases and the difference between the target value and the estimated value (mean of the interval) remains small.

As a final note, the comparison of all the approaches considered (cf. Table 3.6) indicates that regression approach remains the best suited for the current RUL estimation problem, especially GRU models.

Training structures	100		500		1000	
Models	MAPE (%)	Acc	MAPE (%)	Acc	MAPE (%)	Acc
Classification models						
RNN ($n_c = 20$)	9.24	**0.84**	5.84	0.90	6.95	0.78
LSTM ($n_c = 20$)	7.32	0.73	6.51	0.83	7.06	0.72
GRU ($n_c = 20$)	7.83	0.70	6.92	0.77	7.39	0.75
1D-CNN ($n_c = 20$)	6.65	0.82	6.33	0.84	6.36	0.84
TCN ($n_c = 20$)	6.62	0.82	5.55	0.98	5.56	**0.99**
TCN ($n_c = 64$)	2.00	0.83	1.57	**0.99**	1.58	**0.99**
TCN ($n_c = 128$)	**1.06**	0.73	0.83	0.93	0.82	0.98
TCN ($n_c = 256$)	1.12	0.64	0.39	0.93	0.38	0.97
TCN ($n_c = 512$)	1.26	0.26	**0.26**	0.76	**0.22**	0.85
Regression models						
RNN	2.95	-	1.54	-	0.67	-
LSTM	**0.65**	-	1.15	-	1.24	-
GRU	1.22	-	**0.02**	-	**0.04**	-
1D-CNN	3.13	-	0.82	-	0.54	-
TCN	2.35	-	0.66	-	0.69	-

Table 3.6: RUL estimation performance of all fine tuned RNN (RNN, LSTM and GRU) and CNN models (1D-CNN and TCN) on the testing dataset.

3.5 Discussion and Conclusion

This chapter proposed a framework for fatigue damage prognostics problems to address the challenge of data scarcity and provide an answer to the first refined research question. A framework and code for synthetically generating arbitrarily large data sets for a realistic fatigue damage prognostics problem has been presented. The proposed framework generates multivariate run-to-failure time series data for structures subject to fatigue loading, consisting of synthetic mechanical strain data sets (i.e. synthetic strain gauges) and associated RUL based on the Paris-Erdogan crack growth model.

As illustration, pre-cracked Aluminum alloy 7075-T6 plates were considered, which are typical of aeronautic structures, and the applicability of some of the most commonly used DL models to address failure prognostics (including RNN, LSTM, GRU, 1D-CNN, and TCN) have been studied. Also, different formulations of the RUL estimation problem were investigated, in terms of regression (point-wise estimates) or classification (estimation bounds). The current research has shown that the performance of prognostics algorithms strongly depend on the realization of the amount of labeled training data, where more available training data leads to a:

1. better performance of the models;
2. better identification of the most appropriate approach to use;
3. better identification of the most suitable models for the problem.

These results confirm the necessity of labeled data in fatigue damage prognostics problems. Nevertheless, none of the algorithms was found to be the best on both learning approaches. Indeed, Recurrent neural networks appear to be the best suited for regression task (especially GRU model), while the convolutional neural networks seem to be for classification tasks (especially TCN model) on this problem. So far, regression approach seems to be more suited for the considered RUL estimation than classification approach, but increasing the number of classes seems to improve the performance of classification models rendering them competitive with regression methods.

In this illustration case study, good results were expected and obtained, without the "flaws" of real world data such as noise. Hence, in future work, it would be interesting to complexify the dataset and making it as realistic as possible, by adding the noise or varying the initialization parameters such as:

1. **Crack propagation parameters**
    - Initial crack size a_0
    - Paris-Erdogan's law parameters m and C
2. **Strain gauges**
    - Number of the gauges placed
    - Position of the gauges placed
    - Angle of the gauges placed
3. **Generated data sets**
    - Number of generated structures (training, validation, testing)
    - Data collection interval Δk

The author believes that the proposed framework will help facilitate the benchmarking of latest ML algorithms for fatigue damage prognostics applications, in particular in the aerospace domain (e.g. fuselage panels). However, in engineering domains such as aerospace, while one of the main challenges of DL techniques is the difficulty of obtaining sufficient amounts of labelled data, the availability of unlabelled data is increasing due to the advancements in sensing technologies. Therefore, the second refined research questions is still to be answered and will be discussed in the following chapter.

CHAPTER 4

Application of Self-Supervised Learning in Fatigue Damage Prognostics Problems

"A pile of rocks ceases to be a rock pile when somebody contemplates it with the idea of a cathedral in mind."

Antoine Saint-Exupéry

Content

4.1 Motivation

Due to the advancements in sensing technologies, the availability of unlabelled data is increasing while labelled data is lacking. For recall, in our context labelled data is considered data providing the remaining useful lifetime (RUL) for each point in time, from the entry in service and up to failure. Exploiting unlabelled data (e.g. raw sensors data of structures replaced before the end of their service life) has therefore become a major goal in ML in order address data scarcity and improve learning performance.

To address this limitation, an emerging learning paradigm is investigated in the current chapter chapter: Self-Supervised Learning, a sub-category of unsupervised learning approaches (see Chapter 2). As mentioned in Chapter **2**, this approach has already shown tremendous performances in many fields such as in Natural Language Processing (*e.g.* GPT-3 [26]) or Image

Processing [27]. However, there is only a limited amount of existing research that focuses on the potential of Self-Supervised Learning for prognostics, and particularly for fatigue damage prognostics problems.

The objective of this chapter is to evaluate a possible solution to the second refined question: *is it possible to learn meaningful representations from unlabeled data and use it to enhance related supervised predictive tasks on a fatigue damage prognostics problem ?* The current research aims to investigate whether pre-training DL models in a self-supervised way on unlabelled sensors data can be useful for RUL estimation with only Few-Shots Learning, *i.e.* with scarce labelled data. In this research, data scarcity in a fatigue damage prognostics problem is addressed, and the application concerns estimating the RUL of aluminum alloy panels (typical of aerospace structures) subject to fatigue cracks from strain gauge data. The framework presented in the previous chapter is used in order to generate realistic synthetic datasets, allowing to investigate the influence of the dataset size on the predictive performance. Also, given the obtained results in the previous chapter, the RUL estimation problem will be considered as a regression problem, and the GRU networks will be re-used as the basic deep learning prediction model.

4.2 Self-Supervised Learning for Data Scarcity in a Fatigue Damage Prognostic Problem (Article 2)

The content in this section corresponds to a submitted work for publication. Reprinted, with permission, from *Anass Akrim, Christian Gogu, Rob Vingerhoeds and Michel Salaün*. "Self-Supervised Learning for Data Scarcity in a Fatigue Damage Prognostic Problem". *preprint submitted for publication*. This article is referred to as Article 2 in the current manuscript.

Self-Supervised Learning for Data Scarcity in a Fatigue Damage Prognostic Problem

Anass Akrim[a,b,*], Christian Gogu[a,b], Rob Vingerhoeds[b], Michel Salaün[a,b]

[a]*Institut Clément Ader (UMR CNRS 5312) INSA/UPS/ISAE/Mines Albi, Université de Toulouse, 3 rue Caroline Aigle, 31400 Toulouse, France*
[b]*ISAE-SUPAERO, Université de Toulouse, 10 Avenue Edouard Belin, 31400 Toulouse, France*

Abstract

With the increasing availability of data for Prognostics and Health Management (PHM), Deep Learning (DL) techniques are now the subject of considerable attention for this application, often achieving more accurate Remaining Useful Life (RUL) predictions. However, one of the major challenges for DL techniques resides in the difficulty of obtaining large amounts of labelled data on industrial systems. To overcome this lack of labelled data, an emerging learning technique is considered in our work: Self-Supervised Learning, a sub-category of unsupervised learning approaches. This paper aims to investigate whether pre-training DL models in a self-supervised way on unlabelled sensors data can be useful for RUL estimation with only Few-Shots Learning, *i.e.* with scarce labelled data. In this research, a fatigue damage prognostics problem is addressed, through the estimation of the RUL of aluminum alloy panels (typical of aerospace structures) subject to fatigue cracks from strain gauge data. Synthetic datasets composed of strain data are used allowing to extensively investigate the influence of the dataset size on the predictive performance. Results show that the self-supervised pre-trained models are able to significantly outperform the non-pre-trained models in downstream RUL prediction task, and with less computational expense, showing promising results in prognostic tasks when only limited labelled data is available.

Keywords:
Prognostics and Health Management (PHM), Remaining Useful Life (RUL), Deep Learning (DL), Data Scarcity, Self-Supervised Learning (SSL)

Nomenclature

Abbreviations

AE	Autoencoder
AR	Autoregressive
DGN	Deep Gated Recurrent Unit Network

*Corresponding author
Email address: anass.akrim@gmail.com (Anass Akrim)

DL Deep Learning

GRU Gated Recurrent Unit

$LSTM$ Long Short-Term Memory

$MAPE$ Mean Absolute Percentage Error

ML Machine Learning

MSE Mean Squared Error

$MSPA$ Multi-Steps Prediction Autoregressive

PHM Prognostics and Health Management

RNN Recurrent Neural Networks

RUL Remaining Useful Life

SSL Self-Supervised Learning

Notations

D_L Labelled Dataset

D_U Unlabelled Dataset

T_f Time of failure

X^L/y^L Labelled input signal/Corresponding RUL label

X^U Unlabelled input signal

Variables

d Ratio of the total lifetime of a sequence

h Length of the sliding window

n_g Number of sensor time series

N_L Number of labelled structures

n_L Number of labelled samples

N_L^{Test} Number of labelled structures for testing

N_L^{Train} Number of labelled structures for training

N_U Number of unlabelled structures

n_U Number of unlabelled samples

N_U^{Train} Number of unlabelled structures for training

n_U^{Train} Number of unlabelled samples for training

1. Introduction

Prognostics and Health Management (PHM) is a reseach domain addressing failure mechanisms of real systems in order to better manage the use of information on equipment operating conditions [1]. Its implementation can improve the efficiency of maintenance support [2], optimize the maintenance plan and therewith equipment availability [3], help industry to balance safety and economic profit [4]. For many mechanical structures and notably aerospace structures, fatigue damage is one of the major modes of failure. Therefore, fatigue monitoring and prediction of fatigue life in structures, *i.e.* Remaining Useful Life (RUL) estimation, represents one of the major challenges to be solved for paving the way towards predictive structural maintenance.

Among the approaches used for PHM, Data-Driven models have gained more and more attention in the PHM community, especially the latest Deep Learning (DL) techniques [5], redefining state-of-the-art performances in a wide range of areas in recent years [6]. However, their effectiveness depends on the quantity and quality of available labelled data. Currently, data scarcity represents a scientific bottleneck in many engineering fields (*e.g.* in healthcare [7], in energy [8], water and environmental engineering [9, 10], etc.), which makes it difficult to apply the latest Machine Learning (ML) methods. Many approaches have been proposed to address data scarcity in these various domains, as recently reviewed by [11–13]. As faults are rare and structures can be replaced before reaching failure, data scarcity is becoming one of the most important challenges in PHM [14, 15]. Nevertheless, while labelled data is lacking, the availability of raw sensors data is increasing due to the advancements in sensing technologies. This data is considered as "unlabelled" in the context of prognostics as, for sensor data at a given point in time, the true RUL is unknown and cannot be determined unless the sensor measurements are available all the way to failure. In most engineering applications, this is unattainable, since the parts will be replaced before failure, and this is particularly true for aerospace mechanical structures, thus the majority of sensor data is unlabelled, meaning that no associated RUL is available for it. Exploiting such unlabelled sensors data during training has become a major goal in ML in order to improve learning performance. Therefore, the research question addressed in this paper can be stated as follows: *is it possible to learn meaningful representations from unlabelled data and use it to enhance related supervised predictive tasks on a fatigue damage prognostics problem?*

In the Artifical Intelligence (AI) community, a recent learning technique to extract knowledge from unlabelled data was proposed to address the challenge of data scarcity: Self-Supervised Learning (SSL) [16], a sub-category of unsupervised learning approaches. SSL has already shown tremendous performances in many AI fields such as in Natural Language Processing (*e.g.* GPT-3 [17]) or Image Processing [18]. Nevertheless, the applicability of this approach remains largely unexplored in the engineering fields, a domain in which data scarcity is an increasingly challenging issue [9, 19, 20]. Currently, there is only a limited amount of existing research that focuses on the potential of Self-Supervised Learning for Prognostics [21, 22], and particularly for fatigue damage prognostics problems.

In order to address this limitation, this paper aims to investigate whether pre-training

DL models in a self-supervised way on unlabelled sensors data on a fatigue damage prognostics problem can be useful for RUL estimation with only Few-Shots Learning, *i.e.* with scarce labelled data. The interest is in estimating the RUL of aluminum alloy panels (typical of aerospace structures) subject to fatigue cracks from strain gauge data. A synthetically generated dataset is used for this purpose, composed of a large unlabelled dataset (i.e. strain gauges data of structures before failure) for pre-training, and a smaller labelled dataset (i.e. strain gauges data of structures until failure) for fine-tuning on the RUL prediction task. The synthetic dataset is based on a framework previously developed by the authors [23].

The remainder of the paper is structured as follows. Section 2 provides a background on the state-of-the-art DL techniques in prognostics for PHM. In Section 3, the proposed methodology is presented. Section 4 presents the experimental settings used in this study for pre-training and fine-tuning phases, and the results obtained with the deep learning-based approaches trained in a self-supervised manner are analyzed. The impact of the size of the data available for pre-training as well as the choice of the pre-text task will be investigated, and different DL models are compared for the self supervised learning task. Section 5 summarizes the aspects of the approach considered in this paper and identifies potential future work. Finally, Section 6 concludes the current research paper and provides future outlooks. Fig. 1 illustrates the outline of this paper and summarizes the objective of each section.

2. Background

In this section, the application of DL techniques in the field of prognostics for PHM and related work on Self-Supervised Learning are presented.

2.1. Deep Learning in Prognostics for PHM

As more data becomes available in the engineering domain, there is a recent surge of interest in using Deep Learning in Prognostics and Health Management [24, 25]. In prognostics applications, Time Series Forecasting models are most commonly used to predict the RUL of systems or structures, given the format of acquired data in PHM (*e.g.* data collected from sensors, vibration signals, etc.), and the most commonly used algorithms for these tasks are Recurrent Neural Networks (RNN) [26]. Given the sequential nature of the sensor data in the prognostics field (*e.g.* sensors data), good results have been obtained within the PHM community by using RNNs, such as Standard RNNs, Long Short-Term Memory (LSTM) networks, and Gated Recurrent Unit networks (GRU) [14, 27, 28].

The lack of available labelled data is becoming a major challenge in the application of machine learning to PHM, which, as a field, suffers from a high data acquisition cost compared to other domains in which machine learning has proven useful (*e.g.* Natural Language Processing). In Prognostics tasks, a label can constitute the RUL at each time step of measurements, which is generally difficult to acquire and often can be a time-consuming and expensive investment for experts. However, due to advancements in sensing technologies in engineering fields, the availability of unlabelled data is increasing (*e.g.* raw sensors data of structures replaced before reaching failure). Since failure is not reached when the structures are replaced, the RUL at replacement and at each previous

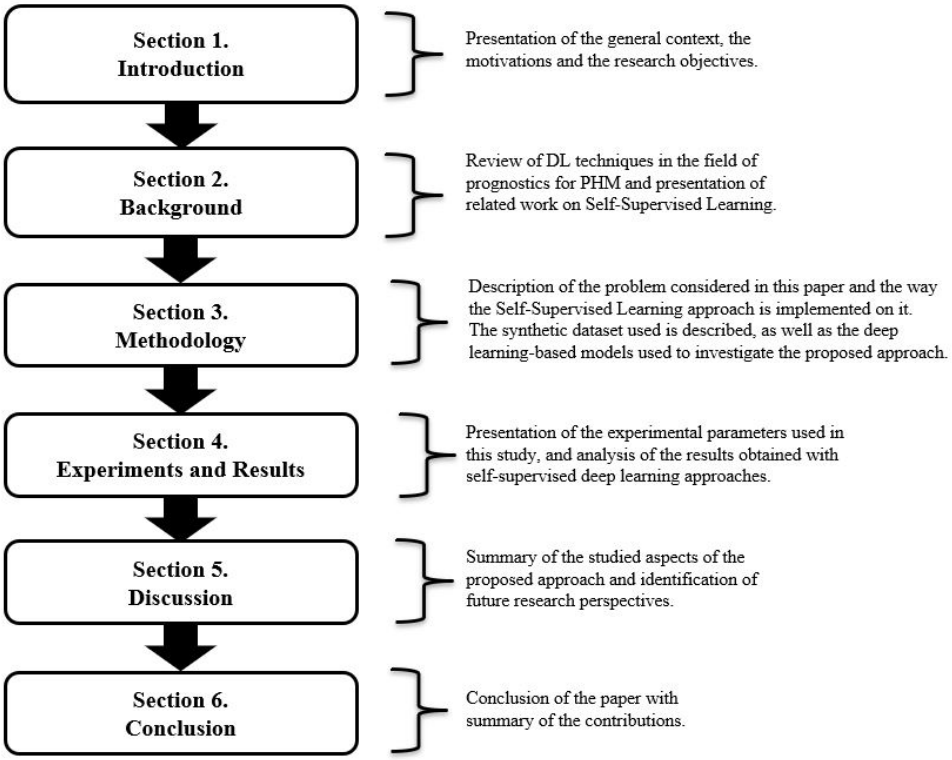


Figure 1: Outline of this paper.

timestep is not known. Thus the sensor data is unlabelled according to the previously introduced definition of a label in the PHM context. Exploiting unlabelled data during training has therefore become a major goal in order to improve learning performance.

2.2. Self-Supervised Learning

Similarly to *self-taught learning* as presented in [29], Self-Supervised Learning (SSL) consists in learning meaningful and general representations from unlabelled data (during a *pre-training phase*) by solving a so-called *pretext task* without requiring the data to be labelled. These representations are then applicable to a wide range of related supervised tasks (*i.e. downstream task*) with only few labelled data (*i.e.* "Few-Shots Learning"). SSL aims to improve predictive performance on the downstream task through the use of unlabelled data, thus avoiding the extensive cost of collecting and annotating large-scale datasets [30]. This learning paradigm has already proven that it can significantly improve the performance of downstream tasks for many AI applications such as in Natural Language Processing (*e.g.* GPT-3 [17]) or Image Processing [18, 31]. GPT-3 [17] was one of the largest self-supervised learning systems released by OpenAI in 2019.

There are few recent developments that have shown the potential of the SSL paradigm in engineering fields [32–35], and several PHM researchers considered this approach to address the data scarcity in fault diagnostics problems, showing promising results [36, 37]. However, to date, there is, to the best of the authors' knowledge, only a limited amount of existing research that focuses on the application of SSL to prognostic problems in PHM, for example for RUL estimation on the NASA C-MAPSS[1] dataset [39, 40, 21, 22]. One of the first to explore this learning paradigm in prognostics in order to deal with the problem of lack of labelled data is Yoon et al. [39], using a pre-trained variational autoencoder (VAE [41]) that makes use of available unlabelled data to learn a latent space representation in an unsupervised manner; the pre-text task being the minimization of the reconstruction error. The extracted features by the VAE model are then fed as inputs to an RNN model for RUL estimation, trained in a supervised manner by varying the fraction of labelled engines data down to 1% in order to investigate if the SSL approach can enhance predictive tasks when only a small amount of labelled data is available. Results showed that their proposed method was able to outperform a non pre-trained supervised RNN-model when all the labelled dataset is available as well as in other scenarios when the available labelled data is highly limited with only a small labelled fraction of the training data. The authors in [40] used a Restricted Boltzmann Machine model (RBM) [42] for pre-training on unlabelled dataset with a reconstruction pre-text task, and an LSTM model for RUL prediction. Results showed that this SSL approach could improve the RUL prediction accuracy compared to the purely supervised learning approach (*i.e.* predictive model without the initial pre-training stage), both when the training data is completely labelled and when the labelled training data is reduced. It is worth noting that the methods proposed in [39] and [40] were not presented as SSL approaches, but are considered as such in this paper since the proposed methods follow the same procedure as

[1]NASA C-MAPSS [38] is a publicly available dataset of simulated turbofan engines commonly used to benchmark RUL estimation algorithms. The dataset is divided into four subsets (FD001 - FD004) of different operating conditions and possible fault modes.

described earlier. However, Krokotsch et al. [21] highlighted two shortcomings of these two previous studies:

1. the approaches were evaluated only on one subset of the C-MAPSS dataset out of four in each study, rendering these investigations limited;
2. pre-training was performed on unlabelled data of engines that contain the point of failure, which should not be the case in real scenarios, since the RUL labels for all the data could be deduced based on the knowledge of the failure time.

To overcome these limitations in [21], the investigation was performed over all subsets of the C-MAPSS data set and the unsupervised pre-training phase was performed over truncated time series, assuming that realistic unlabelled data does not contain features near the time of failure (corresponding to sensors data of structures replaced before reaching failure). Results showed that:

1. the proposed SSL approach can outperform the supervised baseline that used only the labelled data. Both approaches were trained on only few labelled time series for RUL estimation (*i.e.* Few-Shots learning);
2. the proposed pre-training model outperformed two competing pre-training models, including AE and RBM using a reconstruction pre-text task (*i.e.* the output y corresponds to an estimation of the input x).

These results suggest that the choice of the pre-training model (or pre-text task) matters. Recently, Guo et al. [22] proposed a pre-training method based on masked autoencoders [43] to perform SSL on the C-MAPSS datasets. Results showed that their pre-trained model outperformed the fully supervised model in RUL estimation. Unfortunately, there are no clear guidelines for selecting the right pre-text task that learns meaningful representations from unlabelled time series data (*e.g.* sensors data) during the pre-training phase. Furthermore, one of the main challenges for extensive investigations on the potential of SSL in PHM resides in the difficulty of having scalable open-source dataset, similar to those available in Natural Language Processing or Image Processing. Thus, despite demonstrating encouraging results, the domain of SSL is still largely unexplored in the prognostics field and is in contrast with the increasing amount of unlabelled data available in industry, having the potential to enable predictive maintenance. Table 1 summarizes the applications of self-supervised learning in PHM identified in this paper.

Authors	Year	Downstream task	Pre-training tasks	Application
Yoon et al. [39]	2017	RUL estimation	Reconstruction of the input signal (variational autoencoder).	Turbofan engines
Ellefsen et al. [40]	2019	RUL estimation	Reconstruction of the input signal (restricted boltzmann machine).	Turbofan engines
Krokotsch et al. [21]	2022	RUL estimation	Reconstruction of the input signal (autoencoder and restricted boltzmann machine); learn a distance or similarity metric between pairs of data (siamese network).	Turbofan engines
Guo et al. [22]	2022	RUL estimation	Reconstruction of the input signal (masked autoencoder).	Turbofan engines
Hahn et al. [36]	2021	Fault diagnostics	Reconstruction of the input signal (variational autoencoder).	Milling tools
Ding et al. [37]	2022	Fault diagnostics	Contrastive learning (deep convolutional network).	Bearings

Table 1: Summary of identified applications of SSL approaches in PHM.

3. Methodology

The authors of this paper seek to advance the field of data scarcity in fatigue damage prognostic problems by investigating Deep Self-Supervised Learning on an associated RUL estimation problem. In this section, a description of the dataset involved is provided, followed by a description of the problem considered in this paper and the way the Self-Supevised Learning approach is implemented on it. The deep learning-based models used to investigate the SSL approach are also presented and detailed in this section. Note that data and code for the learning procedure are publicly available on `https://github.com/ansak95/DeepSSL`.

3.1. Data Description

In the current research study, a synthetic dataset for a realistic fatigue damage prognostics problem is generated, based on a framework previously proposed by the authors [23]. It consists of synthetic multivariate run-to-failure time series data for structures subject to fatigue crack propagation (*e.g.* fuselage panels). Indeed, the proposed framework generates synthetic data sets of mechanical strain data (i.e. virtual strain gauges), by simulating the crack growths in structures based on the Paris-Erdogan model [44]. Strain data was considered as sensor data since we consider a mechanical fatigue propagation problem and strain data is one of the main, easily measurable, quantities of interest allowing to determine crack propagation. Furthermore, strain gage measurement is a mature technique that can be relatively easily implemented on various kinds of structures. The

strain data, or measurement sequences, are obtained until the crack size a reaches the critical crack size a_{crit}, considered as the time of failure (necessary to compute the RUL at each time step for example). Finally, the generated strain data are used as sensors time series data available for prognostics problem such as RUL estimation. This setup can be seen representative of real experiments under fatigue loading where the strain state is monitored at multiple strain gauge positions (blue, orange and green crosses), illustrated in Fig. 2.

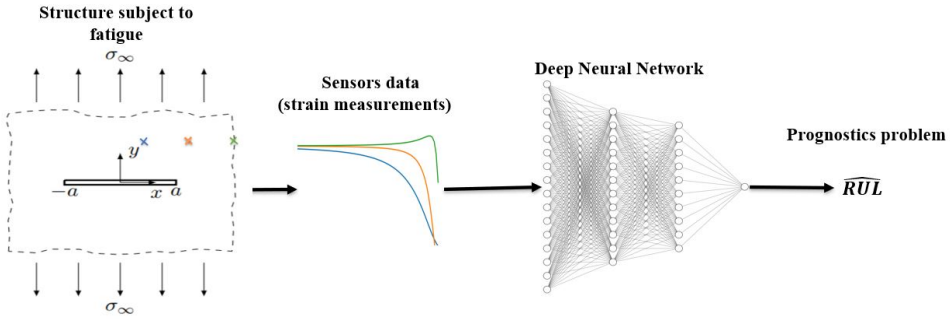


Figure 2: Illustration of 3 run-to-failure time series generated (*i.e.* strain data) used as sensors data, and as input for prognostic problems (*e.g.* RUL estimation).

In the current research, the multivariate dataset used contains the variations of the strains at $n_g = 3$ positions in the panel as a function of the number of cycles, where n_g is the number of the time series. More details about the dataset are given in [23], and an illustration of a generated sequence (*i.e.* three placed gauges) for a single structure until failure is given in Fig. 3.

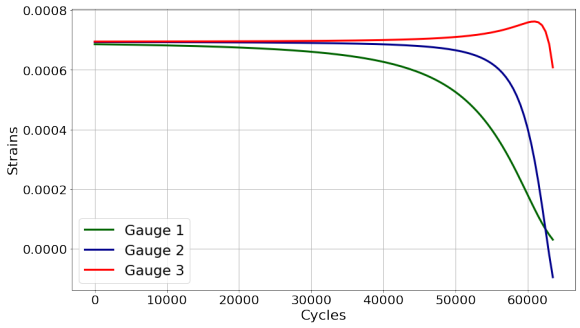


Figure 3: Strain values time series corresponding to a random sensor sample generated reaching failure (*i.e.* a labelled sequence).

Given the sequential nature of the sensors data, the time series generated are processed sequentially on a *sliding window* approach of size h: at each time-step t, the input of the predictive models corresponds to the current and past measurements, such that $X_t := (x_{t-h+1}, \ldots, x_t) \in \mathbb{R}^{n_g \times h}$ where $h = 30$ is the length of the sliding window (note

that the value of parameter h was set after preliminary experiments). Fig. 4 illustrates the sliding window approach used in this work.

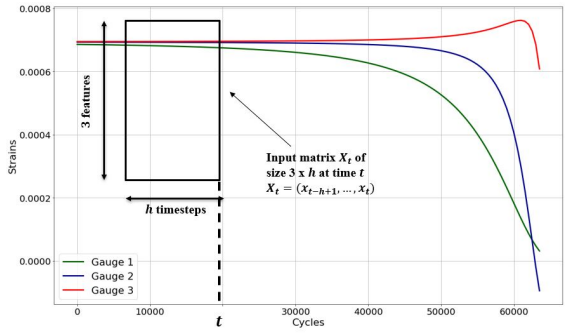


Figure 4: Illustration of the processing of the input data using a "sliding window" of size h to predict the corresponding output at each timestep t.

3.2. The proposed Self-Supervised Learning Approach

The Self-Supervised learning paradigm aims to extract useful features from unlabelled data in a self-supervised manner that can subsequently benefit supervised training on few labelled samples. Hence it is typically composed of:

1. *a pre-training phase*: a predictive data-driven model is trained on a raw unlabelled dataset in an unsupervised (or *self-supervised*) manner in order to learn abstract features.
2. *a fine-tuning phase*: the pre-trained model is coupled to a non-pre-trained model (*e.g.* for neural networks a linear layer or data-driven model) and then trained on a set of labelled data in a supervised manner.

The pre-training in SSL is essentially performed with deep learning models. Indeed, the architecture of DL models is in the form of a stack of layers of neurons, and the last layer is used to obtain the final output. Knowledge transfer is typically performed by removing this last layer and replacing it with a new non-trained output linear layer (or a predictive model). The working of SSL can be illustrated in Figure 5. This strategy allows to reuse the learned knowledge in terms of global architecture of the pre-trained network, which works as a *features extractor*, and to exploit it as a starting point for a downstream task (*i.e.* fine-tuning phase). It also provides faster learning time in downstream predictive tasks compared to non-pre-trained models, since it is not necessary to train the pre-trained layers but only the new output linear layer (or predictive model). This aspect will be discussed in Section 4.4.2. Note that some machine learning models are not suitable for pre-training in SSL paradigm, as their architecture is not composed of layers that can be easily extracted and reused for knowledge transfer (*e.g.* Support Vector Machines [45], Random Forests [46], Gaussian Processes [47]). Nevertheless, there are recent developments of these models that can be used in knowledge transfer (*e.g.* Deep Gaussian Processes [48, 49]).

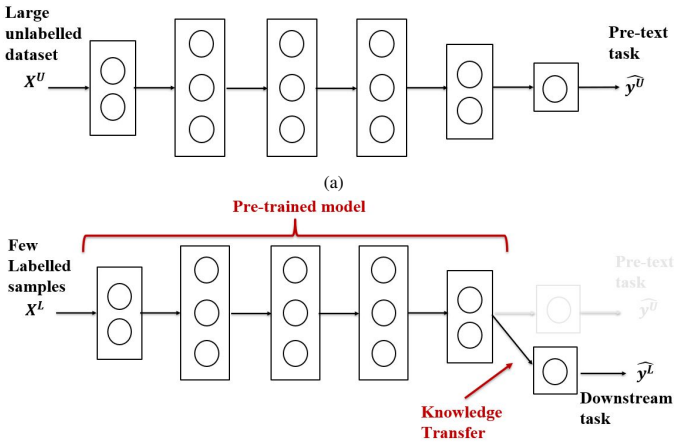


Figure 5: A schematic view of the Self-Supervised Learning procedure (a): Pre-training phase in a self-supervised way, (b): Fine-tuning phase (supervised training on downstream tasks).

3.2.1. Problem statement

To clearly formulate the problem, the synthetic data used in this work is composed of:

1. A large set of unlabelled data $D_U = \{X_i^U\}_{i=1}^{n_U}$, where n_U is the number of unlabelled samples, $X_i^U \in \mathbb{R}^{n_g \text{ x } h}$ the input signal with n_g sensors and h time steps. The unlabelled set D_U refers to strain measurement sequences of structures before reaching failure.
2. A smaller set of labelled data $D_L = \{(X_i^L, y_i^L)\}_{i=1}^{n_L}$, where n_L is the number of labelled samples, $X_i^L \in \mathbb{R}^{n_g \text{ x } h}$ the input signal with n_g sensors and h time steps, $y_i^L \in \mathbb{R}$ the corresponding RUL label. The labelled set D_L refers to strain measurement sequences of structures until failure.

Note that the samples of both domains D_U and D_L are multivariate time series sampled from related distributions.

Therefore in this paper, the pre-training phase of the proposed SSL approach consists of pre-training a DL model on unlabelled sensors dataset D_U in a self supervised manner, called *pre-text task*. The pre-trained model is then fine-tuned on a specific downstream Prognostics task, *i.e.* RUL estimation, using only limited amounts of labelled data (*i.e.* strain data of structures until failure, on which the RUL is known at each timestep). The pre-trained model is then fine-tuned on a specific downstream Prognostics task, *i.e.* RUL estimation, using only limited amounts of labelled data D_L. In this work, Deep Gated Recurrent Unit (GRU [27]) networks, or DGN, are used as the basic deep prediction model, because of their sequential properties and good regressive performance found in previous work [23]) (see Appendix A for more details about the GRU networks). Note that a DGN consists of a stack of GRU layers in this work. Fig. 6 summarizes the proposed SSL approach in this paper.

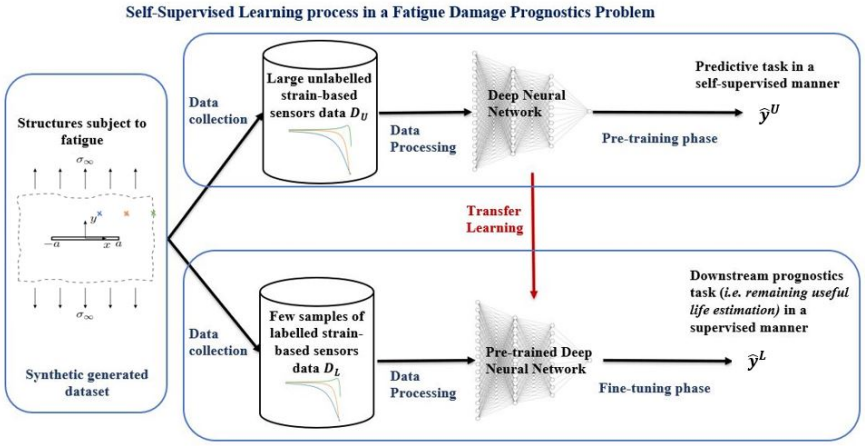


Figure 6: Flow chart of the proposed Self-Supervised Learning framework.

3.2.2. Pre-training phase

In order to vary the pre-text tasks and inspired by [50, 31], two types of models are used and compared in this work: 1)Autoencoders (AE) and 2)Autoregressive (AR) models.

3.2.2.1. Autoencoder architecture in pre-training phase. An Autoencoder (AE) is an artificial neural network that is often used in learning the discriminating features of a dataset in an unsupervised manner [51]. It is composed of two blocks: encoder and decoder (see Fig. 7 for a simplified architecture of the model). The encoder seeks to learn the underlying features of the input data X_t at time step t. These learned features z_t are generally of reduced dimension (number of neurons less than the number of input features). The goal of the decoder is thus to recreate the original data from these underlying learned features. In recent years, Autoencoders have been successful in prognostics applications in terms of feature extraction [52–54], which motivated the use of its architecture as a reference model for abstract representation learning in this work.

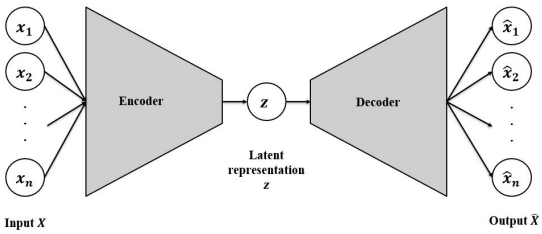


Figure 7: The architecture of basic Autoencoders. Note that encoders and decoders can be composed of one or more hidden layers.

In pre-training, the output of the Autoencoder (AE) is an estimation of the unlabelled input signal $X_t^U = (x_{t-h+1}^U, ..., x_t^U)$ such that $y_t^U = X_t$. A schematic view of the investigated

AE model in the proposed SSL framework is given in Fig. 8.

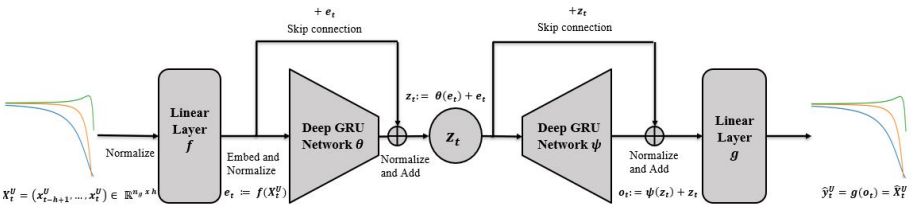


Figure 8: Flow chart of the pre-training phase of the Autoencoder model (AE) in the proposed Self-Supervised Learning framework.

The architecture of the AE model is organized as follows:

1. The input data X_t^U is first embedded through a linear layer[2] f in order to expand the dimension of the data and learn abstract features;
2. The output of the following layer e_t corresponds to a normalized[3] transformation of the embedded input $f(X_t^U)$;
3. The resulting embedded and normalized transformation of the data e_t is then fed to an encoder θ and decoder ψ. Note that both encoder and decoder are Deep GRU networks (DGN), *i.e.* stack of GRU layers;
4. z_t is considered as the learned representation by the model and will be used for feature extraction in the following;
5. In this architecture, two skip connections[4] are used through deep GRU networks such that $z_t = \theta(e_t) + e_t$ and $o_t = \psi(z_t) + z_t$;
6. The output linear layer g is then used to generate an estimation $\hat{y}_t^U$ of the unlabelled input signal X_t^U, *i.e.* an estimation of the input signal such that $\hat{y}_t^U = \hat{X}_t^U$.

3.2.2.2. Autoregressive architecture in pre-training phase. An Autoregressive (AR) model g_h can be defined as a sequential model governed by an Autoregressive process of order h that models the future outcome of a sequence at time $t + 1$, using its previous h realizations. Autoregressive modeling captures the temporal dependencies between sequential input data, which makes it useful in learning better features. Inspired by the autoregressive DL models used in [17, 31] for pre-training, the proposed AR model in this paper consists of a Deep GRU network in which the input is a sequence of h time steps at t such that $X_t^U = (x_{t-h+1}^U, ..., x_t^U)$, and the output is an estimation of the data of the next timestep such that $y_t^U = x_{t+1}^U$, according to the following formula:

[2]The input linear layer is used as an alternative to the embedding layers used in Natural Language Processing [55] since the input data is continuous is this work, converting each time step data into a fixed length vector of defined size.

[3]The layer normalization [56] are used for regularized training and faster convergence

[4]Skip connections in DL architectures, also called *residual connections* or *shortcut connections*, consist in skipping some layers in the neural network and feeding the output of one layer as the input to the next layers [57], used to solve the *degradation problem* (e.g. ResNet [58]). In this paper, skip connections are proposed to establish a direct connection through deep GRU networks in order to avoid information loss and learn robust sequential representation, which has already proven to be effective for deep recurrent neural networks [59]

$$\hat{y}_t^U = \hat{x}_{t+1}^U = \mathcal{F}_h(x_t^U, x_{t-1}^U, \ldots, x_{t-h+1}^U) \tag{1}$$

where $\mathcal{F}_h$ denotes the AR model governed by an autoregressive process of order h. A schematic view of the investigated AR models in the proposed SSL framework is given in Fig. 9.

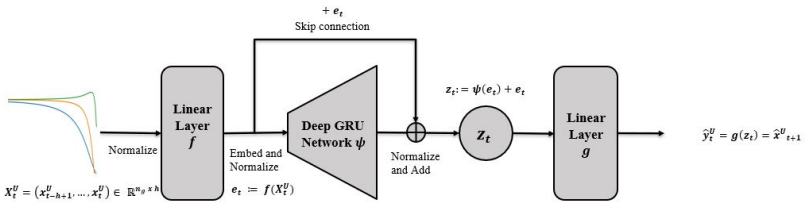


Figure 9: Flow chart of the pre-training phase of the Autoregressive model (AR) in the proposed Self-Supervised Learning framework.

The architecture of the AR model is organized as follows:

1. The embedding linear layer f is used in order to expand the dimension of the input data and learn abstract features;
2. The output of the following layer e_t corresponds to a normalized transformation of the embedded input $f(X_t^U)$;
3. The resulting embedded and normalized transformation of the data e_t is then fed to a Deep GRU Network ψ, composed of a stack of GRU layers;
4. z_t is considered as the learned representation by the model and will be used for feature extraction in the following;
5. In this architecture, a skip connection is used such that $z_t = \psi(e_t) + e_t$;
6. The output linear layer g is then used to generate an estimation $\hat{y}_t^U$ of the input signal X_t^U, *i.e.* the data of the next timestep such that $\hat{y}_t^U = \hat{x}_{t+1}^U$.

3.2.3. Fine-tuning phase

In the fine-tuning phase, an RUL estimation problem is considered, hence the output of the predictive models is a point-wise estimation of the RUL such that $y_t^L = RUL_t$. The embedding z_t of the input data is extracted (see Fig. 10 and Fig. 11), the weights of the hidden pre-trained layers are frozen, then for fine-tuning a simple GRU layer ϕ followed by an output linear layer $\tilde{g}$ are used such that:

$$\begin{aligned} \hat{y}_t^L &= \tilde{g} \circ \phi(z_t) \\ &= \hat{RUL}_t \end{aligned} \tag{2}$$

where the function ϕ refers to the fine-tuning GRU layer and the function $\tilde{g}$ to the output linear layer. Note that, in the fine-tuning phase it is common to use only a linear layer for training, but the authors found that adding a GRU layer significantly improves the performance of the approach on this RUL estimation problem.

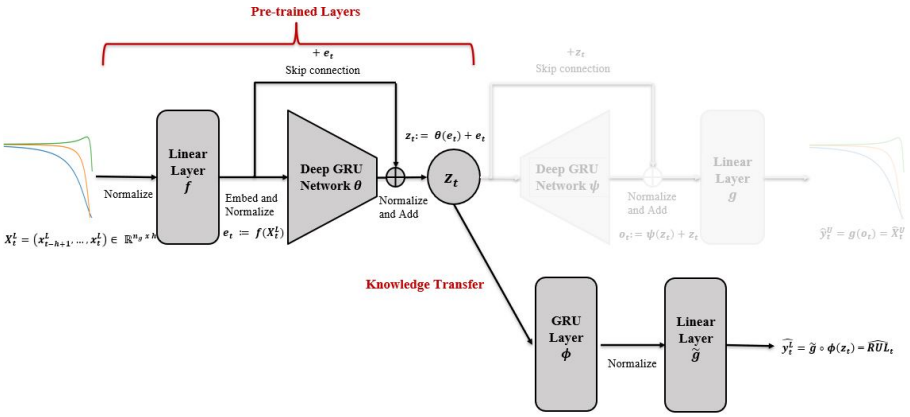


Figure 10: Flow chart of the fine-tuning phase of the Autoencoder model (AE) in the proposed Self-Supervised Learning framework.

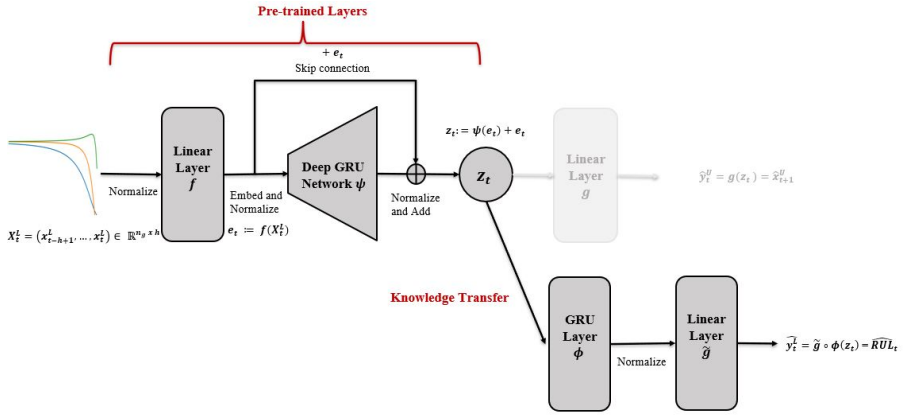


Figure 11: Flow chart of the fine-tuning phase of the Autoregressive model (AR) in the proposed Self-Supervised Learning framework.

Finally, in order to investigate the added value of the SSL approach in prognostics, the pre-trained models are compared with their non-pre-trained counterpart architecture, illustrated in Fig. 10 and Fig. 11. Note that the architectures of the pre-trained and non-pre-trained models are the same, the difference residing in the absence of pre-training on unlabelled data and the corresponding *knowledge transfer*. Also, only the non-pre-trained weights for the pre-trained models are trained (*i.e.* trainable weights of the fine-tuning model), while all the trainable parameters of the non-pre-trained models are trained. Note that the pre-trained model with an autoregressive pre-text task followed by a GRU model for fine-tuning will be referred to as the "autoregressive model" in the following for simplicity.

4. Experiments and Results

4.1. Preparation of Data

In this experiment, an Aluminum alloy 7075-T6 plate was considered, which is typical of aeronautic structures. Considering that the evolution of the changes from one cycle to another are small (see Fig. 3), it was decided to collect the data every $\Delta k = 500$ loading-unloading cycles, as in [23].

In the current paper, a training set and a testing set are generated. The training set is composed of:

1. N_U^{Train} unlabelled structures for the pre-training phase,
2. N_L^{Train} labelled structures for the fine-tuning phase.

Note that the number of structures N_U^{Train} and N_L^{Train} are varied; this will be described in the following subsections. The testing set is composed of N_L^{Test} labelled structures. It is used to evaluate the RUL estimation performance of the trained models (in fine-tuning) as a data set that was not used during training. The parameters used to generate the dataset according to the framework described in [23] are summarized in Table 2.

Parameter	Denotation	Type	Value	Unit
Elastic parameters				
Young's modulus	E	Deterministic	71.7	GPa
Poisson's ratio	ν	Deterministic	0.33	-
Strain field parameters				
Maximum stress intensity	σ_{max}	Uniform distribution	$\mathcal{U}(75, 85).10^6$	Pa
Fracture toughness	K_I	Deterministic	$19, 7.10^6$	$Pa\sqrt{m}$
Strain gauges				
Number of gauges placed	n_g	Deterministic	3	-
Position of the gauges placed	$(x_i, y_i)_{i=1,..,n_g}$	Deterministic	$(3, 14), (14, 14), (25, 14)$	mm
Angle of the gauges placed	θ	Deterministic	45	deg
Initialization parameters				
Initial crack size	a_0	Gaussian distribution	$\mathcal{N}(\mu_{a_0}, \sigma_{a_0})$	m
Mean of a_0	μ_{a_0}	Deterministic	5.10^{-4}	m
Standard deviation of a_0	σ_{a_0}	Deterministic	$2, 5.10^{-4}$	m
Paris-Erdogan's law parameters	$(m, \log C)$	Multivariate Gaussian distribution	$\mathcal{N}(\mu_m, \sigma_m, \mu_{\log C}, \sigma_{\log C}, \rho)$	-
Mean of m	μ_m	Deterministic	$3, 4$	-
Standard deviation of m	σ_m	Deterministic	$0, 25$	-
Mean of C	μ_C	Deterministic	1.10^{-10}	-
Standard deviation of C	σ_C	Deterministic	5.10^{-11}	-
Correlation coefficient of m and $\log C$	ρ	Deterministic	-0.996	-
Generated data set				
Number of unlabelled structures for training	N_U^{Train}	Deterministic	$(100, 1000, 5000, 10000)$	-
Number of labelled structures for training	N_L^{Train}	Deterministic	$(5, 10, 20, 50, 100)$	-
Number of labelled structures for testing	N_L^{Test}	Deterministic	100	-
Data collection interval	Δk	Deterministic	500	-

Table 2: Parameters for numerical study. The full model description is available in [23].

4.2. Experimental settings in pre-training phase

As the structures subjected to fatigue can be replaced before reaching failure at any time, the proposed approach has been investigated on four degradation scenarios: for pre-training, available sequences of unlabelled data are incomplete at d = 60%, 70%, 80%, and 90% of their total lifetime, where d is the ratio of the total lifetime of a sequence. To illustrate the size of the strain data sequences available, these four degradation scenarios are illustrated in Fig. 12.

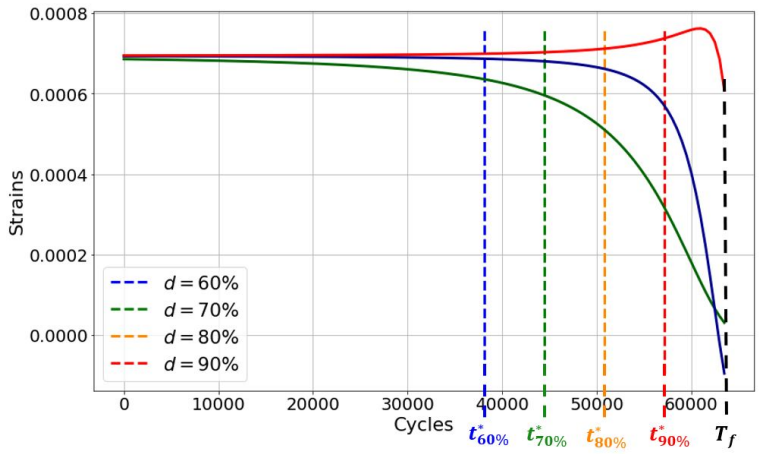


Figure 12: Four degradation scenarios depicted on the sequence of a structure. For each scenario, the available strain data correspond to the measurements from time 0 to time $t^*_d := d$ x T_f, where d is the ratio of the total lifetime of a sequence, and T_f is the time of failure.

Moreover, the number of pre-training structures, denoted N_U^{Train}, for which strain sequences were available was varied in order to investigate the effect of the amount of unlabelled data. The investigated models (autoencoder and autoregressive model) were therefore pre-trained on $N_U^{Train} = 100, 1000, 5000$, and 10000 unlabelled structures. As mentioned before, strain data are collected every 500 cycles, and a sliding window approach of $h = 30$ is used (see Section 3). Thus, as an illustration, Table 3 summarises the number of pre-training samples n_U for the autoencoder model in each degradation scenario.

In each training procedure, 95% of the dataset was used for training (in terms of the number of structures), while 5% of it was used for validation. The validation set is used for monitoring and adjusting the training phase, using the mean absolute percentage error (*MAPE*) metric. During training, the aim is to minimise the mean squared error (*MSE*) loss function L_{MSE} such that:

$$L_{MSE} = \frac{1}{n_U}\sum_{i=1}^{n_U}(y_i^U - \hat{y}_i^U)^2 \tag{3}$$

$$MAPE = \frac{1}{n_U}\sum_{i=1}^{n_U}\left|\frac{y_i^U - \hat{y}_i^U}{y_i^U}\right| * 100 \tag{4}$$

Ratio of the total lifetime d / Number of pre-training structures N_U^{Train}	d = 60%	d = 70%	d = 80%	d = 90%
$N_U^{Train} = 100$	$n_U^{Train} = 11880$	$n_U^{Train} = 14346$	$n_U^{Train} = 16819$	$n_U^{Train} = 19283$
$N_U^{Train} = 1000$	$n_U^{Train} = 114537$	$n_U^{Train} = 138451$	$n_U^{Train} = 162511$	$n_U^{Train} = 186443$
$N_U^{Train} = 5000$	$n_U^{Train} = 571515$	$n_U^{Train} = 690932$	$n_U^{Train} = 811015$	$n_U^{Train} = 930545$
$N_U^{Train} = 10000$	$n_U^{Train} = 1137959$	$n_U^{Train} = 1375948$	$n_U^{Train} = 1615261$	$n_U^{Train} = 1853470$

Table 3: Number of unlabelled pre-training samples n_U^{Train} used in this work for the autoencoder. Note that in this work, strain data are collected every $\Delta_k = 500$ cycles and a sliding window approach of $h = 30$ is used.

where n_U is the number of unlabelled samples with $\hat{y}_\cdot^U$ being the prediction and $y_\cdot^U$ the target value. Note that $y_i^U = (x_{i-h+1}^U, ..., x_i^U)$ for the AE model and $y_i^U = x_{i+1}^U$ for the AR model. The Adam optimizer [60] was used with default parameters and the learning rate was decreased incrementally. The learning rates of $10^{-2}, 10^{-3}, 10^{-4}$ were sequentially used for a predefined number of epochs, saving the model weights each time the validation loss decreases; the weights of the best model were loaded each time the learning rate was lowered. At the end of the procedure, the model was trained on the whole dataset (training and validation sets) with a lower learning rate of 10^{-5} until convergence. Calculations were performed using PyTorch's core library in Python on NVIDIA V100 GPUs, hence the batch size was chosen to be as large as possible in order to speed up calculations, here $2^{12} = 4096$ depending on the available memory of the used GPUs, and not too large in order to avoid numerical instability. The model hyperparameters were optimized using a Grid Search algorithm, listed in Table 4:

- Autoencoder model illustrated in Fig. 8: the embedding linear layer f is composed of 64 neurons, the Deep GRU Networks θ and ψ were each composed of 2 layers of GRU, 64 neurons, and a dropout of 0.1, that is to say nearly 100.000 parameters.
- Autoregressive model illustrated in Fig. 9: the embedding linear layer f is composed of 64 neurons, the Deep GRU Network ψ was composed of 4 GRU layers, 64 neurons, and a dropout of 0.1, that is to say nearly 100.000 parameters.

Hyperparameters	**Search Space**	**Autoencoder model**	**Autoregressive model**
Linear layer f - neurons	{32, 64}	64	64
Deep GRU network θ - neurons	{32, 64, 128, 256}	64	-
Deep GRU network θ - layers	{1, 2, 4, 8}	2	-
Deep GRU network θ - dropout	{0, 0.1, 0.2, 0.3}	0.1	-
Deep GRU network ψ - neurons	{32, 64, 128, 256}	64	64
Deep GRU network ψ - layers	{1, 2, 4, 8}	2	4
Deep GRU network ψ - dropout	{0, 0.1, 0.2, 0.3}	0.1	0.1

Table 4: Hyperparameters of the pre-training phase.

Note that the authors found that, given the training data, the search space considered was sufficient to obtain good results, whereas deep neural networks with a larger number of layers/neurons performed poorer with longer training (probably due to more difficult convergence).

4.3. Experimental settings in Fine-tuning phase

For fine-tuning, as illustrated in Fig. 10 and Fig. 11, the embedding z_t of the input data was extracted, the weights of the hidden layers were frozen, and a GRU model was used for the downstream task. The models are then trained on N_L available labelled structures, using a sliding window approach similar to that used in the previous pre-training phase. The fine-tuning model was composed of a single GRU layer, 32 neurons, 0.1 in dropout to regularize, and followed by an output linear layer (the hyperparameters were optimized using a Grid Search algorithm on the autoencoder pre-trained model, listed in Table 5).

Hyperparameters	**Search Space**	**Fine-tuning model**
Deep GRU network ϕ - neurons	{32, 64}	32
Deep GRU network ϕ - layers	{1, 2}	1
Deep GRU network ϕ - dropout	{0, 0.1, 0.2, 0.3}	0.1
Batch size	{32, 64}	32

Table 5: Hyperparameters of the fine-tuning phase.

The pre-trained models were then compared with their non-pre-trained "counterpart" (*i.e.* same architecture but all model weights were reset) on few shots learning. The number of available labelled training structures N_L^{Train} were varied, such that: $N_L^{Train} = 5, 10, 20, 50$ and 100 labelled structures (*i.e.* strain data of structures reaching failure at time T_f, thus for which the RUL is available for each timestep between times 0 and T_f). Calculations were performed using PyTorch's core library in Python on a machine with 62GB of RAM and an NVIDIA GeForce GTX 1080 Ti 11 GB GPU.

After training during the fine-tuning phase, the models are evaluated on the testing set composed of $N_L^{Test} = 100$ different labelled structures. For each structure, a unique RUL estimation is performed at a time t_n^*. For each structure $n \in \{1, \dots, N_L^{Test}\}$, the parameter t_n^* is randomly drawn such that $t_n^* \sim T_f^n \times \mathcal{U}([0,33; 0,9])$, where T_f^n is the time of failure for the n-th structure. This means that the test prediction for the RUL is done at a time t_n^* which is drawn uniformly between 33% and 90% of the sequence's length. Hence, the input data for the model is $X_{t_n^*}^L = (x_{t_n^*-h+1}^L, \dots, x_{t_n^*}^L) \in \mathbb{R}^{n_g \text{ x } h}$ and the output of the model is $\hat{y}_{t_n^*}^L = R\hat{U}L_{t_n^*} \in \mathbb{R}$. As the RUL estimation problem is considered as a regression problem in this paper, the aim is to minimize a mean squared error loss L_{MSE} during training, and the mean absolute percentage error (*MAPE*) metric is used to evaluate the performance of the investigated models such that:

$$L_{MSE} = \frac{1}{n_L} \sum_{i=1}^{n_L} (y_i^L - \hat{y}_i^L)^2 \tag{5}$$

$$MAPE = \frac{1}{n_L} \sum_{i=1}^{n_L} |\frac{y_i^L - \hat{y}_i^L}{y_i^L}| * 100 \tag{6}$$

where n_L is the number of labelled samples with $\hat{y}_\cdot^L$ being the RUL prediction and $y_\cdot^L$ the target RUL value.

As a limited amount of labelled data leads to epistemic uncertainty, it is difficult to make a reliable comparison. Hence, a 5-fold cross validation was used by varying the split between the training and validation set, as illustrated in Fig. 13, which gives an average *MAPE* error and its standard deviation to quantify the uncertainty when evaluated on the test set.

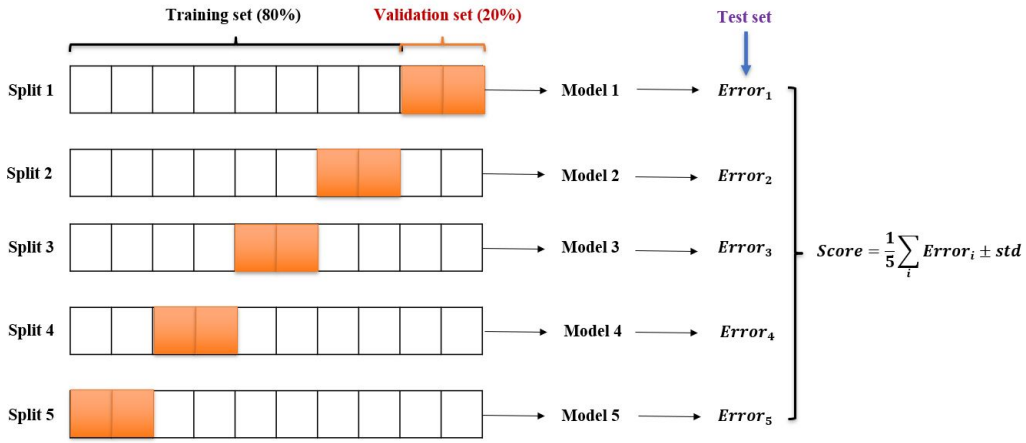


Figure 13: Flow chart of the 5-fold cross validation used in this paper.

4.4. Results

4.4.1. Pre-training analysis and comparison of pre-text tasks

First, the performance of the pre-trained Autoencoder (AE) and non-pre-trained counterpart were compared on the considered RUL estimation problem, by varying the number of unlabelled samples in pre-training. The results are presented in Table 6.

A first remark that can be drawn from the results in Table 6 is that pre-training the model is not always beneficial. For example, for an AE model pre-trained on 100 structures, the accuracy of the RUL estimation is not always better compared to the non-pre-trained model (AE). The term *negative transfer* can be used when the transfer method decreases predictive performance [61]. Moreover, considerable variability in results can be observed when models are pre-trained on very few unlabelled samples (for example when fine-tuned on 5 labelled samples), which could be due to over-fitting during pre-training. However, it can also be observed that as the number of unlabelled samples increases, the pre-training becomes more efficient and allows to have better results than a non-pre-trained model when few labelled structures are available, especially for the model pre-trained on 10000 structures. For example, the AE pre-trained models with $N_U^{Train} = 10000$ unlabelled structures and fine-tuned on $N_L^{Train} = 10$ structures has an *MAPE* of about 11-12% while the non-pre-trained model with the same $N_L^{Train} = 10$ structures has an *MAPE* of about 23%. Overall, results in Table 6 show that for the Autoencoder model, the number of unlabelled samples in pre-training matters: the more unlabelled samples, the more efficient the self supervised learning is for each of the 4

		MAPE (%)	Mean ± St.	Dev.	
Labelled structures N_L^{Train}	5	10	20	50	100
Pre-trained model (d = 60%)					
Autoencoder N_U^{Train} = 100	29.36 ± 3.81	21.16 ± 2.86	11.84 ± 4.12	1.95 ± 0.25	1.87 ± 0.36
Autoencoder N_U^{Train} = 1.000	25.95 ± 0.74	14.78 ± 3.13	5.59 ± 1.47	1.42 ± 0.17	1.31 ± 0.06
Autoencoder N_U^{Train} = 5.000	28.55 ± 0.99	**12.01 ± 2.31**	4.80 ± 0.92	**1.27 ± 0.11**	**1.08 ± 0.03**
Autoencoder N_U^{Train} = 10.000	**24.60 ± 4.30**	12.70 ± 1.76	**3.48 ± 0.93**	1.41 ± 0.04	1.13 ± 0.12
Pre-trained model (d = 70%)					
Autoencoder N_U^{Train} = 100	28.02 ± 1.96	20.99 ± 3.73	11.70 ± 3.78	1.61 ± 0.43	1.43 ± 0.25
Autoencoder N_U^{Train} = 1.000	30.76 ± 1.83	18.20 ± 3.76	6.42 ± 1.69	1.65 ± 0.36	1.29 ± 0.16
Autoencoder N_U^{Train} = 5.000	26.10 ± 0.88	18.73 ± 2.72	6.67 ± 1.30	1.58 ± 0.23	1.25 ± 0.08
Autoencoder N_U^{Train} = 10.000	**25.38 ± 5.99**	**11.85 ± 2.64**	**3.77 ± 0.61**	**1.29 ± 0.06**	**1.08 ± 0.12**
Pre-trained model (d = 80%)					
Autoencoder N_U^{Train} = 100	34.96 ± 9.59	17.72 ± 4.43	8.92 ± 3.33	1.33 ± 0.13	1.41 ± 0.20
Autoencoder N_U^{Train} = 1.000	26.46 ± 2.51	15.78 ± 3.77	4.74 ± 0.79	1.31 ± 0.13	1.10 ± 0.05
Autoencoder N_U^{Train} = 5.000	30.19 ± 6.56	17.18 ± 3.03	6.19 ± 2.07	1.60 ± 0.22	1.17 ± 0.18
Autoencoder N_U^{Train} = 10.000	**24.77 ± 4.80**	**12.06 ± 3.82**	**4.27 ± 0.72**	**1.18 ± 0.13**	**1.01 ± 0.06**
Pre-trained model (d = 90%)					
Autoencoder N_U^{Train} = 100	28.27 ± 2.47	21.56 ± 0.98	9.91 ± 3.29	1.57 ± 0.35	1.56 ± 0.16
Autoencoder N_U^{Train} = 1.000	30.27 ± 1.15	18.53 ± 5.68	6.51 ± 1.63	1.74 ± 0.43	1.13 ± 0.15
Autoencoder N_U^{Train} = 5.000	25.99 ± 1.73	17.06 ± 4.09	5.24 ± 1.60	1.43 ± 0.31	1.06 ± 0.07
Autoencoder N_U^{Train} = 10.000	**22.97 ± 5.69**	**11.04 ± 3.61**	**3.39 ± 0.67**	**1.22± 0.12**	**0.88 ± 0.09**
Non-pre-trained model					
Autoencoder architecture	27.72 ± 0.65	23.07 ± 5.94	8.12 ± 1.87	1.40 ± 0.33	**0.83 ± 0.18**

Table 6: *MAPE* mean values (in %) plus or minus its standard deviation as a function of the number of labelled structures used and of the training scenario for the autoencoder case. Best performance is represented in bold for each case.

scenarios. The autoregressive model shows similar performances, presented in Table 7.

		MAPE (%) Mean ± St. Dev.			
Labelled structures N_L^{Train}	5	10	20	50	100
Pre-trained model (d = 60%)					
Autoregressive N_U^{Train} = 100	36.15 ± 13.48	20.18 ± 5.19	12.17 ± 3.31	2.29 ± 0.22	1.70 ± 0.19
Autoregressive N_U^{Train} = 1.000	28.14 ± 3.20	16.65 ± 1.21	8.80 ± 3.13	1.48 ± 0.18	1.21 ± 0.13
Autoregressive N_U^{Train} = 5.000	26.53 ± 1.57	13.58 ± 2.14	7.25 ± 3.09	1.22 ± 0.02	1.00 ± 0.01
Autoregressive N_U^{Train} = 10.000	**22.48 ± 6.06**	**7.34 ± 0.95**	**2.63 ± 0.80**	**1.14 ± 0.04**	**1.03 ± 0.06**
Pre-trained model (d = 70%)					
Autoregressive N_U^{Train} = 100	28.95 ± 1.74	16.98 ± 2.96	11.85 ± 2.73	2.01 ± 0.20	1.51 ± 0.34
Autoregressive N_U^{Train} = 1.000	26.76 ± 2.49	16.34 ± 1.38	8.54 ± 1.89	1.53 ± 0.20	1.18 ± 0.16
Autoregressive N_U^{Train} = 5.000	25.18 ± 2.70	10.96 ± 2.46	6.79 ± 2.51	1.33 ± 0.13	1.27 ± 0.21
Autoregressive N_U^{Train} = 10.000	**24.43 ± 4.08**	**8.46 ± 1.53**	**2.42 ± 0.53**	**1.20 ± 0.06**	**1.14 ± 0.17**
Pre-trained model (d = 80%)					
Autoregressive N_U^{Train} = 100	30.47 ± 3.47	20.04 ± 2.59	10.68 ± 4.25	2.34 ± 0.85	1.48 ± 0.07
Autoregressive N_U^{Train} = 1.000	**26.10 ± 2.14**	13.66 ± 3.22	6.01 ± 1.45	1.44 ± 0.05	1.17 ± 0.03
Autoregressive N_U^{Train} = 5.000	26.46 ± 2.46	12.60 ± 1.33	6.91 ± 1.28	1.38 ± 0.11	1.09 ± 0.05
Autoregressive N_U^{Train} = 10.000	26.50 ± 2.58	**8.09 ± 3.28**	**2.39 ± 0.26**	**1.39 ± 0.20**	**1.07 ± 0.11**
Pre-trained model (d = 90%)					
Autoregressive N_U^{Train} = 100	35.76 ± 7.41	17.64 ± 2.58	8.66 ± 2.26	1.60 ± 0.17	1.33 ± 0.19
Autoregressive N_U^{Train} = 1.000	25.00 ± 3.89	17.31 ± 1.72	8.59 ± 2.79	1.45 ± 0.15	1.29 ± 0.20
Autoregressive N_U^{Train} = 5.000	23.01 ± 3.23	12.97 ± 2.91	3.15 ± 0.45	1.21 ± 0.08	1.05 ± 0.03
Autoregressive N_U^{Train} = 10.000	**22.69 ± 2.36**	**8.83 ± 1.61**	**3.39 ± 0.40**	**1.25 ± 0.07**	**0.99 ± 0.06**
Non-pre-trained model					
Autoregressive architecture	28.09 ± 1.63	21.10 ± 1.92	7.52 ± 1.59	1.15 ± 0.09	**0.79 ± 0.09**

Table 7: *MAPE* mean values (in %) plus or minus its standard deviation as a function of the number of labelled structures used and of the training scenario for the autoregressive case. Best performance is represented in bold for each case.

In order to illustrate the differences in performance between the autoencoder and the autoregressive pre-text tasks, the *MAPE* of these models for $N_U^{Train} = 10000$ structures as well as the *MAPE* of their non-pre-trained counterparts are provided in Fig. 14. For very few labelled structures ($N_L^{Train} = 5$), results illustrated in Fig. 14 do not allow to clearly distinguish between the two models due to the limited number of labelled samples, leading all models to work relatively poorly.

Nevertheless, results show that both pre-trained models clearly outperform their non-pre-trained counterpart in Few-Shots learning (more than 5 but less than 50 structures). The AR pre-trained model significantly outperforms the AE pre-trained one when fine-tuned on 10 or 20 structures, and has almost three times less estimation error than the best non-pre-trained model. These results make sense since the autoregressive task and the RUL estimation task have in common the task of predicting future outcome, and may need to capture the temporal dependencies of the input signal. However, it can also be seen that as the number of labelled samples increases, the difference between the pre-trained and non-pre-trained models is reduced (*e.g.* trained on more than 50 structures).

Given the good performance of the autoregressive model, some further variations of this concept were investigated. Hence, an extended Autoregressive model was proposed,

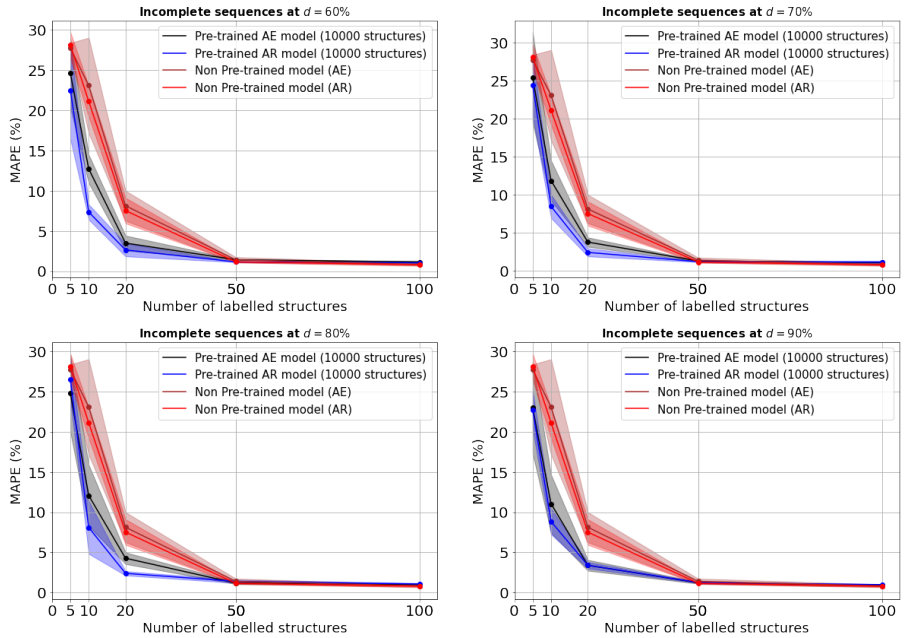


Figure 14: Comparison of pre-trained and non-pre-trained models in RUL estimation for the testing set (100 samples). The *MAPE* metric (%) is used, and here the target value to estimate is the RUL.

denoted multi-steps prediction autoregressive (MSPA) model, for which the pre-text task consists in estimating at each timestep t the data from the next timestep $t+1$ until the timestep $t+q$, such that $y_t^U = (\hat{x}_{t+1}^U, \ldots, \hat{x}_{t+q}^U)$, with $q \in \mathbf{N}^*$, as illustrated in Fig. 15.

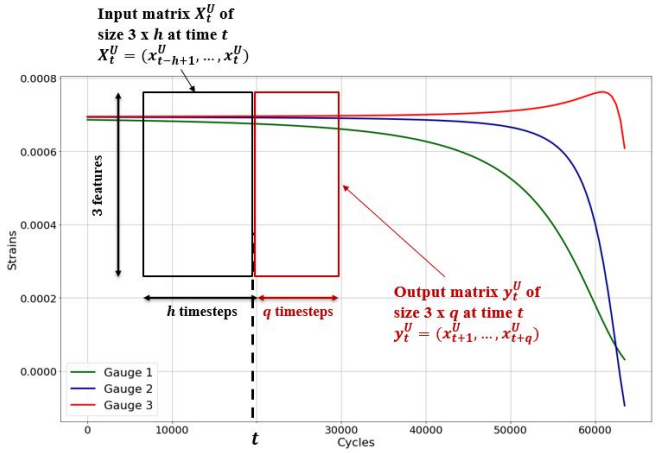


Figure 15: Illustration of the extended Autoregressive model (MSPA) pre-text task.

Results in Table 8 show that increasing the prediction time horizon in pre-training can improve the predictive performance of the fine-tuned models, when few labelled structures for training are available (N_L^{Train} = 5 or 10). For example, the MSPA model with a time horizon of q = 30 and fine-tuned on N_L^{Train} = 5 structures has an $MAPE$ of about 13% when d = 90%, while the initial autoregressive model trained under the same conditions has an $MAPE$ of about 22%. Moreover, on very few labelled training structures (N_L^{Train} = 5), the results show that the parameter d has a significant influence on the pre-training of the MSPA models: the higher d is, the more the available sequence data is close to the given failure time T_f (see Section 4.1) and the better the RUL estimation performance of the MSPA pre-trained models. This improvement of the MSPA performance compared to the AR one makes sense since when at d = 90%, predicting q = 30 timesteps means predicting the strain data until the time of failure. Being able to accurately predict until time of failure facilitates of course the downstream RUL prediction task. However, note that this performance does not hold when more labelled samples for training are available (N_L^{Train} greater than 20) by becoming worse than those of the initial autoregressive model and the non-pre-trained model, which does not allow general conclusions to be drawn. One possible explanation for this worsening is that training the DL models with values of q greater than 1 is significantly more challenging. Some of the variations seen may then be related to the pre-training phase being not yet fully converged.

Future work could seek to better control the training convergence of the pre-text task for the MSPA models. As a final remark, note that the authors also tried as outputs of the pre-text task predicting $y_t^U = \hat{x}_{t+q}^U$ only, instead of predicting the entire time-windows $(\hat{x}_{t+1}^U, \ldots, \hat{x}_{t+q}^U)$. As the results obtained after fine-tuning were similar for the two approaches, in this study only the pre-text task considering the entire time-windows has been presented and described in this paper.

		MAPE (%)	Mean ± St.	Dev.	
Labelled structures N_L^{Train}	5	10	20	50	100
Pre-trained models (d = 60%)					
Autoregressive $q = 1$	22.48 ± 6.06	8.08 ± 1.27	**2.63 ± 0.80**	**1.14 ± 0.04**	**1.03 ± 0.06**
MSPA $q = 10$	25.90 ± 4.02	8.81 ± 2.08	2.86 ± 0.80	1.22 ± 0.13	1.04 ± 0.07
MSPA $q = 20$	23.93 ± 6.54	**7.81 ± 0.56**	3.55 ± 0.67	1.34 ± 0.19	1.06 ± 0.08
MSPA $q = 30$	**18.88 ± 4.02**	8.24 ± 1.33	4.95 ± 0.80	1.57 ± 0.21	1.23 ± 0.15
Pre-trained models (d = 70%)					
Autoregressive $q = 1$	24.43 ± 4.08	8.46 ± 1.53	**2.42 ± 0.53**	**1.20 ± 0.06**	**1.14 ± 0.17**
MSPA $q = 10$	21.33 ± 3.85	8.25 ± 0.88	4.34 ± 0.90	1.43 ± 0.12	1.17 ± 0.04
MSPA $q = 20$	20.99 ± 5.43	7.95 ± 1.03	4.62 ± 0.56	1.70 ± 0.17	1.22 ± 0.24
MSPA $q = 30$	**16.56 ± 3.16**	**7.52 ± 0.99**	4.79 ± 0.87	2.68 ± 0.42	1.45 ± 0.04
Pre-trained models (d = 80%)					
Autoregressive $q = 1$	26.50 ± 2.58	8.09 ± 3.28	**2.39 ± 0.26**	**1.39 ± 0.20**	**1.07 ± 0.11**
MSPA $q = 10$	17.75 ± 4.53	8.96 ± 0.53	3.53 ± 1.03	1.47 ± 0.11	1.18 ± 0.10
MSPA $q = 20$	14.84 ± 3.22	7.03 ± 1.30	4.68 ± 0.77	2.27 ± 0.41	1.55 ± 0.17
MSPA $q = 30$	**13.32 ± 2.78**	**5.83 ± 0.53**	4.55 ± 0.98	2.88 ± 0.47	2.16 ± 0.18
Pre-trained models (d = 90%)					
Autoregressive $q = 1$	22.69 ± 2.36	8.83 ± 1.61	**3.39 ± 0.40**	**1.25 ± 0.07**	**0.99 ± 0.06**
MSPA $q = 10$	14.68 ± 3.55	8.84 ± 1.73	6.55 ± 0.85	1.44 ± 0.05	1.19 ± 0.06
MSPA $q = 20$	13.92 ± 5.16	**7.15 ± 0.81**	5.22 ± 0.59	1.85 ± 0.25	1.39 ± 0.14
MSPA $q = 30$	**13.24 ± 2.34**	7.45 ± 0.73	4.42 ± 2.04	1.48 ± 0.21	1.18 ± 0.09
Non-pre-trained models					
Autoregressive architecture	28.09 ± 1.63	21.10 ± 1.92	7.52 ± 1.59	1.15 ± 0.09	**0.79 ± 0.09**

Table 8: *MAPE* mean values (in %) plus or minus its standard deviation as a function of the number of labelled structures used and of the training scenario for the autoregressive case. Best performance is represented in bold for each case.

4.4.2. Freezing pre-trained layers during learning

In the fine-tuning phase of the previous subsection, the weights of the pre-trained layers were frozen, and only the weights of the fine-tuning model were trainable (*i.e.* GRU network for fine-tuning as illustrated in Fig. 11. Therefore, the authors also sought to investigate the effect of not freezing the pre-trained layers during the fine-tuning phase. Unfreezing the layers means that the pre-text task is basically used to find a good starting point for the training of the full network architecture. As the autoregressive model showed the best performance so far, the investigation was done on this model. In the fine-tuning phase, the model based on the autoregressive structure is composed of 125121 trainable parameters when the pre-trained layers are not frozen, against 25025 trainable parameters when they are frozen. Results in Table 9 show that the two approaches perform almost similarly, so it is difficult to determine whether it is better to freeze or not the layers in this RUL estimation problem. Note that both pre-trained models remain better in RUL estimation than their non-pre-trained counterpart, with or without frozen pre-trained layers, which confirms the benefits of the pre-training in all the cases.

		MAPE (%)	Mean ± St.	Dev.	
Labelled structures N_L^{Train}	5	10	20	50	100
Pre-trained models (d = 60%)					
Autoregressive - Freeze layers	22.48 ± 6.06	8.08 ± 1.27	**2.63 ± 0.80**	**1.14 ± 0.04**	**1.03 ± 0.06**
Autoregressive - Unfreeze layers	**21.13 ± 4.87**	**6.52 ± 0.56**	3.36 ± 0.80	**1.13 ± 0.08**	**1.05 ± 0.06**
Pre-trained models (d = 70%)					
Autoregressive - Freeze layers	**24.43 ± 4.08**	**8.46 ± 1.53**	**2.42 ± 0.53**	1.20 ± 0.06	1.14 ± 0.17
Autoregressive - Unfreeze layers	24.75 ± 3.64	8.93 ± 1.93	3.20 ± 1.18	**1.14 ± 0.12**	**1.01 ± 0.09**
Pre-trained models (d = 80%)					
Autoregressive - Freeze layers	**26.50 ± 2.58**	8.09 ± 3.28	**2.39 ± 0.26**	1.39 ± 0.20	1.07 ± 0.11
Autoregressive - Unfreeze layers	27.04 ± 4.74	**7.81 ± 1.63**	2.99 ± 0.85	**1.16 ± 0.08**	**1.02 ± 0.08**
Pre-trained models (d = 90%)					
Autoregressive - Freeze Layers	22.69 ± 2.36	8.83 ± 1.61	**3.39 ± 0.40**	1.25 ± 0.07	0.99 ± 0.06
Autoregressive - Unfreeze Layers	**22.41 ± 3.05**	**8.74 ± 2.65**	4.37 ± 1.29	**1.16 ± 0.10**	**0.85 ± 0.12**
Non-pre-trained models					
Autoregressive architecture	28.09 ± 1.63	21.10 ± 1.92	7.52 ± 1.59	1.15 ± 0.09	**0.79 ± 0.09**

Table 9: *MAPE* mean values (in %) plus or minus its standard deviation as a function of the number of labelled structures used and of the training scenario for the autoregressive case, with or without freezing pre-trained layers. Best performance is represented in bold for each case.

Nevertheless, it should be noted that freezing the weights of the pre-trained layers considerably reduces the number of trainable parameters (25025 trainable parameters when the pre-trained layers are frozen, against 125121 trainable parameters when they are not), and therefore reduces the computational complexity during training. Indeed, Fig. 16 shows that freezing the pre-trained layers speeds up the calculations considerably (1.5 to 2 times less time than other models), while both investigated learning approaches in this subsection perform almost similarly as shown in Table 9.

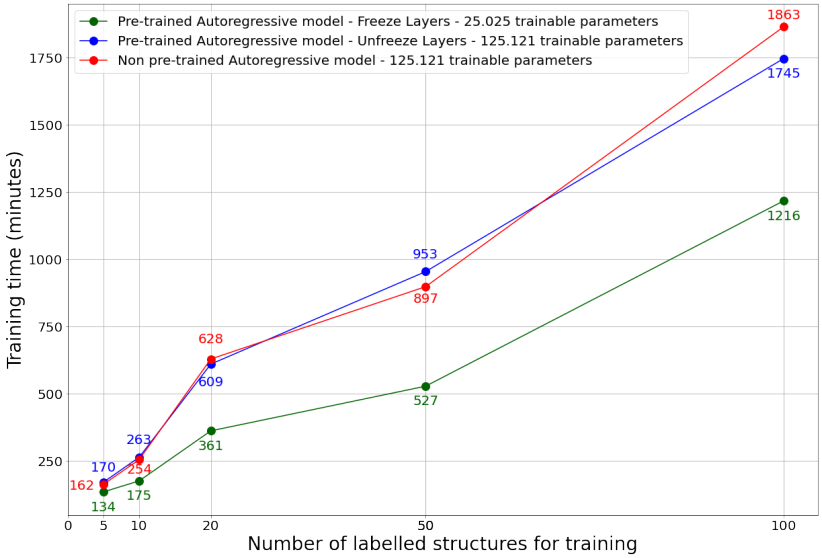


Figure 16: Mean training time during fine-tuning phase (in minutes).

5. Discussion

Based on the previous results we now summarize and discuss some of the effects observed:

- *Number of unlabelled samples for pre-training:* Results obtained confirmed that the number of pre-training samples matters. They have shown that pre-training does not always improve predictive performance when the number of pre-training samples is not sufficient, and may even decrease predictive performance (*i.e. negative transfer*). Nevertheless, these investigations indicated that as the number of unlabelled samples increases, the pre-training becomes more efficient and allows to have better results than a non-pre-trained model when few labelled structures are available. A research direction to further improve the pre-training process would be to select the available unlabelled sample, with the aim of extracting the most useful features from the data and avoiding over-fitting, in the spirit of deep active learning [62]. On the application we considered, for example, it could be interesting to implement an adaptive pre-training strategy to select the training samples and remove unnecessary samples (*e.g.* sensor data with very little variation).

- *Pre-text task:* In this work, several pre-training tasks were compared (*i.e.* input signal estimation and prediction of the future outcome of a sequence) in order to identify which one is most appropriate for the considered case study, and by extension for other engineering case studies using time series sensor data. Experiments have shown that autoregressive pre-training tasks outperform the AE model in pre-training, and capture useful representations from the sensor data (*i.e.* temporal dependencies of the input signal) for RUL estimation tasks. Moreover, results showed that increasing the prediction time horizon of autoregressive models in pre-training can improve the predictive performance, notably when few labelled structures are

available. In next steps, it would be interesting to explore other pre-text tasks (*e.g.* Contrastive learning, which aims at learning similar or dissimilar representations from source data [16]). Another interesting research direction would be to embed a Bayesian framework to the models in the pre-training phase (*e.g.* Variational Autoencoders [41]), in order to address the random nature of the data such as noise or measurement errors (*i.e.* aleatoric uncertainty). Note that the VAE model, although it has shown promising results in learning meaningful representations from raw unlabelled data [63, 64], has been studied and implemented in this work but the results were unsatisfactory. This could be due to the stochastic nature of the model during sampling: thus it requires further investigation in the future.

6. Conclusion

In this paper, a Self-Supervised Learning approach for fatigue damage prognostics problem was proposed and investigated. The approach is based on combination of a pretext and a downstream task. In the pretext task a model is trained using a large number of raw (unlabelled) sensor data with the aim of learning general representations linked to the degradation process. No RUL data is available during this pretext task, only raw sensor data (strains in our case), as the data is obtained only on structures that have not yet reached failure. Then, in a subsequent downstream task, a new model, aimed at predicting the RUL, is adjusted based on the previously pretrained model and based on a limited number of labelled RUL data obtained on structures that have reached failure. Multiple scenarios were investigated within this framework, including varying the pretext task, the models and the dataset properties.

The results obtained showed that self supervised learning is efficient in prognostics and can improve RUL estimation performances especially when only a limited amount of labelled data is available. Overall, these investigations indicate that pre-trained models are able to significantly outperform the respective non-pre-trained counterpart models in the RUL prediction task, while at the same time lowering training computational costs. Accordingly, the proposed approach can significantly reduce the need for labelled data for a given prediction accuracy, or alternatively significantly improve the prediction accuracy for the same amount of (limited) labelled data. Furthermore, the authors of this paper believe that the potential of this learning approach will benefit researchers in a variety of similar engineering fields using sensors or time series data (*e.g.* in energy [65], or water and environmental engineering [9]) and that it can be reused to overcome the lack of available labelled data.

In next steps, it would be interesting to explore other pre-text tasks (*e.g.* contrastive learning, ensemble learning, etc.) or other models (*e.g.* masked autoencoders), adaptive activation functions [66–69]). Furthermore, as uncertainty quantification remains a challenging and ubiquitous task in real-world ML applications (*e.g.* in engineering domains such as transportation engineering [70] or water and environmental applications [71]), it could be interesting to use Bayesian machine learning models in SSL (*e.g.* Deep Gaussian Processes) to quantify uncertainty in downstream prognostics tasks. Another future work perspective consists in combining strain data with other type of sensor data (*e.g.*

ultrasound mappings) in a self-supervised framework in order to further improve prediction results. Future work is also aimed at investigating how the proposed self-supervised prognostics framework behaves on an actual engineering problem involving real-world data.

Acknowledgements

This work was partially funded by the French "Occitanie Region" under the Predict project. This funding is gratefully acknowledged. This work has been partly carried out on the supercomputers PANDO (ISAE-SUPAERO, Toulouse) and Olympe (CALMIP, Toulouse, project n°21042). Authors are grateful to ISAE-SUPAERO and CALMIP for the hours allocated to this project.

Appendix A. Deep Gated Recurred Unit networks

Introduced by Cho et al. [72] Gated Recurrent Unit, or GRU, are a variant of recurrent neural networks which solves the time-delay problem existing in traditional RNNs. This approach has gained in popularity in recent years due to its relative simplicity (*i.e.* lower complexity and faster computation [27]), while the same ability to capture the mapping relationships among time series data [73]. The structure of the GRU network is shown in Fig. A.17.

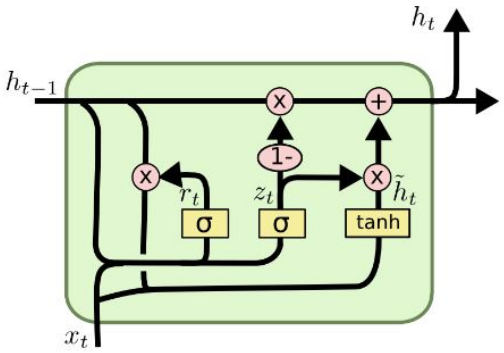


Figure A.17: GRUs architecture. Source in [74]

The formulas that govern the computation happening in a GRU network are as follow[72]:

$$\begin{aligned} z_t &= \sigma(W_z X_t + U_z h_{t-1}) \\ r_t &= \sigma(W_f X_t + U_f h_{t-1}) \\ \tilde{h}_t &= tanh(W_{\tilde{h}} X_t + U_{\tilde{h}}[h_{t-1} * r_t]) \\ h_t &= z_t * \tilde{h}_t + (1 - z_t) * h_{t-1} \end{aligned} \tag{A.1}$$

where x_t is the input sequence at time step t, h_t a hidden state, z_t the update gate, r_t the reset gate, $\tilde{h}_t$ a cell state, $\sigma(.)$ represents the sigmoid activation function and $tanh(.)$ the

hyperbolic tangent non-linear function, W and U denote the weight matrices which are learned during training. The pink circles represent pointwise operations (*e.g.* addition, multiplication). The idea behind the GRU network is that in each unit, the update gate z_t must select whether the hidden state h_t is to be updated with a new hidden state $\tilde{h}_t$; the reset gate r_t must decide whether the previous hidden state h_{t-1} is ignored. More details can be found in [72].

References

[1] X. Shao-feng, E. Yun-fei, L. Xiao-ling, L. Yu-dong, C. Yi-qiang, Development and application of prognostics and health management technology, in: Proceedings of the 20th IEEE International Symposium on the Physical and Failure Analysis of Integrated Circuits (IPFA), IEEE, 2013, pp. 3–7.

[2] D. Mao, C. Lv, J. Shi, Y. Zou, Z. Guo, Research of the military aircraft maintenance support mode based on the prognostics and health management, in: 2010 Prognostics and System Health Management Conference, IEEE, 2010, pp. 1–6.

[3] V. Atamuradov, K. Medjaher, P. Dersin, B. Lamoureux, N. Zerhouni, Prognostics and health management for maintenance practitioners-review, implementation and tools evaluation, International Journal of Prognostics and Health Management 8 (060) (2017) 1–31.

[4] Z. Wen, Y. Liu, Applications of prognostics and health management in aviation industry, in: 2011 Prognostics and System Health Managment Confernece, IEEE, 2011, pp. 1–5.

[5] K. L. Tsui, N. Chen, Q. Zhou, Y. Hai, W. Wang, Prognostics and health management: A review on data driven approaches, Mathematical Problems in Engineering 2015 (2015).

[6] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, nature 521 (7553) (2015) 436–444.

[7] S. Jadon, Covid-19 detection from scarce chest x-ray image data using few-shot deep learning approach, in: Medical Imaging 2021: Imaging Informatics for Healthcare, Research, and Applications, Vol. 11601, SPIE, 2021, pp. 161–170.

[8] T. Berthou, B. Duplessis, P. Stabat, P. Rivière, D. Marchio, Urban energy models validation in data scarcity context: Case of the electricity consumption in the french residential sector, in: Building Simulation 2019, 2019.

[9] S. Borzooei, Y. Amerlinck, S. Abolfathi, D. Panepinto, I. Nopens, E. Lorenzi, L. Meucci, M. C. Zanetti, Data scarcity in modelling and simulation of a large-scale wwtp: stop sign or a challenge, Journal of Water Process Engineering 28 (2019) 10–20.

[10] A. Gutierrez-Torre, J. L. Berral, D. Buchaca, M. Guevara, A. Soret, D. Carrera, Improving maritime traffic emission estimations on missing data with crbms, Engineering Applications of Artificial Intelligence 94 (2020) 103793.

[11] A. Nandy, C. Duan, H. J. Kulik, Audacity of huge: overcoming challenges of data scarcity and data quality for machine learning in computational materials discovery, Current Opinion in Chemical Engineering 36 (2022) 100778.

[12] A. Gorgoglione, A. Castro, C. Chreties, L. Etcheverry, Overcoming data scarcity in earth science (2020).

[13] M. A. Bansal, D. R. Sharma, D. M. Kathuria, A systematic review on data scarcity problem in deep learning: solution and applications, ACM Computing Surveys (CSUR) 54 (10s) (2022) 1–29.

[14] O. Fink, Q. Wang, M. Svensén, P. Dersin, W.-J. Lee, M. Ducoffe, Potential, challenges and future directions for deep learning in prognostics and health management applications, Engineering Applications of Artificial Intelligence 92 (2020) 103678.

[15] A. Theissler, J. Pérez-Velázquez, M. Kettelgerdes, G. Elger, Predictive maintenance enabled by machine learning: Use cases and challenges in the automotive industry, Reliability engineering & system safety 215 (2021) 107864.

[16] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, F. Makedon, A Survey on Contrastive Self-Supervised Learning, Technologies 9 (1) (Mar. 2021).

[17] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, arXiv preprint arXiv:2005.14165 (2020).

[18] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, G. E. Hinton, Big self-supervised models are strong semi-supervised learners, Advances in neural information processing systems 33 (2020) 22243–22255.

[19] Q.-X. Zhu, K.-R. Hou, Z.-S. Chen, Z.-S. Gao, Y. Xu, Y.-L. He, Novel virtual sample generation using conditional gan for developing soft sensor with small data, Engineering Applications of Artificial Intelligence 106 (2021) 104497.

[20] R. Rocchetta, Q. Gao, D. Mavroeidis, M. Petkovic, A robust model selection framework for fault detection and system health monitoring with limited failure examples: Heterogeneous data fusion and formal sensitivity bounds, Engineering Applications of Artificial Intelligence 114 (2022) 105140.

[21] T. Krokotsch, M. Knaak, C. G¨uhmann, Improving Semi-Supervised Learning for Remaining Useful Lifetime Estimation Through Self-Supervision, International Journal of Prognostics and Health Management 13 (1) (Jan. 2022). doi:10.36001/ijphm.2022.v13i1.3096.

[22] H. Guo, H. Zhu, J. Wang, V. Prahlad, W. K. Ho, T. H. Lee, Masked self-supervision for remaining useful lifetime prediction in machine tools, arXiv preprint arXiv:2207.01219 (2022).

[23] A. Akrim, C. Gogu, T. Guillebot de Nerville, P. Strähle, B. Waffa Pagou, M. Salaün, R. Vingerhoeds, A framework for generating large data sets for fatigue damage prognostic problems, in: 2022 IEEE International Conference on Prognostics and Health Management (ICPHM), IEEE, 2022, pp. 25–33. doi:10.1109/ICPHM53196.2022.9815692.

[24] J. J. M. Jimenez, S. Schwartz, R. Vingerhoeds, B. Grabot, M. Salaün, Towards multi-model approaches to predictive maintenance: A systematic literature survey on diagnostics and prognostics, Journal of Manufacturing Systems 56 (2020) 539–557.

[25] A. Voulodimos, N. Doulamis, G. Bebis, T. Stathaki, Recent developments in deep learning for engineering applications, Computational intelligence and neuroscience 2018 (2018).

[26] H. Hewamalage, C. Bergmeir, K. Bandara, Recurrent neural networks for time series forecasting: Current status and future directions, International Journal of Forecasting 37 (1) (2021) 388–427.

[27] R. Rana, Gated recurrent unit (gru) for emotion classification from noisy speech, arXiv preprint arXiv:1612.07778 (2016).

[28] M. Baptista, H. Prendinger, E. Henriques, Prognostics in aeronautics with deep recurrent neural networks, in: PHM Society European Conference, Vol. 5, 2020, pp. 11–11.

[29] R. Raina, A. Battle, H. Lee, B. Packer, A. Y. Ng, Self-taught learning: transfer learning from unlabeled data, in: Proceedings of the 24th international conference on Machine learning, 2007, pp. 759–766.

[30] L. Jing, Y. Tian, Self-supervised visual feature learning with deep neural networks: A survey, IEEE transactions on pattern analysis and machine intelligence (2020).

[31] M. Chen, A. Radford, R. Child, J. Wu, H. Jun, D. Luan, I. Sutskever, Generative pretraining from pixels, in: International Conference on Machine Learning, PMLR, 2020, pp. 1691–1703.

[32] G. Yengera, D. Mutter, J. Marescaux, N. Padoy, Less is more: Surgical phase recognition with less annotations through self-supervised pre-training of cnn-lstm networks, arXiv preprint arXiv:1805.08569 (2018).

[33] M. Endo, K. L. Poston, E. V. Sullivan, L. Fei-Fei, K. M. Pohl, E. Adeli, Gaitforemer: Self-supervised pre-training of transformers via human motion forecasting for few-shot gait impairment severity estimation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2022, pp. 130–139.

[34] J. Yu, H. Yin, X. Xia, T. Chen, J. Li, Z. Huang, Self-supervised learning for recommender systems: A survey, arXiv preprint arXiv:2203.15876 (2022).

[35] S. Shurrab, R. Duwairi, Self-supervised learning methods and applications in medical imaging analysis: A survey, PeerJ Computer Science 8 (2022) e1045.

[36] T. V. Hahn, C. K. Mechefske, Self-supervised learning for tool wear monitoring with a disentangled-variational-autoencoder, International Journal of Hydromechatronics 4 (1) (2021) 69–98.

[37] Y. Ding, J. Zhuang, P. Ding, M. Jia, Self-supervised pretraining via contrast learning for intelligent incipient fault detection of bearings, Reliability Engineering & System Safety 218 (2022) 108126.

[38] A. Saxena, K. Goebel, D. Simon, N. Eklund, Damage propagation modeling for aircraft engine run-to-failure simulation, in: 2008 international conference on prognostics and health management, IEEE, 2008, pp. 1–9.

[39] A. S. Yoon, T. Lee, Y. Lim, D. Jung, P. Kang, D. Kim, K. Park, Y. Choi, Semi-supervised learning with deep generative models for asset failure prediction, arXiv preprint arXiv:1709.00845 (2017).

[40] A. L. Ellefsen, E. Bjørlykhaug, V. Æsøy, S. Ushakov, H. Zhang, Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture, Reliability Engineering & System Safety 183 (2019) 240–251.

[41] D. P. Kingma, M. Welling, Auto-encoding variational bayes, arXiv preprint arXiv:1312.6114 (2013).

[42] G. E. Hinton, Deep belief networks, Scholarpedia 4 (5) (2009) 5947.

[43] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, R. Girshick, Masked autoencoders are scalable vision learners, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 16000–16009.

[44] P. Paris, F. Erdogan, A critical analysis of crack propagation laws, Journal of Basic Engineering 85 (4) (1963) 528–533. doi:10.1115/1.3656900.

[45] C. Cortes, V. Vapnik, Support-vector networks, Machine learning 20 (3) (1995) 273–297.

[46] T. K. Ho, Random decision forests, in: Proceedings of 3rd international conference on document analysis and recognition, Vol. 1, IEEE, 1995, pp. 278–282.

[47] C. E. Rasmussen, Gaussian processes in machine learning, in: Summer School on Machine Learning, Springer, 2003, pp. 63–71.

[48] A. Damianou, N. Lawrence, Deep gaussian processes, in: Artificial Intelligence and Statistics, 2013, pp. 207–215.

[49] M. Kandemir, Asymmetric transfer learning with deep gaussian processes, in: International Conference on Machine Learning, PMLR, 2015, pp. 730–738.

[50] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, J. Tang, Self-supervised learning: Generative or contrastive, IEEE Transactions on Knowledge and Data Engineering (2021).

[51] D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning representations by back-propagating errors, nature 323 (6088) (1986) 533–536.

[52] L. Ren, Y. Sun, J. Cui, L. Zhang, Bearing remaining useful life prediction based on deep autoencoder and deep neural networks, Journal of Manufacturing Systems 48 (2018) 71–77.

[53] J. Ma, H. Su, W.-l. Zhao, B. Liu, Predicting the remaining useful life of an aircraft engine using a stacked sparse autoencoder with multilayer self-learning, Complexity 2018 (2018).

[54] C. Sun, M. Ma, Z. Zhao, S. Tian, R. Yan, X. Chen, Deep transfer learning based on sparse autoencoder for remaining useful life prediction of tool in manufacturing, IEEE Transactions on Industrial Informatics 15 (4) (2018) 2416–2425.

[55] O. Hrinchuk, V. Khrulkov, L. Mirvakhabova, E. Orlova, I. Oseledets, Tensorized embedding layers for efficient model compression, arXiv preprint arXiv:1901.10787 (2019).

[56] J. L. Ba, J. R. Kiros, G. E. Hinton, Layer normalization, arXiv preprint arXiv:1607.06450 (2016).

[57] N. Adaloglou, Intuitive explanation of skip connections in deep learning, AI Summer (2020).

[58] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[59] B. Yue, J. Fu, J. Liang, Residual recurrent neural networks for learning sequential representations, Information 9 (3) (2018) 56.

[60] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).

[61] L. Torrey, J. Shavlik, Transfer learning, in: Handbook of research on machine learning applications and trends: algorithms, methods, and techniques, IGI Global, 2010, pp. 242–264.

[62] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, X. Wang, A survey of deep active learning, ACM computing surveys (CSUR) 54 (9) (2021) 1–40.

[63] Y. Zhu, M. R. Min, A. Kadav, H. P. Graf, S3vae: Self-supervised sequential vae for representation disentanglement and data generation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 6538–6547.

[64] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, I. Sutskever, Jukebox: A generative model for music, arXiv preprint arXiv:2005.00341 (2020).

[65] S. Jain, R. C. Shah, W. Brunette, G. Borriello, S. Roy, Exploiting mobility for energy efficient data collection in wireless sensor networks, Mobile networks and Applications 11 (3) (2006) 327–339.

[66] A. D. Jagtap, K. Kawaguchi, G. E. Karniadakis, Adaptive activation functions accelerate convergence in deep and physics-informed neural networks, Journal of Computational Physics 404 (2020) 109136.

[67] A. D. Jagtap, K. Kawaguchi, G. Em Karniadakis, Locally adaptive activation functions with slope recovery for deep and physics-informed neural networks, Proceedings of the Royal Society A 476 (2239) (2020) 20200334.

[68] A. D. Jagtap, Y. Shin, K. Kawaguchi, G. E. Karniadakis, Deep kronecker neural networks: A general framework for neural networks with adaptive activation functions, Neurocomputing 468 (2022) 165–180.

[69] A. D. Jagtap, G. E. Karniadakis, How important are activation functions in regression and classification? a survey, performance comparison, and future directions, arXiv preprint arXiv:2209.02681 (2022).

[70] E. Mazloumi, G. Rose, G. Currie, S. Moridpour, Prediction intervals to account for uncertainties in neural network predictions: Methodology and application in bus travel time prediction, Engineering Applications of Artificial Intelligence 24 (3) (2011) 534–542.

[71] B. Ghiasi, R. Noori, H. Sheikhian, A. Zeynolabedin, Y. Sun, C. Jun, M. Hamouda, S. M. Bateni, S. Abolfathi, Uncertainty quantification of granular computing-neural network model for prediction of pollutant longitudinal dispersion coefficient in aquatic streams, Scientific Reports 12 (1) (2022) 1–15.

[72] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, arXiv preprint arXiv:1406.1078 (2014).

[73] P. T. Yamak, L. Yujian, P. K. Gadosey, A comparison between arima, lstm, and gru for time series forecasting, in: Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence, 2019, pp. 49–55.

[74] C. Olah, Understanding lstm networks (2015).
URL colah.github.io/posts/2015-08-Understanding-LSTMs/

4.3 Discussion and Conclusion

The aim in this chapter was to investigate whether pre-training DL models in a self-supervised way on unlabelled sensors data can be useful for RUL estimation with only Few-Shots Learning, *i.e.* with scarce labelled data. Results show that in some conditions, where only few labelled training data is available, self-supervised pre-trained models are able to significantly outperform the non-pre-trained models in downstream RUL prediction task, while also involving less computational expense.

Thus this chapter brought some elements to answer the following research question were provided: *is it possible to learn meaningful representations from unlabeled data and use it to enhance related supervised predictive tasks on a fatigue damage prognostics problem ?* This study highlights the applicability of the proposed method to fatigue damage prognostics problems and its robustness when a limited amount of labelled data is available. The proposed learning strategy is in itself a contribution of this thesis as it can be reused in related industrial applications to overcome the lack of available labelled data. Nevertheless, the investigations also highlight the limitations of this approach concerning particularly the amount of available labelled data and the choice of the pre-text task.

CHAPTER 5

Conclusion and perspective for future work

"The value of an idea lies in the using of it."

Thomas Edison

Content

This chapter concludes the content of this manuscript by summarizing the work of the research, the limitations encountered, and more importantly the perspectives of future work that can serve as inspiration for future research.

5.1 Summary of the thesis

The research work presented in this thesis are part of the field of PHM, which aims to develop maintenance support tools, but also to improve autonomous decision-making to prevent failure. This thesis focuses only on the prognostics aspects, which has as main goal to predict the remaining time until failure (*i.e.* RUL) based on available data of the engineering structure or system.

As data availability increases due to advances in sensing technologies (*e.g.* sensor data), particular emphasis has been placed on the latest ML techniques in prognostics, notably DNNs. To summarize the lessons learnt during this research, it is necessary to come back to the research questions that initially motivated this work:

RQ1 : What are the current trends in prognostics for predictive maintenance ?

RQ2 : Which data-driven Machine Learning techniques are most commonly used to address failure prognostics, and what are their main key strengths and limitations ?

RQ3 : What are the existing related challenges and current research areas ?

These questions motivated the literature review presented in Chapter 2, that allowed to identify the current trends in prognostics for predictive maintenance. As mentioned earlier, the choice of the best suited approach to use in prognostics depends notably on the knowledge about the degradation mechanism and the available data. Among prognostics methods, the latest ML models (notably DNNs) have become a rapidly growing research direction in the field. These approches do not require an understanding of the physical principles of degradation nor expert knowledge and require only that monitoring data be available. Moreover, with the increasing availability of data for PHM, DL techniques are now the subject of considerable attention for this application, often achieving more accurate RUL predictions (e.g. recurrent and convolutional neural networks). Indeed these approaches are characterized by their ability to model a complex, multidimensional and nonlinear system without requiring a physical modeling of its behaviour. Combining two or more DL models (*i.e.* DL multi-model approaches) has also been massively investigated during the last years by PHM researches in order to overcome the complexity of prognostics tasks and increase prediction accuracy, appearing to be a promising way forward. However, their performance can sometimes be limited. Thus, the literature review pointed out the existing related challenges of the current ML algorithms in prognostics, including *data scarcity* and *uncertainty*.

In this thesis, particular emphasis has been placed on the data scarcity issue in the context of DL for two main reasons:

1. Deep Learning has become a rapidly growing research direction in the prognostics field. However, one of the major challenges for DL techniques resides in the difficulty of obtaining large amounts of labelled data for their training.

2. So far, a limited amount of existing research in the prognostics field has sought to address this data scarcity challenge, especially for fatigue damage prognostics problems.

Indeed, in many engineering domains, including in aerospace, acquiring labelled data is difficult, expensive and time consuming for an expert. Labelled data is lacking and the availability

of unlabelled data is increasing due to advancements in sensing technologies (e.g. raw sensors of structures replaced before failure). Thus, this research aims at proposing possible solutions to advance the field of data scarcity, especially in fatigue damage prognostics problems. This motivated a refinement of the research questions in this thesis:

1. How to facilitate the comparative evaluation of the latest ML models (notably DNNs) in fatigue damage prognosis problems?
2. Is it possible to learn meaningful representations from unlabeled data and use it to enhance related supervised predictive tasks on a fatigue damage prognostics problem ?

Chapter 3 aimed to provide an answer to the first refined question and sought to address the limited availability of large public datasets for fatigue damage prognostics problems. In this chapter, a framework and code for synthetically generating arbitrarily large data sets for a realistic fatigue damage prognostics problem was presented. The objective of this development is to facilitating the comparative evaluation of the latest DL algorithms in the research field, in particular in the aerospace domain. As illustration, pre-cracked Aluminum alloy 7075-T6 panels were considered, which are typical of aeronautic structures, and the applicability of some of the most commonly used Deep Learning models to address failure prognostics have been studied (including RNN, LSTM, GRU, 1D-CNN, and TCN). Also, different formulations of the RUL estimation problem were investigated, in terms of regression (point-wise estimates) or classification (estimation bounds). The results showed that for the specific application considered, the regression approach seems to be more suited for RUL estimation than the classification approach, and recurrent neural networks appear to be the best suited for regression task (especially a GRU model). Therefore, in this thesis, the RUL estimation problem was considered as a regression problem thereafter, and GRU networks were used as the basic deep learning prediction model.

Finally, with regards to the second refined question, Chapter 4 addresses the challenge of data scarcity by investigating the learning paradigm of self-supervised learning (SSL) in a fatigue damage prognostics problem, when only limited labelled data is available. The current research investigates whether pre-training DL models in a self-supervised way on unlabelled sensors data can be useful for RUL estimation with only Few-Shots Learning. As one of the main challenges for extensive investigations of the potential of SSL in PHM resides in the difficulty of having scalable open-source datasets, similar to those available in NLP or Image Processing, the framework presented in Chapter 3 is used in order to generate realistic synthetic datasets for a fatigue damage prognostics problem. Results showed that the self-supervised pre-trained models are able to significantly outperform the non-pre-trained models in downstream RUL prediction task, and with less computational expense, showing promising results in prognostics tasks when only limited labelled data is available.

5.2 Limitations

The research work presented in this thesis will contribute to advance the field of data scarcity in many engineering domains, including in aerospace. The author believes that this is a first step towards a new approach, providing an open framework and a new methodology for the prognostics problems of fatigue damage, opening many perspectives in the field, and more generally in the field of PHM.

However, it is acknowledged that research never ends and that at the end of any doctoral thesis there will always be work to be done. Indeed, the development time of an innovative project in the field is usually longer than a 3-years thesis project. In addition to the normal workload of a Ph.D. thesis, the period during which it was prepared was also characterised by the Covid-19 health crisis (March 2020). The crisis occurred at the beginning of this thesis and forced the closure of laboratories in France and around the world, which significantly affected the research schedule (notably the incapacity to use the lab's computer).

Moreover, as the subject of this thesis is innovative and exploratory by nature, the proposed perspectives may not be exhaustive. For example, the SSL approach investigated in this thesis was applied on an illustrative dataset generated without the "flaws" of real world data such as noise. It is only through more realistic case studies that the true potential of the proposed solutions can be revealed, as well as the real research gaps that should be filled.

Nevertheless, facing this limitation has been very formative and enriches the research experience. It helps the researcher to be robust, to gain creativity and skills when faced with challenges that are beyond his competence (*e.g.* VPN set-up for remote access to the lab's computer during lockdown).

5.3 Perspectives for future research

Although the developments presented in this thesis have provided some answers to the defined research questions, several aspects remain to be explored further. Indeed, a key part of any thesis manuscript is to indicate what has been missed and what is expected to be done in order to continue in the research direction. Therefore, the perspectives of future work can be oriented in three directions:

1. Perspectives related to the open-source framework proposed in the current research.
2. Perspectives related to the formulation of the RUL estimation problem for ML applications.
3. Perspectives related to the SSL approach investigated in this research.

5.3.1 Perspectives related to the open-source framework proposed in the current research

In Chapter 3, a framework and code for synthetically generating arbitrarily large data sets for a realistic fatigue damage prognostics problem. In the illustration case study, good results were expected and obtained, without the "flaws" of real world data such as noise. Thus as a key area for future work, it would be interesting to complexify the generated dataset, making it as realistic as possible, by adding the noise or varying the initialization parameters such as:

1. **Crack propagation parameters**
    - Initial crack size a_0
    - Paris-Erdogan's law parameters m and C
2. **Strain gauges**
    - Number of the gauges placed
    - Position of the gauges placed
    - Angle of the gauges placed
3. **Generated data sets**
    - Number of generated structures (training, validation, testing)
    - Data collection interval Δk

Furthermore, the author believes that this framework might be useful in real-world cases in the absence of training data or with very limited labeled data, allowing the generation of a realistic synthetic datasets useful for pre-training-based approaches (*e.g.* Transfer Learning or Self-Supervised Learning).

5.3.2 Perspectives related to the formulation of the RUL estimation problem for ML applications

In Chapter 3, different formulations of the RUL estimation problem were investigated, in terms of regression (point-wise estimates) or classification (estimation bounds). Results showed that a regression approach seems to be more suited for the RUL estimation problem than a classification approach, but increasing the number of classes seems to improve the performance of classification models rendering them competitive with regression methods. Therefore, in future work, it might be interesting to further investigate classification approaches according to following items:

1. Vary the number of classes;

2. Vary the parameter w used to control the variation of the parabolic function for class generation (*e.g.* make the intervals near the point of failure smaller);

3. Explore other DL models for classification (*e.g.* use a multi-model approach combining the potential of GRU and TCN models).

5.3.3 Perspectives related to the SSL approach investigated in this research

In Chapter 4, the SSL approach applied to a fatigue damage prognostics problem showed promising results, especially when large amounts of unlabelled data and only few labelled data are available. Nevertheless, the field of investigation is still vast, and several research directions remain to be explored such as:

1. Explore other pre-text tasks (*e.g.* contrastive learning, multi-modal learning, ensemble learning, etc.) or other models during the pre-training phase (*e.g.* variational autoencoders or VAE). Note that the VAE model, although it has shown promising results in learning meaningful representations from raw unlabelled data [232], [233], has been studied and implemented in this work but the results were unsatisfactory. This could be due to the stochastic nature of the model during sampling: thus it requires further investigation in the future;

2. Implement an adapted pre-training strategy to select the training samples, remove unnecessary samples and therefore avoid overfitting (*e.g.* remove sensors data with very little variations which may become unnecessary for training);

3. Investigate the robustness of pre-trained models to domain shift, *i.e.* evaluate the performance of the fine-tuned models on a related data set following a slightly different distribution;

4. Investigate the ability of pre-text tasks to generalize and the invariance of learned representations by exploring distinct downstream tasks (*e.g.* fault diagnosis; however, this downstream task requires a more complex generated dataset to be used);

5. Further customization of the metric used in the pre-training and fine-tuning phases, where the models were compared using the mean square error (MSE) metric for regression problems. In the pre-training phase, it would be interesting to also take into account the time or memory complexity metrics given the increasing amount of available unlabelled data in PHM; a customization of the metric used in the fine-tuning phase would be interesting according to the considered downstream task, *e.g.* for RUL estimation it would be interesting

to give more weight to the prediction of RUL values close to 0 as it is more critical to generate accurate predictions near the point of failure of the structures;

6. Improve the control of training and convergence to deal with the stochastic nature of the models and make the results more reproducible, thus facilitating their comparison (*e.g.* lower learning rate, regularization, etc.);

7. Adding a Bayesian framework to the deep learning models in the downstream tasks in order to quantify the uncertainty in prognostics (see Chapter 2), thereby facilitating the comparison of the models considered and improving the reliability in the subsequent maintenance decision making. In addition, it could be interesting to add a Bayesian framework to the models in the pre-training phase (*e.g.* Monte Carlo Dropout, variational autoencoders), in order to address the random nature of the data such as noise or measurement errors (*i.e.* aleatoric uncertainty).

5.4 Epilogue

The results obtained in this thesis have been presented in the context of the project "Predict", funded by the French "Occitanie Region". The current manuscript attempted to summarize the research experiences accumulated over the last three years specifically in the topic of the latest DL techniques applied to failure prognostics field, an interesting topic that still leaves a lot of challenges to be solved. The initial goal of this thesis was to propose possible solutions to overcome the lack of available labelled data in the engineering field. The topic covered (*i.e.* fatigue damage prognostics problems) is directly related to the needs of the aeronautical industry in order to improve the maintenance of aircrafts. Note that the suggested methods in this work can also be applied in other engineering fields related to predictive maintenance, and should be considered in future research.

List of publications

This section provides the list of conferences and publications released during this thesis.

Peer reviewed International Conferences with Proceedings

- Anass Akrim, Christian Gogu, Thomas Guillebot de Nerville, Paul Strahle, Brondon Waffa Pagou, Rob Vingerhoeds and Michel Salaün. "A Framework for Generating Large Data Sets for Fatigue Damage Prognostic Problems." In: 2022 IEEE International Conference on Prognostics and Health Management (ICPHM), Detroit (Michigan), United States. DOI: 10.1109/ICPHM53196.2022.9815692.
- Anass Akrim, Christian Gogu, Rob Vingerhoeds and Michel Salaün. "Remaining Useful Life prediction with a Deep Self-Supervised Learning Approach". Accepted for publication in the 8th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS), Oslo, Norway, 5-9th June 2022.

International Journal

- Anass Akrim, Christian Gogu, Rob Vingerhoeds and Michel Salaün. "Self-Supervised Learning for Data Scarcity in a Fatigue Damage Prognostic Problem". *preprint submitted for publication*

Bibliography

[1] E. Zio, "Prognostics and health management of industrial equipment," in *Diagnostics and prognostics of engineering systems: methods and techniques*, IGI Global, 2013, pp. 333–356.

[2] V. Atamuradov, K. Medjaher, P. Dersin, B. Lamoureux, and N. Zerhouni, "Prognostics and health management for maintenance practitioners-review, implementation and tools evaluation," *International Journal of Prognostics and Health Management*, vol. 8, no. 060, pp. 1–31, 2017.

[3] N. AFNOR, "Terminologie de la maintenance," *NF-EN*, vol. 13306, pp. X60–319, 2001.

[4] D. Mao, C. Lv, J. Shi, Y. Zou, and Z. Guo, "Research of the military aircraft maintenance support mode based on the prognostics and health management," in *2010 Prognostics and System Health Management Conference*, IEEE, 2010, pp. 1–6.

[5] Z. Wen and Y. Liu, "Applications of prognostics and health management in aviation industry," in *2011 Prognostics and System Health Managment Confernece*, IEEE, 2011, pp. 1–5.

[6] X. Shao-feng, E. Yun-fei, L. Xiao-ling, L. Yu-dong, and C. Yi-qiang, "Development and application of prognostics and health management technology," in *Proceedings of the 20th IEEE International Symposium on the Physical and Failure Analysis of Integrated Circuits (IPFA)*, IEEE, 2013, pp. 3–7.

[7] G. Gola and B. H. Nystad, "From measurement collection to remaining useful life estimation: Defining a diagnostic-prognostic frame for optimal maintenance scheduling of choke valves undergoing erosion," in *Annual Conference of the Prognostics and Health Management Society*, 2011, pp. 26–29.

[8] I. C. Monitoring, "Condition monitoring and diagnostics of machines — prognostics— part 1: General guidelines," *ISO 13381-1:2015(en)*, 2015.

[9] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *2008 international conference on prognostics and health management*, IEEE, 2008, pp. 1–9.

[10] A. K. Jardine, D. Lin, and D. Banjevic, "A review on machinery diagnostics and prognostics implementing condition-based maintenance," *Mechanical systems and signal processing*, vol. 20, no. 7, pp. 1483–1510, 2006.

[11] W. Wang, "Joint prediction of remaining useful life and failure type of train wheelsets: A multi-task learning approach," *arXiv preprint arXiv:2101.03497*, 2021.

[12] O. Bektas, J. Marshall, and J. A. Jones, "Comparison of computational prognostic methods for complex systems under dynamic regimes: A review of perspectives," *Archives of Computational Methods in Engineering*, pp. 1–13, 2019.

[13] J. J. M. Jimenez, S. Schwartz, R. Vingerhoeds, B. Grabot, and M. Salaün, "Towards multi-model approaches to predictive maintenance: A systematic literature survey on diagnostics and prognostics," *Journal of Manufacturing Systems*, vol. 56, pp. 539–557, 2020.

[14] O. Fink, Q. Wang, M. Svensén, P. Dersin, W.-J. Lee, and M. Ducoffe, "Potential, challenges and future directions for deep learning in prognostics and health management applications," *Engineering Applications of Artificial Intelligence*, vol. 92, p. 103 678, 2020.

[15] K. Javed, "A robust & reliable data-driven prognostics approach based on extreme learning machine and fuzzy clustering.," Ph.D. dissertation, 2014.

[16] K. L. Tsui, N. Chen, Q. Zhou, Y. Hai, and W. Wang, "Prognostics and health management: A review on data driven approaches," *Mathematical Problems in Engineering*, vol. 2015, 2015.

[17] X.-S. Si, W. Wang, C.-H. Hu, and D.-H. Zhou, "Remaining useful life estimation–a review on the statistical data driven approaches," *European journal of operational research*, vol. 213, no. 1, pp. 1–14, 2011.

[18] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015, ISSN: 1476-4687. DOI: 10.1038/nature14539.

[19] A. Theissler, J. Pérez-Velázquez, M. Kettelgerdes, and G. Elger, "Predictive maintenance enabled by machine learning: Use cases and challenges in the automotive industry," *Reliability engineering & system safety*, vol. 215, p. 107 864, 2021.

[20] M. Hamadache, J. H. Jung, J. Park, and B. D. Youn, "A comprehensive review of artificial intelligence-based approaches for rolling element bearing phm: Shallow and deep learning," *JMST Advances*, vol. 1, no. 1, pp. 125–151, 2019.

[21] W. Zhang, D. Yang, and H. Wang, "Data-driven methods for predictive maintenance of industrial equipment: A survey," *IEEE Systems Journal*, vol. 13, no. 3, pp. 2213–2227, 2019.

[22] Y. Wang, "Development of predictive structural maintenance strategies for aircraft using model-based prognostics," Ph.D. dissertation, 2017.

[23] J. Luo, M. Namburu, K. Pattipati, L. Qiao, M. Kawamoto, and S. Chigusa, "Model-based prognostic techniques [maintenance applications]," in *Proceedings AUTOTESTCON 2003. IEEE Systems Readiness Technology Conference.*, Ieee, 2003, pp. 330–340.

[24] A. Cubillo, S. Perinpanayagam, and M. Esperon-Miguez, "A review of physics-based models in prognostics: Application to gears and bearings of rotating machinery," *Advances in Mechanical Engineering*, vol. 8, no. 8, p. 1 687 814 016 664 660, 2016.

[25] P. Khumprom and N. Yodo, "A data-driven predictive prognostic model for lithium-ion batteries based on a deep learning algorithm," *Energies*, vol. 12, no. 4, p. 660, 2019.

[26] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, 2020.

[27] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton, "Big self-supervised models are strong semi-supervised learners," *Advances in neural information processing systems*, vol. 33, pp. 22 243–22 255, 2020.

[28] P. Nectoux, R. Gouriveau, K. Medjaher, E. Ramasso, B. Chebel-Morello, N. Zerhouni, and C. Varnier, "Pronostia: An experimental platform for bearings accelerated degradation tests.," in *IEEE International Conference on Prognostics and Health Management, PHM'12.*, IEEE Catalog Number: CPF12PHM-CDR, 2012, pp. 1–8.

[29] B. Saha and K. Goebel, "Battery data set," *NASA AMES prognostics data repository*, 2007.

[30] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[31] W. A. Little, "The existence of persistent states in the brain," in *From High-Temperature Superconductivity to Microminiature Refrigeration*, Springer, 1974, pp. 145–164.

[32] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[33] F. Camci, *Process monitoring, diagnostics and prognostics using support vector machines and hidden Markov models*. Wayne State University, 2005.

[34] H.-E. Kim, A. C. Tan, J. Mathew, E. Y. Kim, and B.-K. Choi, "Machine prognostics based on health state estimation using svm," in *Asset Condition, Information Systems and Decision Models*, Springer, 2012, pp. 169–186.

[35] X. Chen, Z. Shen, Z. He, C. Sun, and Z. Liu, "Remaining life prognostics of rolling bearing based on relative features and multivariable support vector machine," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 227, no. 12, pp. 2849–2860, 2013.

[36] T. Benkedjouh, K. Medjaher, N. Zerhouni, and S. Rechak, "Fault prognostic of bearings by using support vector data description," in *2012 IEEE Conference on Prognostics and Health Management*, IEEE, 2012, pp. 1–7.

[37] S. Dong and T. Luo, "Bearing degradation process prediction based on the pca and optimized ls-svm model," *Measurement*, vol. 46, no. 9, pp. 3143–3152, 2013.

[38] Y. Li, X. Shan, W. Zhao, and G. Wang, "A ls-svm based approach for turbine engines prognostics using sensor data,"

[39] H. Drucker, C. J. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, "Support vector regression machines," in *Advances in neural information processing systems*, 1997, pp. 155–161.

[40] M. Moustapha, J.-M. Bourinet, B. Guillaume, and B. Sudret, "Comparative study of kriging and support vector regression for structural engineering applications," *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*, vol. 4, no. 2, 2018.

[41] K.-R. Müller, A. J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik, "Predicting time series with support vector machines," in *International Conference on Artificial Neural Networks*, Springer, 1997, pp. 999–1004.

[42] C. Sun, Z. Zhang, and Z. He, "Research on bearing life prediction based on support vector machine and its application," in *Journal of Physics: Conference Series*, IOP Publishing, vol. 305, 2011, p. 012 028.

[43] M. Akuruyejo, S. Kowontan, and J. B. Ali, "A data-driven approach based health indicator for remaining useful life estimation of bearings," in *2017 18th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA)*, IEEE, 2017, pp. 284–289.

[44] T. Benkedjouh, K. Medjaher, N. Zerhouni, and S. Rechak, "Remaining useful life estimation based on nonlinear feature reduction and support vector regression," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 7, pp. 1751–1760, 2013.

[45] F. Friedrichs and C. Igel, "Evolutionary tuning of multiple svm parameters," *Neurocomputing*, vol. 64, pp. 107–117, 2005.

[46] K. Goebel, B. Saha, and A. Saxena, "A comparison of three data-driven techniques for prognostics," in *62nd meeting of the society for machinery failure prevention technology (mfpt)*, 2008, pp. 119–131.

[47] T.-N. Do and F. Poulet, "Mining very large datasets with svm and visualization.," in *ICEIS (2)*, 2005, pp. 127–141.

[48] H.-Z. Huang, H.-K. Wang, Y.-F. Li, L. Zhang, and Z. Liu, "Support vector machine based estimation of remaining useful life: Current research status and future trends," *Journal of Mechanical Science and Technology*, vol. 29, no. 1, pp. 151–163, 2015.

[49] H. Li, Z. Zhang, and Z. Liu, "Application of artificial neural networks for catalysis: A review," *Catalysts*, vol. 7, no. 10, p. 306, 2017.

[50] A. Krenker, J. Bešter, and A. Kos, "Introduction to the artificial neural networks," *Artificial Neural Networks: Methodological Advances and Biomedical Applications. InTech*, pp. 1–18, 2011.

[51] M. S. Kan, A. C. Tan, and J. Mathew, "A review on prognostic techniques for non-stationary and non-linear rotating systems," *Mechanical Systems and Signal Processing*, vol. 62, pp. 1–20, 2015.

[52] T. Van Tung and B.-S. Yang, "Machine fault diagnosis and prognosis: The state of the art," *International Journal of Fluid Machinery and Systems*, vol. 2, no. 1, pp. 61–71, 2009.

[53] J. Sikorska, M. Hodkiewicz, and L. Ma, "Prognostic modelling options for remaining useful life estimation by industry," *Mechanical systems and signal processing*, vol. 25, no. 5, pp. 1803–1836, 2011.

[54] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[55] B. Rao, P. S. Pai, and T. Nagabhushana, "Failure diagnosis and prognosis of rolling-element bearings using artificial neural networks: A critical overview," in *Journal of Physics: Conference Series*, IOP Publishing, vol. 364, 2012, p. 012 023.

[56] N. Gebraeel, M. Lawley, R. Liu, and V. Parmeshwaran, "Residual life predictions from vibration-based degradation signals: A neural network approach," *IEEE Transactions on industrial electronics*, vol. 51, no. 3, pp. 694–700, 2004.

[57] A. Riad, H. Elminir, and H. Elattar, "Evaluation of neural networks in the subject of prognostics as compared to linear regression model," *International Journal of Engineering & Technology*, vol. 10, no. 6, pp. 52–58, 2010.

[58] O. Bektas and J. A. Jones, "Narx time series model for remaining useful life estimation of gas turbine engines," in *Proceedings of Third European Conference of the Prognostics and Health Management Society*, 2016.

[59] R. Haynes, G. Joshi, and N. Bradley, "Machine learning based prognostics of fatigue crack growth in notch pre-cracked aluminum 7075-t6 rivet hole," in *Proceedings of the Annual Conference of the PHM Society*, vol. 10, 2018.

[60] P. Paris and F. Erdogan, "A critical analysis of crack propagation laws," 1963.

[61] N. Pugno, M. Ciavarella, P. Cornetti, and A. Carpinteri, "A generalized paris' law for fatigue crack growth," *Journal of the Mechanics and Physics of Solids*, vol. 54, no. 7, pp. 1333–1349, 2006.

[62] D. An, N. H. Kim, and J.-H. Choi, "Practical options for selecting data-driven or physics-based prognostics algorithms with reviews," *Reliability Engineering & System Safety*, vol. 133, pp. 223–236, 2015.

[63] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the national academy of sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.

[64] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," 2014.

[65] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*, IEEE, 2013, pp. 6645–6649.

[66] K. Yao, G. Zweig, M.-Y. Hwang, Y. Shi, and D. Yu, "Recurrent neural networks for language understanding," in *INTERSPEECH*, 2013.

[67] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[68] M. I. Jordan, "Serial order: A parallel distributed processing approach," in *Advances in psychology*, vol. 121, Elsevier, 1997, pp. 471–495.

[69] J. Yan and P. Wang, "Blade material fatigue assessment using elman neural networks," in *ASME 2007 International Mechanical Engineering Congress and Exposition*, American Society of Mechanical Engineers Digital Collection, 2007, pp. 59–64.

[70] S. E. Kramti, J. B. Ali, L. Saidi, M. Sayadi, and E. Bechhoefer, "Direct wind turbine drivetrain prognosis approach using elman neural network," in *2018 5th International Conference on Control, Decision and Information Technologies (CoDIT)*, IEEE, 2018, pp. 859–864.

[71] A. Malhi, R. Yan, and R. X. Gao, "Prognosis of defect propagation based on recurrent neural networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 3, pp. 703–711, 2011.

[72] Y. Peng, H. Wang, J. Wang, D. Liu, and X. Peng, "A modified echo state network based remaining useful life estimation approach," in *2012 IEEE Conference on Prognostics and Health Management*, IEEE, 2012, pp. 1–7.

[73] M. M. Rigamonti, P. Baraldi, E. Zio, *et al.*, "Echo state network for the remaining useful life prediction of a turbofan engine," in *annual conference of the prognostics and health management society 2015*, 2016, pp. 255–270.

[74] H. Jaeger, "Echo state network," *scholarpedia*, vol. 2, no. 9, p. 2330, 2007.

[75] J. Liu, A. Saxena, K. Goebel, B. Saha, and W. Wang, "An adaptive recurrent neural network for remaining useful life prediction of lithium-ion batteries," National Aeronautics and Space Administration Moffett Field CA Ames Research, Tech. Rep., 2010.

[76] M. Gori and A. Tesi, "On the problem of local minima in backpropagation," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 1, pp. 76–86, 1992.

[77] X. Wang, Z. Tang, H. Tamura, M. Ishii, and W. Sun, "An improved backpropagation algorithm to avoid the local minima problem," *Neurocomputing*, vol. 56, pp. 455–460, 2004.

[78] J. J. Moré, "The levenberg-marquardt algorithm: Implementation and theory," in *Numerical analysis*, Springer, 1978, pp. 105–116.

[79] L. Zhi, Y. Zhu, H. Wang, Z. Xu, and Z. Man, "A recurrent neural network for modeling crack growth of aluminium alloy," *Neural Computing and Applications*, vol. 27, no. 1, pp. 197–203, 2016.

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

[81] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[82] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, 2013, pp. 1310–1318.

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

[84] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[85] C. Olah, *Understanding lstm networks*, 2015.

[86] K. T. Nguyen, K. Medjaher, and C. Gogu, "Probabilistic deep learning methodology for uncertainty quantification of remaining useful lifetime of multi-component systems," *Reliability Engineering & System Safety*, vol. 222, p. 108 383, 2022.

[87] M. Yuan, Y. Wu, and L. Lin, "Fault diagnosis and remaining useful life estimation of aero engine using lstm neural network," in *2016 IEEE International Conference on Aircraft Utility Systems (AUS)*, IEEE, 2016, pp. 135–140.

[88] Y. Wu, M. Yuan, S. Dong, L. Lin, and Y. Liu, "Remaining useful life estimation of engineered systems using vanilla lstm neural networks," *Neurocomputing*, vol. 275, pp. 167–179, 2018.

[89] R. Ma, E. Breaz, C. Liu, H. Bai, P. Briois, and F. Gao, "Data-driven prognostics for pem fuel cell degradation by long short-term memory network," in *2018 IEEE Transportation Electrification Conference and Expo (ITEC)*, IEEE, 2018, pp. 102–107.

[90] K. Park, Y. Choi, W. J. Choi, H.-Y. Ryu, and H. Kim, "Lstm-based battery remaining useful life prediction with multi-channel charging profiles," *IEEE Access*, vol. 8, pp. 20 786–20 798, 2020.

[91] Y. Cheng, H. Zhu, J. Wu, and X. Shao, "Machine health monitoring using adaptive kernel spectral clustering and deep long short-term memory recurrent neural networks," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 2, pp. 987–997, 2018.

[92] M. Tipping, *Relevance vector machine*, US Patent 6,633,857, Oct. 2003.

[93] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[94] R. Rana, "Gated recurrent unit (gru) for emotion classification from noisy speech," *arXiv preprint arXiv:1612.07778*, 2016.

[95] P. T. Yamak, L. Yujian, and P. K. Gadosey, "A comparison between arima, lstm, and gru for time series forecasting," in *Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence*, 2019, pp. 49–55.

[96] M. Baptista, H. Prendinger, and E. Henriques, "Prognostics in aeronautics with deep recurrent neural networks," in *PHM Society European Conference*, vol. 5, 2020, pp. 11–11.

[97] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *arXiv preprint arXiv:1901.06032*, 2019.

[98] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1d convolutional neural networks and applications: A survey," *arXiv preprint arXiv:1905.03554*, 2019.

[99] A. Karpathy. "Convolutional neural networks (cnns / convnets)." (), [Online]. Available: `https://cs231n.github.io/convolutional-networks/`.

[100] N. Aloysius and M. Geetha, "A review on deep convolutional neural networks," in *2017 International Conference on Communication and Signal Processing (ICCSP)*, IEEE, 2017, pp. 0588–0592.

[101] X. Li, Q. Ding, and J.-Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Reliability Engineering & System Safety*, vol. 172, pp. 1–11, 2018.

[102] L. Xu, S. Yuan, J. Chen, and Q. Bao, "Deep learning based fatigue crack diagnosis of aircraft structures," in *Proceedings of the 7th Asia-Pacific Workshop on Structural Health Monitoring*, 2018.

[103] Z. Su, C. Zhou, M. Hong, L. Cheng, Q. Wang, and X. Qing, "Acousto-ultrasonics-based fatigue damage characterization: Linear versus nonlinear signal features," *Mechanical Systems and Signal Processing*, vol. 45, no. 1, pp. 225–239, 2014.

[104] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[105] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[106] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," *Advances in neural information processing systems*, vol. 29, 2016.

[107] C. Liu, L. Zhang, and C. Wu, "Direct remaining useful life prediction for rolling bearing using temporal convolutional networks," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, IEEE, 2019, pp. 2965–2971.

[108] M. Jouin, R. Gouriveau, D. Hissel, M.-C. Péra, and N. Zerhouni, "Particle filter-based prognostics: Review, discussion and perspectives," *Mechanical Systems and Signal Processing*, vol. 72, pp. 2–31, 2016.

[109] B. Saha, K. Goebel, and J. Christophersen, "Comparison of prognostic algorithms for estimating remaining useful life of batteries," *Transactions of the Institute of Measurement and Control*, vol. 31, no. 3-4, pp. 293–308, 2009.

[110] L. Liao and F. Köttig, "Review of hybrid prognostics approaches for remaining useful life prediction of engineered systems, and an application to battery life prediction," *IEEE Transactions on Reliability*, vol. 63, no. 1, pp. 191–207, 2014.

[111] D. Fohr, O. Mella, and I. Illina, "New paradigm in speech recognition: Deep neural networks," 2017.

[112] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[113] Y. Zhang, P. Hutchinson, N. A. Lieven, and J. Nunez-Yanez, "Remaining useful life estimation using long short-term memory neural networks and deep fusion," *IEEE Access*, vol. 8, pp. 19 033–19 045, 2020.

[114] W. Bao, X. Miao, H. Wang, G. Yang, and H. Zhang, "Remaining useful life assessment of slewing bearing based on spatial-temporal sequence," *IEEE Access*, vol. 8, pp. 9739–9750, 2020.

[115] A. L. Ellefsen, S. Ushakov, V. Æsøy, and H. Zhang, "Validation of data-driven labeling approaches using a novel deep network structure for remaining useful life predictions," *IEEE Access*, vol. 7, pp. 71 563–71 575, 2019.

[116] A. L. Ellefsen, E. Bjørlykhaug, V. Æsøy, S. Ushakov, and H. Zhang, "Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture," *Reliability Engineering & System Safety*, vol. 183, pp. 240–251, 2019.

[117] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for boltzmann machines," *Cognitive science*, vol. 9, no. 1, pp. 147–169, 1985.

[118] J. Deutsch and D. He, "Using deep learning based approaches for bearing remaining useful life prediction," in *Annual conference of the prognostics and health management society*, 2016.

[119] G. E. Hinton, "Deep belief networks," *Scholarpedia*, vol. 4, no. 5, p. 5947, 2009.

[120] C. Zhang, P. Lim, A. K. Qin, and K. C. Tan, "Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2306–2318, 2016.

[121] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[122] L. Ren, Y. Sun, J. Cui, and L. Zhang, "Bearing remaining useful life prediction based on deep autoencoder and deep neural networks," *Journal of Manufacturing Systems*, vol. 48, pp. 71–77, 2018.

[123] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 1096–1103.

[124] J. Ma, H. Su, W.-l. Zhao, and B. Liu, "Predicting the remaining useful life of an aircraft engine using a stacked sparse autoencoder with multilayer self-learning," *Complexity*, vol. 2018, 2018.

[125] A. Ng *et al.*, "Sparse autoencoder," *CS294A Lecture notes*, vol. 72, no. 2011, pp. 1–19, 2011.

[126] N. Gugulothu, V. TV, P. Malhotra, L. Vig, P. Agarwal, and G. Shroff, "Predicting remaining useful life using time series embeddings based on recurrent neural networks," *arXiv preprint arXiv:1709.01073*, 2017.

[127] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[128] P. Malhotra, V. TV, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Multi-sensor prognostics using an unsupervised health index based on lstm encoder-decoder," *arXiv preprint arXiv:1608.06154*, 2016.

[129] J. Brownlee. "A gentle introduction to lstm autoencoders." (), [Online]. Available: `https://machinelearningmastery.com/lstm-autoencoders/`.

[130] W. Yu, I. Y. Kim, and C. Mechefske, "Remaining useful life estimation using a bidirectional recurrent neural network based autoencoder scheme," *Mechanical Systems and Signal Processing*, vol. 129, pp. 764–780, 2019.

[131] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[132] B. Iwana and S. Uchida, "An empirical survey of data augmentation for time series classification with neural networks. arxiv 2020," *arXiv preprint arXiv:2007.15951*,

[133] B. Fu, W. Yuan, X. Cui, T. Yu, X. Zhao, and C. Li, "Correlation Analysis and Augmentation of Samples for a Bidirectional Gate Recurrent Unit Network for the Remaining Useful Life Prediction of Bearings," *IEEE Sensors Journal*, vol. 21, no. 6, pp. 7989–8001, Mar. 2021, Conference Name: IEEE Sensors Journal, ISSN: 1558-1748. DOI: 10.1109/JSEN.2020.3046653.

[134] S. Kim, N. H. Kim, and J.-H. Choi, "Prediction of remaining useful life by data augmentation technique based on dynamic time warping," en, *Mechanical Systems and Signal Processing*, vol. 136, p. 106 486, Feb. 2020, ISSN: 0888-3270. DOI: 10.1016/j.ymssp.2019.106486.

[135] M. Müller, "Dynamic time warping," *Information retrieval for music and motion*, pp. 69–84, 2007.

[136] P. Yuvapoositanon and P. Intachai, "A singular spectrum analysis-based synthetic dataset generation method for remaining useful life estimation of turbo fan engines,"

[137] J. B. Elsner and A. A. Tsonis, *Singular spectrum analysis: a new tool in time series analysis*. Springer Science & Business Media, 1996.

[138] N. E. Huang, Z. Shen, S. R. Long, M. C. Wu, H. H. Shih, Q. Zheng, N.-C. Yen, C. C. Tung, and H. H. Liu, "The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis," *Proceedings of the Royal Society of London. Series A: mathematical, physical and engineering sciences*, vol. 454, no. 1971, pp. 903–995, 1998.

[139] H. J. Nussbaumer, "The fast fourier transform," in *Fast Fourier Transform and Convolution Algorithms*, Springer, 1981, pp. 80–111.

[140] A. V. Oppenheim, N. S. Hamid, and A. S. Willsky, *Fourier series representation of periodic signals*. Prentice Hall, 1997.

[141] G. Ahn, H. Yun, S. Hur, and S.-Y. Lim, "A time-series data generation method to predict remaining useful life," *Processes*, vol. 9, no. 7, p. 1115, 2021.

[142] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[143] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A symbolic representation of time series, with implications for streaming algorithms," in *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, 2003, pp. 2–11.

[144] L. Torgo, R. P. Ribeiro, B. Pfahringer, and P. Branco, "Smote for regression," in *Portuguese conference on artificial intelligence*, Springer, 2013, pp. 378–389.

[145] M. Arslan, M. Guzel, M. Demirci, and S. Ozdemir, "Smote and gaussian noise based sensor data augmentation," in *2019 4th International Conference on Computer Science and Engineering (UBMK)*, IEEE, 2019, pp. 1–5.

[146] S. Behera and R. Misra, "Generative adversarial networks based remaining useful life estimation for iiot," *Computers & Electrical Engineering*, vol. 92, p. 107 195, 2021.

[147] G. D. Ranasinghe and A. K. Parlikad, "Generating real-valued failure data for prognostics under the conditions of limited data availability," in *2019 IEEE International Conference on Prognostics and Health Management (ICPHM)*, IEEE, 2019, pp. 1–8.

[148] H. Lu, V. Barzegar, V. P. Nemani, C. Hu, S. Laflamme, and A. T. Zimmerman, "GAN-LSTM Predictor for Failure Prognostics of Rolling Element Bearings," in *2021 IEEE International Conference on Prognostics and Health Management (ICPHM)*, Jun. 2021, pp. 1–8. DOI: 10.1109/ICPHM51084.2021.9486650.

[149] A. Le Guennec, S. Malinowski, and R. Tavenard, "Data augmentation for time series classification using convolutional neural networks," in *ECML/PKDD workshop on advanced analytics and learning on temporal data*, 2016.

[150] Q. Wen, L. Sun, F. Yang, X. Song, J. Gao, X. Wang, and H. Xu, "Time Series Data Augmentation for Deep Learning: A Survey," *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pp. 4653–4660, Aug. 2021, arXiv: 2002.12478. DOI: 10.24963/ijcai.2021/631.

[151] D. Erhan, A. Courville, Y. Bengio, and P. Vincent, "Why does unsupervised pre-training help deep learning?" In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, 2010, pp. 201–208.

[152] K. He, R. Girshick, and P. Dollár, "Rethinking imagenet pre-training," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4918–4927.

[153] B. Zoph, G. Ghiasi, T.-Y. Lin, Y. Cui, H. Liu, E. D. Cubuk, and Q. Le, "Rethinking pre-training and self-training," *Advances in neural information processing systems*, vol. 33, pp. 3833–3845, 2020.

[154] S. Ruder, M. E. Peters, S. Swayamdipta, and T. Wolf, "Transfer learning in natural language processing," in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: Tutorials*, 2019, pp. 15–18.

[155] M. Chen, A. Radford, R. Child, J. Wu, H. Jun, D. Luan, and I. Sutskever, "Generative pretraining from pixels," in *International Conference on Machine Learning*, PMLR, 2020, pp. 1691–1703.

[156] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[157] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A Survey on Contrastive Self-Supervised Learning," en, *Technologies*, vol. 9, no. 1, p. 2, Mar. 2021, Number: 1 Publisher: Multidisciplinary Digital Publishing Institute. DOI: 10.3390/technologies9010002.

[158] L. Torrey and J. Shavlik, "Transfer learning," in *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, IGI Global, 2010, pp. 242–264.

[159] A. Zhang, H. Wang, S. Li, Y. Cui, Z. Liu, G. Yang, and J. Hu, "Transfer learning with deep recurrent neural networks for remaining useful life estimation," *Applied Sciences*, vol. 8, no. 12, p. 2416, 2018.

[160] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A Comprehensive Survey on Transfer Learning," *arXiv:1911.02685 [cs, stat]*, Jun. 2020, arXiv: 1911.02685.

[161] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: Transfer learning from unlabeled data," in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 759–766.

[162] C. Sun, M. Ma, Z. Zhao, S. Tian, R. Yan, and X. Chen, "Deep transfer learning based on sparse autoencoder for remaining useful life prediction of tool in manufacturing," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2416–2425, 2018.

[163] W. Mao, J. He, and M. J. Zuo, "Predicting remaining useful life of rolling bearings based on deep feature representation and transfer learning," *IEEE Transactions on Instrumentation and Measurement*, 2019.

[164] Y. Fan, S. Nowaczyk, and T. Rögnvaldsson, "Transfer learning for Remaining Useful Life Prediction Based on Consensus Self-Organizing Models," *arXiv:1909.07053 [cs, stat]*, Sep. 2019, arXiv: 1909.07053.

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

[166] J. J. Montero-Jiménez and R. A. Vingerhoeds, "Enhancing operational fault diagnosis by assessing multiple operational modes," 2018.

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

[168] G. Wilson and D. J. Cook, "A survey of unsupervised deep domain adaptation," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 11, no. 5, pp. 1–46, 2020.

[169] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE transactions on neural networks*, vol. 22, no. 2, pp. 199–210, 2010.

[170] B. Jia, Y. Guan, and L. Wu, "A state of health estimation framework for lithium-ion batteries using transfer components analysis," *Energies*, vol. 12, no. 13, p. 2524, 2019.

[171] P. R. d. O. da Costa, A. Akçay, Y. Zhang, and U. Kaymak, "Remaining useful lifetime prediction via deep domain adaptation," *Reliability Engineering & System Safety*, vol. 195, p. 106 682, 2020.

[172] H. Cheng, X. Kong, Q. Wang, H. Ma, S. Yang, and G. Chen, "Deep transfer learning based on dynamic domain adaptation for remaining useful life prediction under different working conditions," *Journal of Intelligent Manufacturing*, pp. 1–27, 2021.

[173] B. Sun, J. Feng, and K. Saenko, "Correlation alignment for unsupervised domain adaptation," in *Domain Adaptation in Computer Vision Applications*, Springer, 2017, pp. 153–171.

[174] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22, no. 14, e49–e57, 2006.

[175] Z. Wu, S. Yu, X. Zhu, Y. Ji, and M. Pecht, "A weighted deep domain adaptation method for industrial fault prognostics according to prior distribution of complex working conditions," *IEEE Access*, vol. 7, pp. 139 802–139 814, 2019.

[176] S. Yu, Z. Wu, X. Zhu, and M. Pecht, "A domain adaptive convolutional lstm model for prognostic remaining useful life estimation under variant conditions," in *2019 Prognostics and System Health Management Conference (PHM-Paris)*, IEEE, 2019, pp. 130–137.

[177] J. Xu, M. Fang, W. Zhao, Y. Fan, and X. Ding, "Deep Transfer Learning Remaining Useful Life Prediction of Different Bearings," in *2021 International Joint Conference on Neural Networks (IJCNN)*, ISSN: 2161-4407, Jul. 2021, pp. 1–8. DOI: 10.1109/IJCNN52387.2021.9533676.

[178] S. Fu, Y. Zhang, L. Lin, M. Zhao, and S.-s. Zhong, "Deep residual lstm with domain-invariance for remaining useful life prediction across domains," *Reliability Engineering & System Safety*, vol. 216, p. 108 012, 2021.

[179] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The journal of machine learning research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[180] M. Ragab, Z. Chen, M. Wu, C. K. Kwoh, and X. Li, "Adversarial Transfer Learning for Machine Remaining Useful Life Prediction," in *2020 IEEE International Conference on Prognostics and Health Management (ICPHM)*, Jun. 2020, pp. 1–7. DOI: 10.1109/ICPHM49022.2020.9187053.

[181] M. Ragab, Z. Chen, M. Wu, C. S. Foo, C. K. Kwoh, R. Yan, and X. Li, "Contrastive adversarial domain adaptation for machine remaining useful life prediction," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5239–5249, 2020.

[182] B. c. Wen, M. q. Xiao, X. q. Wang, X. Zhao, J. f. Li, and X. Chen, "Data-driven remaining useful life prediction based on domain adaptation," *PeerJ Computer Science*, vol. 7, e690, Sep. 2021, ISSN: 2376-5992. DOI: 10.7717/peerj-cs.690.

[183] Y.-A. Chung and J. Glass, "Generative pre-training for speech with autoregressive predictive coding," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2020, pp. 3497–3501.

[184] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[185] A. S. Yoon, T. Lee, Y. Lim, D. Jung, P. Kang, D. Kim, K. Park, and Y. Choi, "Semi-supervised learning with deep generative models for asset failure prediction," *arXiv preprint arXiv:1709.00845*, 2017.

[186] T. Krokotsch, M. Knaak, and C. G¨uhmann, "Improving Semi-Supervised Learning for Remaining Useful Lifetime Estimation Through Self-Supervision," en, *International Journal of Prognostics and Health Management*, vol. 13, no. 1, Jan. 2022, ISSN: 2153-2648, 2153-2648. DOI: 10.36001/ijphm.2022.v13i1.3096.

[187] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[188] A. D. Kiureghian and O. Ditlevsen, "Aleatory or epistemic? Does it matter?" en, *Structural Safety*, vol. 31, no. 2, pp. 105–112, Mar. 2009, ISSN: 01674730. DOI: 10.1016/j.strusafe.2008.06.020.

[189] X. Bai, X. Wang, X. Liu, Q. Liu, J. Song, N. Sebe, and B. Kim, "Explainable deep learning for efficient and robust pattern recognition: A survey of recent developments," *Pattern Recognition*, p. 108 102, 2021.

[190] M. Oumouni, F. Schoefs, and B. Castanier, "Modeling time and spatial variability of degradation through gamma processes for structural reliability assessment," *Structural Safety*, vol. 76, pp. 162–173, 2019.

[191] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, V. Makarenkov, and S. Nahavandi, "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," en, *Information Fusion*, vol. 76, pp. 243–297, Dec. 2021, ISSN: 15662535. DOI: 10.1016/j.inffus.2021.05.008.

[192] A. P. Dempster, "A generalization of bayesian inference," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 30, no. 2, pp. 205–232, 1968.

[193] X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma, "A survey on ensemble learning," *Frontiers of Computer Science*, pp. 1–18, 2020.

[194] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[195] J. Liu, R. Seraoui, V. Vitelli, and E. Zio, "Nuclear power plant components condition monitoring by probabilistic support vector machine," *Annals of Nuclear Energy*, vol. 56, pp. 23–33, 2013.

[196] H. Wang, X. Bai, and J. Tan, "Uncertainty Quantification of Bearing Remaining Useful Life Based on Convolutional Neural Network," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, Dec. 2020, pp. 2893–2900. DOI: 10.1109/SSCI47803.2020.9308463.

[197] W. Peng, Z. Ye, and N. Chen, "Bayesian Deep-Learning-Based Health Prognostics Toward Prognostics Uncertainty," *IEEE Transactions on Industrial Electronics*, 2020. DOI: 10.1109/TIE.2019.2907440.

[198] J. Caceres, D. Gonzalez, T. Zhou, and E. Droguett, "A Probabilistic Bayesian Recurrent Neural Network for Remaining Useful Life Prognostics Considering Epistemic and Aleatory Uncertainties," *Structural Control and Health Monitoring*, vol. 28, Jun. 2021. DOI: 10.1002/stc.2811.

[199] M. Benker, L. Furtner, T. Semm, and M. F. Zaeh, "Utilizing uncertainty information in remaining useful life estimation via Bayesian neural networks and Hamiltonian Monte Carlo," en, *Journal of Manufacturing Systems*, vol. 61, pp. 799–807, Oct. 2021, ISSN: 0278-6125. DOI: 10.1016/j.jmsy.2020.11.005.

[200] G. Gao, Z. Que, and Z. Xu, "Predicting Remaining Useful Life with Uncertainty Using Recurrent Neural Process," in *2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, Dec. 2020, pp. 291–296. DOI: 10.1109/QRS-C51114.2020.00057.

[201] M. Kraus and S. Feuerriegel, "Forecasting remaining useful life: Interpretable deep learning approach via variational Bayesian inferences," en, *Decision Support Systems*, vol. 125, p. 113 100, Oct. 2019, ISSN: 0167-9236. DOI: 10.1016/j.dss.2019.113100.

[202] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[203] A. Graves, "Practical variational inference for neural networks," *Advances in neural information processing systems*, vol. 24, 2011.

[204] D. Huang, R. Bai, S. Zhao, P. Wen, J. He, S. Wang, and S. Chen, "A Hybrid Bayesian Deep Learning Model for Remaining Useful Life Prognostics and Uncertainty Quantification," in *2021 IEEE International Conference on Prognostics and Health Management (ICPHM)*, Jun. 2021, pp. 1–8. DOI: 10.1109/ICPHM51084.2021.9486527.

[205] M. Betancourt, "A conceptual introduction to hamiltonian monte carlo," *arXiv preprint arXiv:1701.02434*, 2017.

[206] E. Bingham, J. P. Chen, M. Jankowiak, F. Obermeyer, N. Pradhan, T. Karaletsos, R. Singh, P. A. Szerlip, P. Horsfall, and N. D. Goodman, "Pyro: Deep universal probabilistic programming," *J. Mach. Learn. Res.*, vol. 20, 28:1–28:6, 2019.

[207] J. Shi, J. Chen, J. Zhu, S. Sun, Y. Luo, Y. Gu, and Y. Zhou, "Zhusuan: A library for bayesian deep learning," *arXiv preprint arXiv:1709.05870*, 2017.

[208] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*, PMLR, 2016, pp. 1050–1059.

[209] M. Wei, H. Gu, M. Ye, Q. Wang, X. Xu, and C. Wu, "Remaining useful life prediction of lithium-ion batteries based on monte carlo dropout and gated recurrent unit," *Energy Reports*, vol. 7, pp. 2862–2871, 2021.

[210] W. Peng, Z.-S. Ye, and N. Chen, "Bayesian deep-learning-based health prognostics toward prognostics uncertainty," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 3, pp. 2283–2293, 2019.

[211] D. Xiao, C. Qin, J. Ge, P. Xia, Y. Huang, and C. Liu, "Self-attention-based adaptive remaining useful life prediction for igbt with monte carlo dropout," *Knowledge-Based Systems*, p. 107 902, 2021.

[212] L. Biggio, A. Wieland, M. A. Chao, I. Kastanis, and O. Fink, "Uncertainty-aware prognosis via deep gaussian process," *IEEE Access*, vol. 9, pp. 123 517–123 527, 2021.

[213] A. Damianou and N. Lawrence, "Deep gaussian processes," in *Artificial Intelligence and Statistics*, 2013, pp. 207–215.

[214] M. Jankowiak, G. Pleiss, and J. Gardner, "Deep sigma point processes," in *Conference on Uncertainty in Artificial Intelligence*, PMLR, 2020, pp. 789–798.

[215] Y. Xie, J. Zou, C. Peng, Y. Zhu, and F. Gao, "A novel pem fuel cell remaining useful life prediction method based on singular spectrum analysis and deep gaussian processes," *International Journal of Hydrogen Energy*, vol. 45, no. 55, pp. 30 942–30 956, 2020.

[216] P. Tagade, K. S. Hariharan, S. Ramachandran, A. Khandelwal, A. Naha, S. M. Kolake, and S. H. Han, "Deep gaussian process regression for lithium-ion battery health prognosis and degradation mode diagnosis," *Journal of Power Sources*, vol. 445, p. 227 281, 2020.

[217] E.-J. Wagenmakers, M. Lee, T. Lodewyckx, and G. J. Iverson, "Bayesian versus frequentist inference," in *Bayesian evaluation of informative hypotheses*, Springer, 2008, pp. 181–207.

[218] T. Vishnu, P. Malhotra, L. Vig, G. Shroff, *et al.*, "Data-driven prognostics with predictive uncertainty estimation using ensemble of deep ordinal regression models," *International Journal of Prognostics and Health Management*, vol. 10, no. 4, 2019.

[219] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[220] F. Petropoulos, R. J. Hyndman, and C. Bergmeir, "Exploring the sources of uncertainty: Why does bagging for time series forecasting work?" *European Journal of Operational Research*, vol. 268, no. 2, pp. 545–554, 2018.

[221] B. Efron, "Bootstrap methods: Another look at the jackknife," in *Breakthroughs in statistics*, Springer, 1992, pp. 569–593.

[222] Y. Liao, L. Zhang, and C. Liu, "Uncertainty Prediction of Remaining Useful Life Using Long Short-Term Memory Network Based on Bootstrap Method," Jun. 2018, pp. 1–8. DOI: 10.1109/ICPHM.2018.8448804.

[223] J. Cornelius, B. Brockner, S. H. Hong, Y. Wang, K. Pant, and J. Ball, "Estimating and Leveraging Uncertainties in Deep Learning for Remaining Useful Life Prediction in Mechanical Systems," in *2020 IEEE International Conference on Prognostics and Health Management (ICPHM)*, Jun. 2020, pp. 1–8. DOI: 10.1109/ICPHM49022.2020.9187063.

[224] D. Preuveneers, I. Tsingenopoulos, and W. Joosen, "Resource usage and performance trade-offs for machine learning models in smart environments," *Sensors*, vol. 20, no. 4, p. 1176, 2020.

[225] C. M. Bishop, "Bayesian neural networks," *Journal of the Brazilian Computer Society*, vol. 4, pp. 61–68, 1997.

[226] D. M. Hawkins, "The problem of overfitting," *Journal of chemical information and computer sciences*, vol. 44, no. 1, pp. 1–12, 2004.

[227] M. S. Ayhan and P. Berens, "Test-time data augmentation for estimation of heteroscedastic aleatoric uncertainty in deep neural networks," 2018.

[228] Y. LeCun. "Self-supervised learning: The dark matter of intelligence." (), [Online]. Available: https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/.

[229] A. Akrim, C. Gogu, T. Guillebot de Nerville, P. Strähle, B. Waffa Pagou, M. Salaün, and R. Vingerhoeds, "A framework for generating large data sets for fatigue damage prognostic problems," in *2022 IEEE International Conference on Prognostics and Health Management (ICPHM)*, 2022, pp. 25–33. DOI: 10.1109/ICPHM53196.2022.9815692.

[230] S. Rabanser, T. Januschowski, V. Flunkert, D. Salinas, and J. Gasthaus, "The effectiveness of discretization in forecasting: An empirical study on neural time series models," *arXiv preprint arXiv:2005.10111*, 2020.

[231] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *CoRR*, vol. abs/1803.01271, 2018. arXiv: 1803.01271.

[232] Y. Zhu, M. R. Min, A. Kadav, and H. P. Graf, "S3vae: Self-supervised sequential vae for representation disentanglement and data generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6538–6547.

[233] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever, "Jukebox: A generative model for music," *arXiv preprint arXiv:2005.00341*, 2020.