Doctoral theses at NTNU, 2020:178

André Listou Ellefsen

A Data-Driven Prognostics and Health Management System for Autonomous and Semi-Autonomous Ships

Doctoral thesis

André Listou Ellefsen

ISBN 978-82-326-4704-0 (printed ver.)
ISBN 978-82-326-4705-7 (electronic ver.)
ISSN 1503-8181

NTNU
Norwegian University of Science and Technology

NTNU
Norwegian University of Science and Technology
Thesis for the Degree of
Philosophiae Doctor
Faculty of Engineering
Department of Ocean Operations and Civil Engineering

Doctoral theses at NTNU, 2020:178

NTNU
Norwegian University of Science and Technology

André Listou Ellefsen

A Data-Driven Prognostics and Health Management System for Autonomous and Semi-Autonomous Ships

Thesis for the Degree of Philosophiae Doctor

Trondheim, June 2020

Norwegian University of Science and Technology
Faculty of Engineering
Department of Ocean Operations and Civil Engineering

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the Degree of Philosophiae Doctor

Faculty of Engineering
Department of Ocean Operations and Civil Engineering

ISBN 978-82-326-4704-0 (printed ver.)
ISBN 978-82-326-4705-7 (electronic ver.)
ISSN 1503-8181

Doctoral theses at NTNU, 2020:178

Printed by NTNU Grafisk senter

Abstract

Ship autonomy has been one of the most-sought research objectives at the Norwegian University of Science and Technology in Aalesund for the last three years. Through credible research, we aim to maintain our competitive position in both the global and the Norwegian maritime industry by creating autonomous ships that would operate on the surface of the water entirely by themselves. However, as research has progressed, semi-autonomous ships suitable for commercialization have seemed far more likely. Such ships would require captains, engineers, machinists, technicians, etc., to operate and monitor them, especially in demanding maritime operations, either partly onboard or from a remote control center through a satellite data link. Such ships require reliance on automated systems and belonging sensor devices. Consequently, degradation of such systems during operation poses a serious threat to both profitability and safety since there is less or no crew involvement to perform immediate maintenance operations when needed.

In this context, data-driven prognostics and health management (PHM) has emerged as a promising system solution to utilize the vast amount of sensor devices on board both autonomous and semi-autonomous ships (autoships). Such a system aims to utilize algorithms built on historical sensor measurements to provide automatic data pre-processing, detections of faults, isolation of faulty components, predictions of fault probabilities, and estimations of the progression of already-detected and classified fault-types. Through these actions the system can provide intelligent maintenance recommendations or directions when maintenance operations are needed. In other words, the system can provide decision support or automation to devise an ideal maintenance schedule that eliminates failures. Then, this schedule can be used to optimize maintenance operations for the autoship in the next appropriate port of call.

In recent years, deep neural networks (DNNs) have shown great performances to process large amounts of sensor data in the PHM domain. However, their power is strongly dependent on the accessibility of fault and failure data, but such data is rarely analyzed and collected in the maritime industry. The harsh maritime environment further complicates the accuracy of DNNs. This dissertation's primary goal is to address these issues, such that both data-driven PHM and DNNs can meet their potential for autoships.

Since both data-driven PHM systems and the utilization of DNNs are in their infancy in the maritime industry in general, the main objective of research is to develop data-driven algorithms. To achieve this, first, the fundamentals of a data-driven PHM system for autoships is proposed. Then, algorithm development for both fault diagnostics and fault prognostics is conducted through three case studies. The development of a fault-type independent fault detection algorithm for maritime components has been of particularly high priority. In addition, both smart data processing solutions and novel

DNNs to increase the reliability of fault prognostics are proposed. Complicating this task, fault prognostics have not been fully developed for any application. Furthermore, this dissertation proves the advantage of transferring knowledge obtain from benchmark data of airplane engines to the maritime environment, and more specifically, to marine diesel engines in autonomous ferries. The latter acts as the main case study for this dissertation.

Acknowledgment

The research conducted in this dissertation was carried out at the Norwegian University of Science and Technology in Aalesund within the Department of Ocean Operations and Civil Engineering (IHB). The Ph.D. position was financially supported by IHB as part of the Digital Twins For Vessel Life Cycle Service project and the Research Council of Norway, grant no. 280703.

First of all, I'm grateful for the opportunity to pursue a Ph.D. degree under the supervision of Prof. Houxiang Zhang, Prof. Vilmar Æsøy, and Prof. Sergey Ushakov. The guidance and support I received during the last three years are highly appreciated. Especially, I would like to thank my main supervisor, Prof. Houxiang Zhang, for shaping me into an independent researcher. You have influenced me both as an individual and as a scientific researcher. My confidence has gone through the roof under your guidance. Also, I would like to thank Prof. Hans Petter Hildre and Siri Schulerud for their administrative support.

I would like to thank my previous Ph.D. colleague Dr. Emil Dale Bjørlykhaug for his proposal and implementation of the genetic algorithm approach in paper II, as mentioned in Section 3.3. Furthermore, I would like to thank Emil for his valid inputs and suggestions for the implementation of the fault detection algorithm in paper III, as mentioned in Section 4.2. Also, thanks for our daily deep learning discussions and jokes during the first 18 months of my Ph.D. journey.

I would like to give a big thanks to Xu Cheng, who has become my very best foreign friend. Thanks for our daily discussions about life itself, technology, and culture. Thanks for being my guide, wallet, and translator when we visited China in August 2019. Thanks for your important data analysis in papers V and VI.

Thanks to my office mate Peihua Han for his excellent machine learning related inputs in papers VI and VII. Thanks for discovering the multi-regime normalization method, as mentioned in Section 5.1. Thanks for your crucial inputs concerning the development of the dynamic and generic threshold limits, as mentioned in Section 5.2.

Thanks to Finn Tore Holmeset for collecting the data and your expert domain knowledge used in papers V, VI, and VII. After our cooperation, I finally understand the value of human inputs for deep learning algorithms.

Thanks to my colleagues at the Mechatronics group at NTNU in Aalesund. It has been a privilege working with you. Thanks to Dr. Guoyuan Li, Pierre Major, Thiago Gabriel Monteiro, Alberto Maximiliano Crescitelli, Robert Skulstad, Lars Ivar Hatledal, and William Schmidt.

Last but not least, I give a special thanks to my beloved partner in crime, Malene Gjerde Magerholm. Thanks for understanding that my brain is working 24 hours per day 7 days a week. Also, I give a special thanks to my supporting family and friends.

Contents

Abstract i

Acknowledgment iii

List of Publications vii

List of Abbreviations ix

List of Figures xi

List of Tables xiii

1 Introduction 1
- 1.1 Background and motivation . . . 1
- 1.2 Objectives . . . 5
- 1.3 Structure of the dissertation . . . 5

2 Data-Driven PHM System for Autoships 7
- 2.1 Fundamentals of the proposed data-driven PHM system . . . 7
- 2.2 Literature review . . . 10
  - 2.2.1 Benefits and challenges . . . 11
- 2.3 Scope of work . . . 13
- 2.4 Data accumulation, limitations, and assumptions . . . 15
  - 2.4.1 Benchmark data . . . 15
  - 2.4.2 Industrial company . . . 16
  - 2.4.3 Hybrid power lab . . . 16

3 Case study: the C-MAPSS data set 21
- 3.1 Data pre-processing . . . 21
- 3.2 Fault diagnostics . . . 21
  - 3.2.1 Fault detection . . . 22
- 3.3 Fault prognostics . . . 23
  - 3.3.1 Proposed deep neural networks . . . 23
  - 3.3.2 Validation of run-to-failure targets . . . 24
  - 3.3.3 Tuning of hyper-parameters . . . 25

3.3.4 Remaining useful life predictions compared with the literature . . 26

4 Case study: industrial company 29
4.1 Data pre-processing . . . 29
4.2 Fault diagnostics . . . 30
4.2.1 The initial development of the fault detection algorithm . . . 30

5 Case study: marine diesel engines in autonomous ferries 33
5.1 Data pre-processing . . . 33
5.1.1 Feature selection . . . 33
5.1.2 Multi-regime normalization . . . 34
5.2 Fault diagnostics . . . 34
5.2.1 Dynamic and generic threshold limits . . . 35
5.2.2 Online fault detection . . . 36
5.3 Fault prognostics . . . 39
5.3.1 Introducing the SkipRnet . . . 39
5.3.2 RTF targets for supervised training . . . 40
5.3.3 Data split and data augmentation . . . 40
5.3.4 Hyper-parameters and k-fold cross-validation . . . 42
5.3.5 Remaining useful life predictions for the marine diesel engine . . . 42

6 Conclusion 45
6.1 Summary of contributions . . . 45
6.2 Summary of publications . . . 46
6.3 Important directions for future work . . . 47

References 49

Appendix

A Paper I 57

B Paper II 81

C Paper III 95

D Paper IV 105

E Paper V 119

F Paper VI 127

G Paper VII 139

List of Publications

This thesis is based on the research conducted in six journal papers and one conference paper. The seven papers are included in the appendix section of this thesis. In the following list of publications, the papers are listed chronologically by the date of initial submission, from the oldest one to the most recent. Note that paper VII has not yet been accepted for publication by the target journal.

I A. L. Ellefsen, S. Ushakov, V. Æsøy, and H. Zhang, "A Comprehensive Survey of Prognostics and Health Management based on Deep Learning for Autonomous Ships", *IEEE Transactions on Reliability*, vol. 68, no. 2, pp. 720–740, 2019.

II 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.

III A. L. Ellefsen, E. Bjørlykhaug, V. Æsøy, and H. Zhang, "An Unsupervised Reconstruction-Based Fault Detection Algorithm for Maritime Components", *IEEE Access*, vol. 7, pp. 16101–16109, 2019.

IV 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. 71563–71575, 2019.

V A. L. Ellefsen, X. Cheng, F. T. Holmeset, S. Ushakov, V. Æsøy, and H. Zhang, "Automatic Fault Detection for Marine Diesel Engine Degradation in Autonomous Ferry Crossing Operation", in *2019 IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 2195–2200, Aug 2019.

VI A. L. Ellefsen, P. Han, X. Cheng, F. T. Holmeset, V. Æsøy, and H. Zhang, "Online Fault Detection in Autonomous Ferries: Using fault-type in-dependent spectral anomaly detection", *IEEE Transactions on Instrumentation and Measurement*, pp. 1–1, 2020.

VII A. L. Ellefsen, V. Æsøy, and H. Zhang, "Real-time Fault Prognostics in Autonomous Ferries: The Advantage of Data Augmentation and Skip Connections", *Submitted to IEEE Transactions on Reliability*, pp. 1–1, 2020.

The following papers are not included in this thesis but might be considered relevant due to co-authorship and similar topics:

i X. Cheng, A. L. Ellefsen, F. T. Holmeset, G. Li, H. Zhang, and S. Chen, "A Stepwise Feature Selection Scheme for a Prognostics and Health Management System in Autonomous Ferry Crossing Operation", in *2019 IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 1877-1882, Aug 2019.

ii X. Cheng, G. Li, A. L. Ellefsen, S. Chen, H. P. Hildre, and H. Zhang, "A Novel Densely Connected Convolutional Neural Network for Sea State Estimation Using Ship Motion Data", in *IEEE Transactions on Instrumentation and Measurement*, pp. 1-1, Jan 2020.

List of Abbreviations

AS	Anomaly score
AE	Autoencoder
C-MAPSS	Commercial Modular Aero-Propulsion System Simulation
CBM	Condition-based maintenance
DL	Deep learning
DNN	Deep neural network
XAI	Explainable artificial intelligence
FD	Faulty degradation
FNN	Feed-forward neural network
GA	Genetic algorithm
GPU	Graphics processing unit
HDM	Human decision-maker
HDK	Human domain knowledge
LSTM	Long short-term memory
MSE	Mean squared error
NOP	Normal operation
NTNU	Norwegian University of Science and Technology
1D CNN	One-dimensional convolutional neural network
PwL	Piece-wise linear
PdM	Predictive maintenance
PM	Preventive maintenance
PHM	Prognostics and health management
RM	Reactive maintenance
RUL	Remaining useful life

RCC	Remote control center
RBM	Restricted Boltzmann machine
RMSE	Root mean square error
RTF	Run-to-failure
SNR	Signal-to-noise-ratio
VAE	Variational autoencoder

List of Figures

1.1 Predetermined maintenance intervals 2
1.2 PHM flowchart 3

2.1 Hannover Messe 2019 7
2.2 The proposed data-driven PHM system 8
2.3 DNNs mimic the human brain 11
2.4 Scope of work 14
2.5 A turbofan engine 15
2.6 The hybrid power lab 17
2.7 The engine load profiles 18

3.1 DNN structures proposed for the C-MAPSS data set 23
3.2 Comparison of different RTF targets 24

4.1 The sliding window operation 31

5.1 Online detection of the air filter and turbo faults in the marine diesel engine 38
5.2 Online detection of the cooling fault in the marine diesel engine 38
5.3 The SkipRnet 40
5.4 Data augmentation for RTF time-series data 41
5.5 RUL performance evaluations on the test set for the marine diesel engine 43
5.6 RUL prediction results on the test set for the marine diesel engine 43

List of Tables

2.1 Traditional PHM approaches 10
2.2 The C-MAPSS data set 15
2.3 Real-life RTF data collected from a maritime component 16
2.4 Data sets collected from the hybrid power lab 19

3.1 Fault detection results of subset FD001 in the C-MAPSS data set 22
3.2 Recent results on the C-MAPSS data set 27

4.1 Predicted fault time steps on industrial company data 32
4.2 Accuracy evaluation with 100% SNR on industrial company data 32
4.3 Accuracy evaluation with 90% SNR on industrial company data 32
4.4 Accuracy evaluation with 80% SNR on industrial company data 32
4.5 Accuracy evaluation with 70% SNR on industrial company data 32

5.1 Feature selection for the marine diesel engine 34
5.2 Upper and lower threshold values 36
5.3 Validation of predicted fault time steps on marine diesel engine data . . . 37
5.4 Average accuracy evaluation on marine diesel engine data 37
5.5 Final test of the predicted fault time step on marine diesel engine data . 37
5.6 Data split to do fault prognostics for the marine diesel engine 41

1 Introduction

Today, ship autonomy is one of the most-sought research objectives at the Norwegian University of Science and Technology (NTNU) in Aalesund. This dissertation focuses mainly on how to ensure operational availability and safety of critical components associated with autonomous and semi-autonomous ships in a safe, efficient, and cost-beneficial manner.

1.1 Background and motivation

Only six years ago, most people considered autonomous and semi-autonomous ships as a futuristic fantasy [1]. Today, however, this perception has changed drastically as enthusiasm for high degrees of ship autonomy is flourishing among researchers and industry experts in the maritime industry, encompassing both autonomous and semi-autonomous ships. The former would perform all kinds of maritime operations entirely by themselves; the latter would require captains, engineers, machinists, technicians, etc., to operate and monitor them, especially in demanding maritime operations, either partly onboard or from a remote control center (RCC) through a satellite data link [2, 3]. Realistically, semi-autonomous ships are expected to be in commercial use at first, and then develop higher and higher degrees of autonomy as research progresses.

Several projects, including this dissertation, are underway to develop autonomous and semi-autonomous ships (autoships). The industry, as well as academia, anticipate that such vessels will improve both safety and profitability [4]. Autoships demand the use of highly automated systems and belonging sensor devices. Incipient faults and related failures of such systems during operation could lead to disaster since there are few people or no one on board to perform immediate maintenance actions when needed. Therefore, autoships need to transfer real-time operational sensor data to an RCC to analyze previous, current, and future health conditions of critical components. The resulting analysis can then be used to schedule maintenance operations at the next port of call [5]. Today, satellite communication firms, such as Inmarsat, can provide real-time data transmission across the world's oceans [6].

By contrast, maintenance operations on conventional ships today follow either a reactive maintenance (RM) or preventive maintenance (PM) approach [7]. RM is defined as post-failure repair that introduces high risks of downtime, while PM involves predetermined maintenance intervals [8]. NTNU's research vessel R/V Gunnerus provides an example of how PM is used in practice. This vessel has three marine diesel engines in total, where each of the engines has an independent hour counter. As seen in Figure 1.1, the hour counter for one of the engines is 13,075, while the next service is scheduled at 13,200 hours. Then, consecutive services will be performed at both 13,250 and 14,000 running hours. These time-based maintenance intervals are static and purely based on

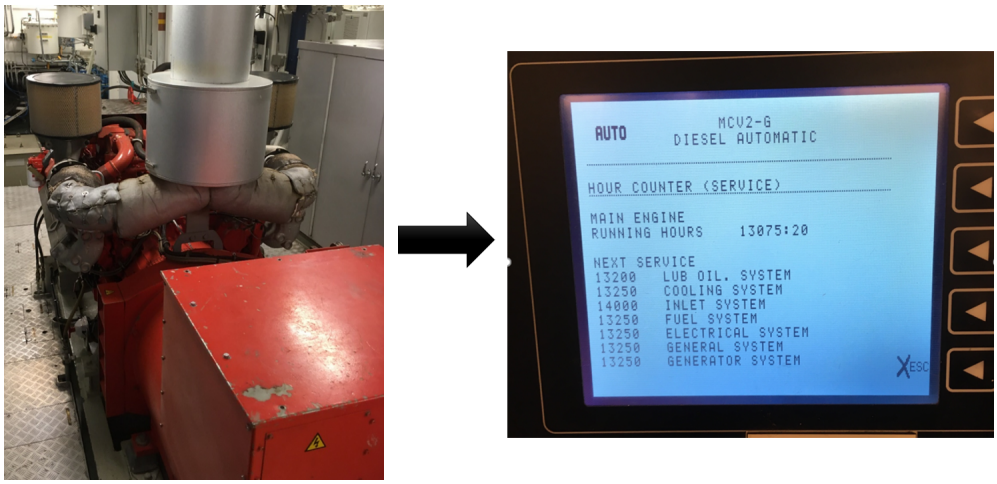


Figure 1.1: One out of three marine diesel engines onboard R/V Gunnerus and its predetermined maintenance intervals.

the experience of either the engine manufacturer or the shipowner. However, engine operations differ on different ships due to unpredictable environmental conditions. This leads to faults and failures occurring randomly [9]. Such kinds of faults and failures are not detected in the current PM system. Ergo, R/V Gunnerus relies heavily on onboard maintenance personnel.

For autoships, RM would create large and unnecessary costs due to random and unplanned downtime. On the other hand, the predetermined maintenance intervals utilized in PM could be scheduled around planned ports of call. This would, of course, provide high reliability, but involve excessive and costly inspections and maintenance actions of completely functional components. Additionally, PM lacks the ability to detect random faults and failures. Thus, the need for a more intelligent and predictive maintenance (PdM) approach is clear. Such a system could automatically alter the maintenance intervals depending on the various conditions in which the marine diesel engine has operated. In this context, data-driven prognostics and health management (PHM) has emerged as a promising solution to utilize the vast amount of sensor devices onboard autoships. As a matter of fact, the U.S. Department of Defense [10], the aerospace industry [11], and the aviation industry [12] integrated PHM with success for over ten years ago.

A data-driven PHM system is considered to be the area of research with the greatest potential to manage maintenance operations for zero-downtime performance of autoships [2, 5, 13, 14]. Such a system goes far beyond both RM and PM and strives to decrease and ultimately eliminate inspections and predetermined maintenance intervals. This will be achieved through the utilization of algorithms built on sensor measurements. As seen in Figure 1.2, PHM is defined by four main actions: data accumulation and pre-processing, fault diagnostics, fault prognostics, and decision support or automation [15, 16]. The first step collects and structures the raw data into valid input

data for the next step. Then, fault diagnostics detect faults, isolate faulty components, and classify different fault-types. The information obtained from fault diagnostics is then used as input for fault prognostics which is designated to predict the progression of already detected and classified faults-types [17]. In other words, fault prognostics estimate the available time before a faulty component will suffer from operational failure. Such estimations are normally referred to as the remaining useful life (RUL) and used to provide decision support or automation to devise an ideal maintenance schedule that eliminates failures. To conduct the four essential actions of a data-driven PHM system, autoships need to transfer real-time operational sensor data, in the format of time-series data, to an RCC. Today, deep learning (DL) algorithms are considered the ideal candidate to process large amounts of time-series data with high accuracy [18].

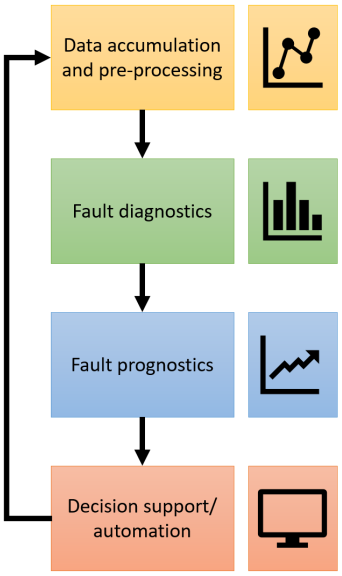


Figure 1.2: PHM flowchart, inspired by [15].

During the last three years, several DL algorithms, in terms of deep neural networks (DNNs), have been proposed in the PHM domain for both fault diagnostics [19, 20, 21] and fault prognostics [22, 23] purposes. DNNs include several layers of non-linear processing stages [24]. Consequently, they are capable of learning statistical patterns in time-series data subjected to high dimensionality and various complexities [25]. This means that DNNs are extremely powerful, but only if sufficient historical run-to-failure (RTF) time-series data is accessible in the training phase. The great potential of both data-driven PHM and DL prompts the first two research questions of this dissertation:

- ***Is a data-driven PHM system based on DL suitable for autoships?***
- ***Which DNNs are applicable?***

To address these questions, it is first necessary to investigate how scholars have applied PHM based on DL in other domains. It is also highly beneficial to investigate which DNNs have been used in each action of a data-driven PHM system. At the same time, successes achieved in other domains does not necessarily mean success in the maritime domain. Maritime operations involve a higher degree of complexity than most land-based operations, as harsh and unpredictable environmental conditions affect how critical systems, components, and sub-components are operated. The resulting uncertainty creates several challenges for successful implementation of a data-driven PHM system.

The marine diesel engine is considered the most critical component on board ships since it has an important role in both propulsion and power generation [26]. When operated in the maritime environment, however, the sensor measurements of the engine are highly connected to the operational loads. Thus, the degradation phenomena cannot be presented directly for DNNs. Additionally, there is a common lack of fault labels and

RTF data in the maritime industry [27]. This is a barrier given that state-of-the art DNNs for fault diagnostics purposes are trained in a supervised manner [28, 29]. Thus, the third and the fourth research questions are defined as follows:

- ***How to automatically detect faults associated with the marine diesel engine?***
- ***What, other than supervised learning, can be used as the learning framework?***

To address the third research question, a strong and valid case study has to be created to do significant research on the degradation phenomena of the marine diesel engine. Additionally, the nature of degradation of typical fault-types might be different from one another. Hence, both data pre-processing and the development of a fault-type independent fault detection algorithm is of high importance. Investigating the fourth research question necessitates the use of semi-supervised or unsupervised learning procedures. In the application of fault detection, semi-supervised learning only uses normal operation (NOP) data for training, while unsupervised learning has no previous knowledge of the input data where only intrinsic properties are used [30]. In autoships, the vast numbers of installed sensors can be utilized to accumulate NOP data to use a semi-supervised learning framework.

If you feed DNNs more data they get better and better [31]. Therefore, researchers typically use largely, publicly accessible benchmark data sets to train and validate their proposed DNNs for fault prognostics [22, 23, 32]. The Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) data set, which consists of numerous simulated RTF data sets depicting the operation of aircraft gas turbine engines, is acknowledged as the benchmark data set within the PHM research area [33]. One of many strengths of DNNs is their generalization power. Thus, the knowledge learned from the C-MAPSS data set can be transferred to other domains, such as, the maritime industry. Nevertheless, the fact that large databases of historical RTF data are nonexistent in the maritime domain represents a problem. Real-life RTF data is time-consuming to acquire. Besides, fault prognostics of real-world systems remain today in its infancy [34]. Due to the large uncertainties that remain in fault prognostics, researchers have called prognostics “the Achilles’ heel" of PHM [35]. Consequently, the fifth and the sixth research questions are as follows:

- ***How can significant RTF data be constructed based on small amounts of already-collected RTF data?***
- ***How can the reliability of DNNs constructed for fault prognostics be improved?***

To address the fifth and sixth research questions, several techniques can be adopted from the computer vision area of DL. Techniques such as data augmentation [36] can be used to create more RTF data, and skip connections [37] have the potential to increase the generalization power of DNNs constructed for fault prognostics. An initial unsupervised pre-training stage to extract abstract degradation related features has also shown improved generalization power [22]. High generalization power towards new field data is

extremely important if DNNs are to be employed in future data-driven PHM systems for autoships to provide real-time and reliable RUL predictions.

1.2 Objectives

In seeking to answer all six research questions, this dissertation seeks to obtain the following research objective:

- ✓ **RO1: Propose a data-driven PHM system for autoships.**

However, as the utilization of data-driven PHM systems is still in its infancy in the maritime industry, it is extremely important to enable knowledge transfer from other domains. Therefore, a comprehensive literature survey of PHM based on DL for autoships has to be conducted. The main purpose is to support creativity and provide inspiration for the maritime industry. The second research objective arises from the first two research questions of this dissertation and is as follows:

- ✓ **RO2: Conduct a comprehensive literature survey of PHM based on DL for autoships.**

Fault diagnostics is the first step of intelligent algorithms to consider in a data-driven PHM system and should incorporate a fault detection algorithm suitable for the maritime environment. Hence, the third research objective arises from research questions three and four:

- ✓ **RO3: Develop a fault-type independent fault detection algorithm for maritime components.**

Fault prognostics is the second step of intelligent algorithms. Fault prognostics is less mature than fault diagnostics in every domain of application. Thus, the fourth research objective arises from research questions five and six:

- ✓ **RO4: Propose techniques and DNNs to increase the reliability of fault prognostics.**

1.3 Structure of the dissertation

The rest of this dissertation is organized as follows. Chapter 2 introduces the theoretical foundation of the proposed data-driven PHM system for autoships. This chapter also discusses benefits and challenges, presents the scope of work, and explains the data collection processes, including assumptions and limitations, for the following case studies. Chapter 3 presents the research results and discusses the first case study, which involves the C-MAPSS data set. This chapter is based on papers II and IV. The research findings and discussion of the second case study are put forward in Chapter 4. This chapter uses RTF data collected from an industrial company and it is based on paper III. The third and final case study is presented in Chapter 5. This chapter uses RTF data collected from a marine diesel engine and it is based on papers V, VI, and VII. Chapter 6 concludes the dissertation, summarizes the contributions, and indicates objectives for future work. All case studies presented here use Microsoft Windows 10, Java 8, deeplearning4j (DL4J) [38] as the DL library, and NVIDIA GeForce GTX 1060 6 GB as the graphics processing unit (GPU).

2

Data-Driven PHM System for Autoships

In four sections, this chapter describes the proposed data-driven PHM system for autoships. Section 2.1 introduces the fundamentals of the proposed system. Section 2.2 summarizes the comprehensive literature review conducted in paper I. It also elaborates on important benefits and challenges affecting the implementation of the proposed system. Section 2.3 details the scope of work of this dissertation. Section 2.4 explains the data sources, including assumptions and limitations, used for experiments, validations, and refinements of the proposed system.

2.1 Fundamentals of the proposed data-driven PHM system

PdM is one of many technological buzzwords that have become prominent in the last three years. However, to the best of my knowledge, no standard definition of PdM exists in the literature. It has often been used as a generic term for condition-based maintenance (CBM) and reliability centered maintenance [8, 40]. Seeking a more specific definition, I visited the Hannover Messe in April 2019, which is one of the worlds largest industry fairs [39]. It was the first year that PdM was an exhibition topic and the Messe responded by organizing an entire section for PdM, as seen in Figure 2.1. After asking a lot of technical questions to several companies offering PdM solutions, I concluded that none of them managed fault prognostics. However, some companies considered fault detection and fault classification to be state-of-the-art in the industry.

Figure 2.1: The PdM section at the Hannover Messe 2019 [39].

My experience at Hannover Messe led me to conclude that PdM is a data-driven PHM system that does not involve fault prognostics. In other words, the term predictive, here, has nothing to do with RUL predictions. Instead, DNNs are used to make real-time detections of anomalies and predictions of fault-types in the current health state of components to facilitate early warnings and fault diagnostics. Thus, PdM, as performed in the industry today, does not make any future health predictions. Autoships, on the other hand, need to schedule maintenance operations based on future health conditions since there are few or no people on board to perform sudden maintenance actions when needed. Therefore, a data-driven PHM system for autoships must provide fault prognostics.

Figure 2.2 illustrates the main actions and the associated sub-actions of the proposed

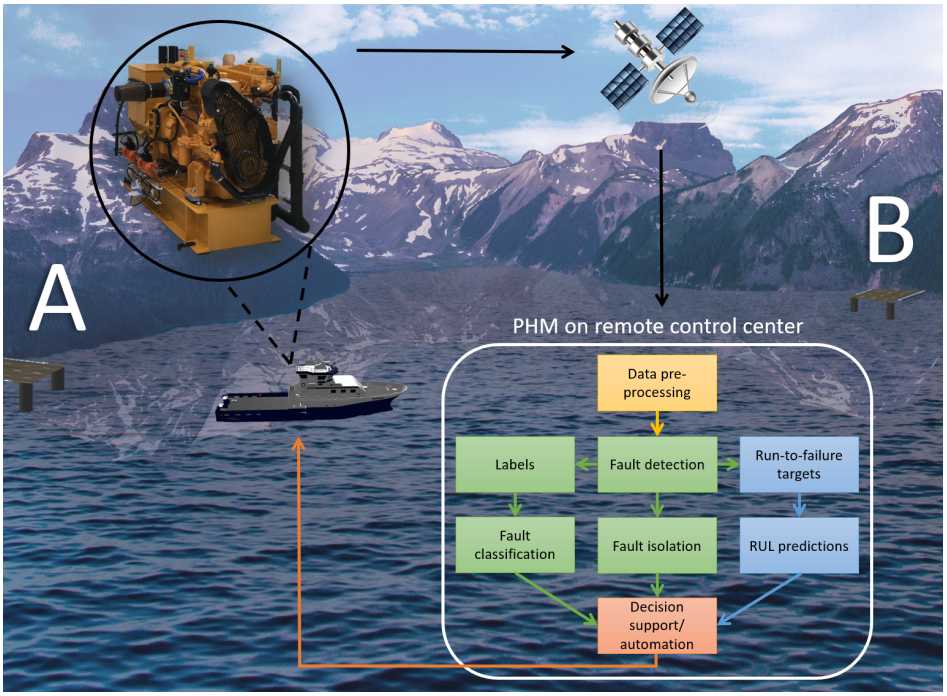


Figure 2.2: An autonomous ferry, crossing a fjord from dock A to B. The resulting analysis obtained from the data-driven PHM system can be used to schedule maintenance operations to the next appropriate dock of call.

data-driven PHM system for autoships, as stated in R01. Furthermore, Figure 2.2 shows an autonomous ferry, crossing a fjord from dock A to B. The marine diesel engine in autonomous ferries has been used in several case studies during this Ph.D. project as such ferries are expected to be in commercial use on the west coast of Norway in the future [3]. See paper V, VI, and VII in appendix E, F, and G, respectively. Due to the fact that there will be limited amounts or no crew members onboard, such ferries need to transfer real-time operational engine data to an RCC to conduct the essential actions of the proposed data-driven PHM system.

The first action is data pre-processing. Due to the various operating conditions the engine is subjected to, a multi-regime normalization method [34] has to be performed on the raw input data to merge the engine loads into one context. Doing so will cause valid input data to be fed, where both the normal operation phenomena and the degradation phenomena are present, to a fault detection algorithm in the next action. Additionally, irrelevant features for the engine will be removed to increase the degradation relevance of the input data [13].

The next action, probably the most crucial, is fault detection. All anomaly detection algorithms are designed to identify deviations from what is considered as normal. In a data-driven PHM system, such deviations or anomalies are considered symptoms of

precursor and/or incipient faults [2, 5]. This action should be performed automatically to indicate that something is wrong. In other words, it indicates that a fault has occurred, but it doesn't indicate which fault-type it is. However, the time step where the fault was detected can be further used to construct both labels for fault classification and RTF targets for RUL predictions. Additionally, fault detection algorithms based on DNNs have the potential to provide fault isolation. Thus, fault detection is considered the most crucial action since the reliability of the algorithm affects subsequent actions. Consequently, the development of a fault-type independent fault detection algorithm, as stated in RO3, during this Ph.D., has been a high priority.

Fault classification aims to provide additional information about detected faults. To do so, fault classification algorithms are employed to classify different fault-types. Based on the detected fault time step in the previous action, the sensor data is automatically labeled with, for example, 0 for normal data points, 1 for one fault-type, 2 for another fault-type, and so on. Then, labeled sensor data is fed to DNNs, including a multi-class classifier, for supervised training. The trained DNNs are then able to predict the probability of which fault-type detected faults belongs to in the current health state of the engine. It is worth noting that normal data points will occur more frequently than faulty data points. Thus, to aid the DNNs in the training phase, it is necessary to bring balance to the labeled sensor data, that is, transforming imbalanced data into balanced data.

To complete the fault diagnostics, fault isolation also needs to be incorporated in the system. Fault isolation tries to provide information about where the fault occurred in the engine. Furthermore, it involves techniques to pinpoint the component that is degraded. Similar to fault classification, this action is also based on the fault detection algorithm. DNNs, such as the variational autoencoder (VAE), can derive a reconstruction of degraded data due to its generative characteristics. This reconstruction can be used to analyze the underlying cause of anomalies to provide fault isolation.

Through fault diagnostics, the system detects anomalies, isolates anomalous components, and predicts the probability of different fault-types. Thus, the next step is to provide information about how faults will progress over time. Fault prognostics algorithms predict the RUL of already-detected and classified fault-types. Such predictions can be used to recommend the ideal maintenance schedule for the ferry. Similar to fault classification, fault prognostics also depend on the accuracy of the fault detection algorithm. The detected fault time step is used to construct RTF targets automatically since DNNs that aim to predict the RUL still depend on supervised training to model degradation processes [17]. It is worth noting that confidence bounds need to be included in any RUL prediction. This is to reduce inherent uncertainties associated with the degradation process and potential flaws in all previous actions of the data-driven PHM system. Maintenance recommendations based on prognostics information should be grounded in confidence bounds instead of a particular RUL value [41].

The final step of the proposed data-driven PHM system is to facilitate decision support or automation to recommend or direct ideal maintenance schedules. Decision support recommends future maintenance operations to a human decision-maker (HDM), while decision automation provides directions for future maintenance operations directly from the system, without the involvement of an HDM. However, as noted in [42], the reliability of data-driven PHM systems needs to be greater than 99% if it is to facilitate

decision automation. Ergo, because of the large uncertainties involved in fault prognostics, an HDM located at the RCC is still required. Additionally, transparent explanations of the outputs from both fault diagnostics and fault prognostics are necessary if HDMs are to understand and trust the system. Such explanations and the outputs have to be shown in a human machine interface. For this purpose, a thin-client web browser can be utilized [43].

2.2 Literature review

> *"Big data can overwhelm traditional approaches and the growth of data is outpacing scientific and technological advances in data analytics."*
>
> - National Institute of Standards and Technology, *2015*

The second research objective, RO2, is to conduct a comprehensive survey of PHM based on DL for autoships. Thus, a literature review paper was written and published during the completion of this Ph.D. research; see paper I in appendix A. This review paper introduces and reviews four well-established DNNs recently applied to various practical fault diagnostics and fault prognostics problems. Furthermore, it discusses benefits, challenges, suggestions, existing problems, and future research opportunities with respect to a data-driven PHM system based on DL for autoships [2].

In the years before lots of researchers jumped on the DL bandwagon, PHM systems depended on so-called traditional diagnostics and prognostics approaches. That is, all other approaches which do not include DNNs. In short, traditional approaches can be divided into data-driven [49] and model-based [26] approaches. Both are based on mathematics. However, the approaches differ in that model-based approaches use algorithms that describe the physics of the component, while data-driven approaches use algorithms built on historical sensor measurements. A combination of these two approaches is called the hybrid approach [50]. Table 2.1 shows the findings in paper I regarding CBM and PHM reviews based on traditional approaches.

Table 2.1: A selection of CBM and PHM reviews based on traditional approaches [2].

Author & Refs.	Year	PHM application	Approaches
Tahan et al. [44]	2017	Gas turbines: diagnostics and prognostics	Data-driven, model-based, and hybrid
Bailey et al. [45]	2015	Engineering systems: diagnostics and prognostics	Data-driven
An et al. [46]	2015	Fatigue crack growth: prognostics	Data-driven and model-based
Lee et al. [40]	2014	Machinery systems: diagnostics and prognostics	Data-driven and model-based
Sikorska et al. [41]	2010	RUL approaches: prognostics	Data-driven and model-based
Vachtsevanos et al. [47]	2006	Book chapter: diagnostics	Data-driven and model-based
Vachtsevanos et al. [35]	2006	Book chapter: prognostics	Data-driven and model-based
Roemer et al. [48]	2006	Engines: prognostics	Data-driven and model-based
Jardine et al. [15]	2006	Machinery systems: diagnostics and prognostics	Data-driven and model-based

With the development of today's interrelated systems, components, and subcomponents and the concurrent rise of big data, traditional approaches confront several challenges [51]. Model-based approaches are reliable if the degradation is modeled precisely [32]. However, they tend to provide low generalization power since they are application-dependent, and hence, time-consuming to expand. Also, traditional data-driven approaches become application-dependent because they require additional dimensionality reduction methods to process the increased volumes of data [52, 53].

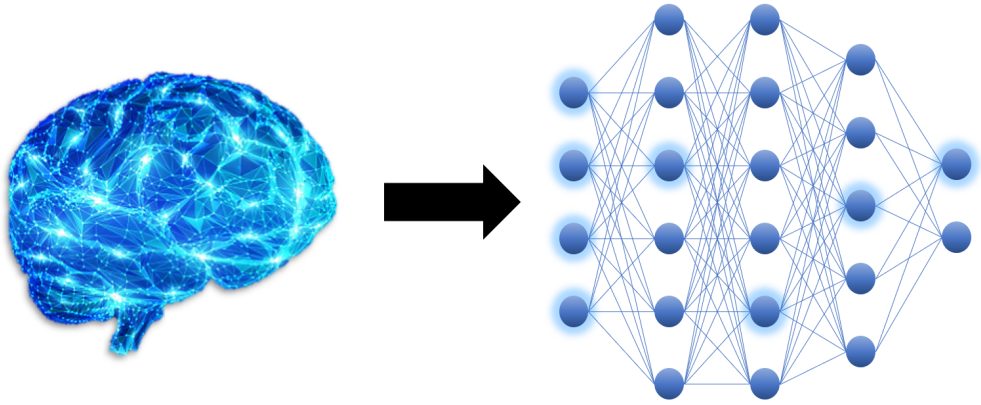


Figure 2.3: DNNs mimic the human brain.

Data guide today's industries. Therefore, it would be both wise and highly beneficial to take advantage of approaches that can process large amounts of data and generalize to new field data and similar industrial applications. This is where DNNs are advantageous. As a matter of fact, the more data you feed DNNs they get better they are [31]. Along with the theory of neuroscience and the utilization of GPUs, DNNs have seen rapid developments in many technological areas, such as self-driving cars [54], computer vision [37, 55], speech recognition [56], language processing [57], and more recently in PHM applications [2]. As seen in Figure 2.3, DNNs mimic the human brain by mathematically approximating the way human neurons and synapses learn by constructing and strengthening weight connections through several iterations. However, unlike a real human brain, DNNs are fundamentally blind to cause and effect. In other words, DNNs cannot interpret and explain their outputs. Also, researchers argue that DNNs cannot ever match true biological intelligence [58].

The four DNNs selected for review in paper I are the autoencoder (AE) and its variations, the convolutional neural network, the deep belief network, and the long-short term memory (LSTM). See appendix A for the complete review of recent applications to PHM of each of these four DNNs. At that time, they were proposed as the four main candidates to be included for both fault diagnostics and fault prognostics in a data-driven PHM system based on DL for autoships.

2.2.1 Benefits and challenges

Benefits

- Conventional ships are often over-engineered by built-in redundancy. For example, R/V Gunnerus incorporates three marine diesel engines. So, if a critical failure occurs, the ship can still complete its operational task to some degree. This design philosophy is highly related to historical inaccessibility to shore [2, 59]. However, telecommunication companies, such as Inmarsat, have launched several data transfer satellites during the last decade, which can provide high-speed broadband connections to ships almost anywhere in the world [3]. This will enable new de-

sign philosophies, including data-driven PHM systems, as options to enhance the current redundancy policy.

- The final goal of a data-driven PHM system is to achieve zero-downtime performance. Real-time RUL predictions, including confidence bounds, of components and sub-components enables HDMs at an RCC to schedule maintenance operations to the next appropriate port of call, or in worst case, dispatching maintenance personnel before a failure occurs when autoships are still in operation [2, 3]. This will significantly increase operational availability, system safety, and cost-benefits. Additionally, reliable predictions, over time, will build trust that autonomous maritime activities are safe [60].
- Back in 2012, the German-based insurance company Allianz reported that between 75% and 96% of all marine accidents are a result of human errors [3]. Such errors generally happen when humans are exhausted and complex maritime conditions require humans to make tough decisions based on experience and intuition alone [60]. Overall, autoships will reduce the influence of HDMs [61]. This is also the case for a data-driven PHM system [2].

Challenges

- Autoships require significant adaptations in the organizational culture of the maritime industry [7]. For example, it is necessary to have confidence in so-called "black-box" systems. A data-driven PHM system based on DL falls into this category as it will recommend directions for future maintenance operations. The most difficult challenge is that today's DNNs lack transparency [62, 63]. Due to the non-linear network structure of DNNs, they do not provide a human-understandable explanation of their outputs. But humans need to understand how outputs are created if they are to trust the system, which is crucial in critical industrial applications, such as health care [64] and autonomous vessels. However, explainable artificial intelligence (XAI) can ease this issue, as it uses methods for visualizing, explaining, and interpreting DNNs [65, 66]. Successfully incorporating XAI in the final action of a data-driven PHM system is extremely important in relation to autoships.
- Another concern is the continuous flow of operational sensor data to the RCC. Autoships depend on diverse automated systems and associated sensor devices to perform their main functions [2]. Thus, the sensor data might become unstructured, while the various operating conditions further complicate the sensor data. The data-driven PHM system has to provide automatic pre-processing procedures that tackle this kind of sensor data complexity. The continuous data flow also presents a cybersecurity challenge [3].
- Conventional ships are typically equipped with systems and equipment from several different manufacturers [67]. This results in several stand-alone and consequently uncoordinated monitoring systems that make the implementation of a data-driven PHM system for more than one component difficult and time-consuming. Thus, future data-driven PHM systems need to be included in the building and design phase of autoships [14].

- Today, conventional ships are usually application-designed and produced in batches of two to ten vessels [7]. A consequence of this is a slow accumulation of failure data compared to, for example, the aviation industry that produces hundreds of the same airplane in a series [2]. In addition to the diversity of equipment and system manufacturers, these are the main reasons for the common lack of RTF data in the maritime industry. Therefore, manufacturers and shipowners need to start saving and sharing their RTF data to build extensive databases. This would be advantageous for the realization of a data-driven PHM system.

2.3 Scope of work

The proposed data-driven PHM system can be divided into four main categories, as seen in Figure 2.4. This dissertation is based on a three-year Ph.D. project. Thus, instead of doing time-limited research in all four categories, this dissertation has focused its research within the most important areas for data-driven algorithm development. The development of a fault-type independent fault detection algorithm for maritime components, as stated in RO3, has been of high priority. The algorithm was first developed in paper III, explored in papers IV and V, and further improved in paper VI. See appendix C, D, E, and F, respectively. As opposed to fault detection, algorithm development of both fault classification and fault isolation have been given low research priority. For example, state-of-the-art DNNs for fault classification already exist [28, 29, 68]. To further improve fault classification, techniques for handling imbalanced data, such as focal loss [69], under- and oversampling [70], and weighted loss functions [71], are important to investigate. This is because the minority classes, which are the fault classes, are of high importance for the proposed data-driven PHM system. For instance, it is not critical if the system miss-classifies a normal condition as a fault condition. On the contrary, if the system miss-classifies a fault condition as a normal condition, it could lead to downtime and a potential disaster for autoships.

As seen in Figure 2.4, great emphasis is also given to fault prognostics. No matter the industrial application, fault prognostics are still under research and development. Thus, to increase the reliability of fault prognostics, as stated in RO4, has been a prime concern throughout this dissertation. First of all, to improve the RUL prediction accuracy of DNNs, they must incorporate diagnostics information in the supervised training phase [13]. Therefore, detected fault time steps, obtained from the fault detection algorithm, are used to construct RTF targets automatically and predict the RUL in paper IV in appendix D. Also, different approaches for constructing RTF targets are heavily investigated. Papers II and VII in appendix B and G, respectively, are also attempts to increase the reliability of fault prognostics. Paper II investigates the effect of unsupervised pre-training in RUL predictions. This initial training step extracts abstract degradation related features that improve the generalization power of DNNs. Paper VII proves the advantage of both data augmentation and skip connections. Consequently, a novel data augmentation technique for time-series data and the SkipRnet are proposed.

It is worth noting that proper data pre-processing is extremely important for DNNs for both fault diagnostics and fault prognostics purposes. Hence, data pre-processing is well-explained in all papers, except the literature review in paper I. The papers with the most novelty, in terms of data pre-processing, are papers VI and VII. Paper VI introduces multi-regime normalization to convert engine loads into one context, while

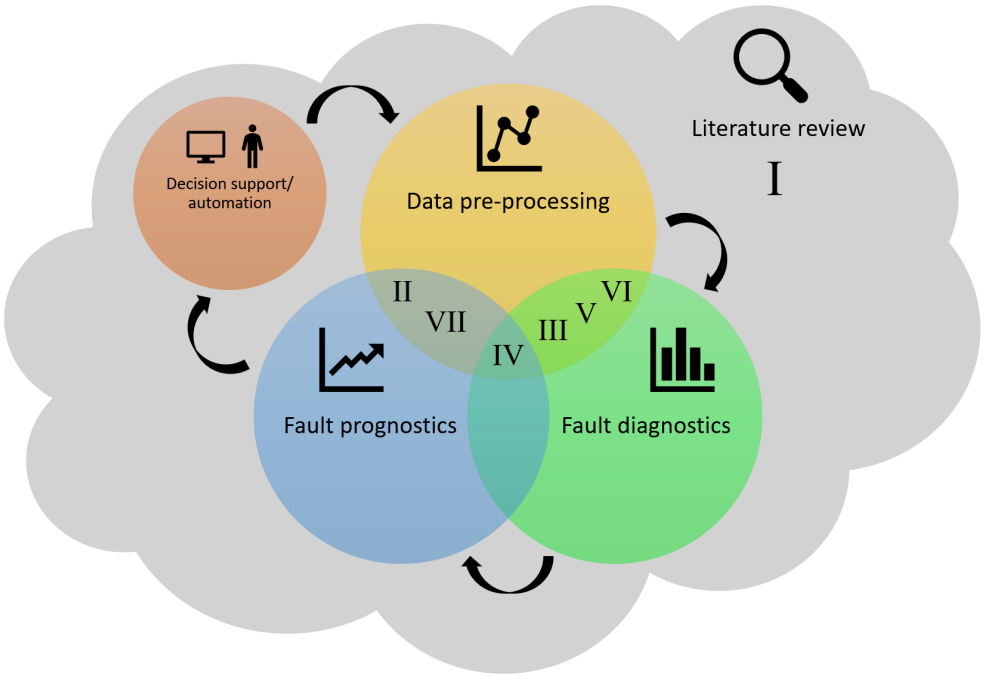


Figure 2.4: Scope of work.

paper VII proposes a novel data augmentation technique to construct more RTF data.

The final action of the proposed data-driven PHM system has been given low research priority during this Ph.D. project. However, if the system is to be employed in future autoships, decision support or automation is extremely important to consider. This final action should in theory be an entire dissertation in itself as XAI has just begun to gain popularity. Additionally, confidence bounds should be incorporated into the decision support or automation category because maintenance recommendations and their corresponding scheduling should be based on confidence bounds rather than a particular RUL prediction.

In this dissertation, the application is aimed towards the maritime industry, and more specifically, at autoships. However, the proposed data-driven PHM system and the accompanying research findings can also easily be applied to a broad range of other industrial domains. The main three features that need to be available are a system that degrades over time and, of course, sensor measurements of related NOP data and RTF data. So, to conduct the following research experiments, the accumulation of operational sensor data has been essential. The following section introduces all data sources, including assumptions and limitations, collected and used in this dissertation.

2.4 Data accumulation, limitations, and assumptions

This section introduces the three main data sources used during this Ph.D. project. Limitations and assumptions made of each data source are also explained.

2.4.1 Benchmark data

In the development process of DNNs for an industrial application, it is highly beneficial to have a publicly available benchmark data set. For example, data sets collected from an industrial application might be subjected to different degrees of complexities. Consequently, two different DNNs proposed for the same industrial application but trained on different data sets might provide results biased by the data. A benchmark data set enables researchers to train, refine, and validate their proposed DNNs on the exact same data set. Therefore, different DNNs can be compared directly without being biased by the data. Besides, the knowledge learned from benchmark data can easily be transferred to other industrial applications because DNNs are generic.

Figure 2.5: A turbofan engine.

Within the PHM domain, the C-MAPSS data set is acknowledged as the benchmark data set for fault prognostics. It is produced by the National Aeronautics and Space Administration and is designed to accelerate the development of data-driven prognostics algorithms [33]. As shown in Table 2.2, the complete data set is further divided into four subsets, where each subset exhibits different complexities. Subset FD001 exhibits the lowest degree of complexity as it is only subjected to one operating condition and one fault-type. In contrast, subset FD004 exhibits the highest degree of complexity. Nevertheless, each subset is divided into a training set and a test set of multiple multivariate time-series. Each time-series includes 24 sensor measurements of a turbofan engine, used in airplanes, as seen in Figure 2.5. Each time-series also starts with different degrees of initial wear and manufacturing variations. All engines operate in normal conditions at the start before they begin to degrade at a random time step during the time-series. The engines in the training sets degrade until failure, and hence, the time-series can be considered to be RTF data. The degradation in the engines in the test sets, however, ends sometime before failure, that is when RUL > 0. Thus, the main objective of the C-MAPSS data set is to predict the correct RUL value for each engine in the test sets. True RUL targets for the last time step for each engine in the test sets are provided to evaluate the RUL predictions.

Table 2.2: The C-MAPSS data set [72].

Data set	FD001	FD002	FD003	FD004
Time-series in the training set	100	260	100	249
Time-series in the test set	100	259	100	248
Operating conditions	1	6	1	6
Fault-types	1	1	2	2

Benchmark data sets do not exist in the maritime industry yet. However, such data sets would be highly beneficial for the research community in the years to come. This could be realized if stakeholders agreed to cooperate to save and share data. Even though the C-MAPSS data set is extensive and highly complex, it is still simulated data. As a consequence, the results might not be as trustworthy as results based on real-life industrial data for most applications, such as autoships. Thus, in addition to benchmark data, case studies based on real-life industrial data are of high importance to conduct credible research. The following subsections describe real-life industrial data sources used during this Ph.D. project.

2.4.2 Industrial company

This data source consists of five real-operation RTF data sets, which have been provided by an industrial company located on the west coast of Norway. All data sets are collected from the same maritime component. The actual name of the maritime component, fault-types, and sensor measurements, cannot be provided in this dissertation due to a confidentiality agreement. As seen in Table 2.3, each data set differs in total time step length T_{total}, where one time step equals one second. Data sets 1 and 4 are subjected to fault-type A, while data sets 2, 3, and 5 are subjected to fault-type B. Similar to the C-MAPSS data set, in each data set, the maritime component is run in NOP condition at the start, then begins to degrade at an unknown time step during the data collection process. The degradation grows in magnitude until failure, and therefore all five data sets can be considered as RTF data. The main objective of all data sets is to detect the time step where the degradation starts, namely, where the fault occurred, automatically. To evaluate predicted detections, valuable human domain knowledge (HDK) provided by the industrial company is used to determine the true fault time step f_t for each data set. The initial 25% of each data set is considered NOP data (training data), while the remaining 75% is considered faulty degradation (FD) data (test data). Each data set has 14 sensor measurements. Additionally, different magnitudes of random white Gaussian noise are added to each training data set in order to create disparate real-life situations. Thus, an assumption is made that real-world noise approximates random white Gaussian noise.

Table 2.3: Real-life RTF data collected from a maritime component [19].

Data set	Fault-type	T_{total}	T_{NOP}	T_{FD}	f_t in T_{FD}
1	A	887	222	665	157
2	B	909	227	682	148
3	B	1859	465	1394	477
4	A	2554	638	1916	1306
5	B	3643	911	2732	787

2.4.3 Hybrid power lab

The data collected from the hybrid power lab at the Department of Ocean Operations and Civil Engineering at NTNU in Aalesund has been the main data source during this Ph.D. project. Unlike benchmark data, data collected from real-life systems is often unstructured. For example, the logging frequency might be different between different sensors, alarms and sensors from different components might have been merged into one

Figure 2.6: The hybrid power lab. The picture to the left shows the automation system, the picture in the middle shows the battery system and the diesel engine, while the picture to the right shows restriction devices used to provoke fault-types [5].

data collection, missing or non-defined values, and so on. Such unstructured data cannot be fed to DNNs directly, and as a result, data pre-processing is often necessary for real-life systems. Unstructured data is also the case for the hybrid power lab. Around 500 alarms and features of all components in the system were reduced to 47 time-variant features [73].

As seen in Figure 2.6, the lab includes a marine automation system to control the entire system, a marine battery system, and a small marine diesel engine. The produced power is supplied back to the power grid to simulate load changes in the system. During the data collection, the engine was run by two different load profiles to replicate two different environmental conditions autonomous ferries may encounter on the west coast of Norway. At the very start, the ferry is assumed to off-load and on-load vehicles before it leaves the dock at a safe and constant velocity. Next, the ferry speeds to a suitable velocity with respect to the weather. This velocity is kept constant until it decreases safely. In the end, the ferry breaks just before it docks. The two profiles are exposed to the same order of magnitude of engine loads, but the length of each engine load varies to reflect different environmental conditions. Figure 2.7 compares the two engine load profiles, profile 1 and profile 2.

Both NOP data and FD data are collected from both profiles. The difference between NOP data and FD data is that a fault is introduced at an unknown time step in the latter. To evaluate predicted fault detections, Finn Tore Holmeset, an engine chief engineer with 13 years of sailing experience and three years of experience with the development of a health monitoring system for rotary machinery, provided expert HDK to determine the true fault time step f_t for each degradation data set. Three different fault-types have been introduced during this dissertation. These are the air filter fault, the cooling system fault, and the turbo fault. The fault-types are provoked to simulate gradual degradation for different subsystems in the engine. The air filter fault

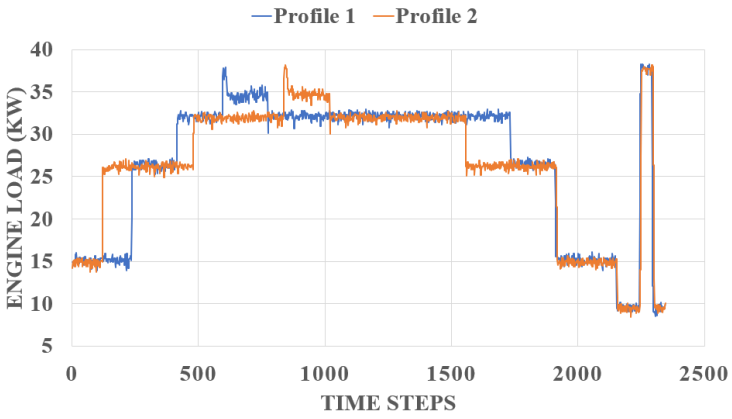


Figure 2.7: Profile 1 vs. profile 2 [14].

demonstrates the effect of a clogged air filter. This fault is provoked by a restriction device, as seen in Figure 2.6, which is gradually adjusted from fully open to 90% closed to reduce the inlet flow of air to the turbocharger. The engine has a secondary water cooling system to cool the primary water cooling system. The primary cooling system is controlled internally in the engine by a bi-metal thermostatic valve, while the secondary cooling system is controlled by a frequency-operated fan circulating air through a heat exchanger. The cooling system fault is a malfunction of the fan that demonstrates loss of cooling efficiency. The turbo fault is introduced to replicate efficiency reduction in the turbocharger. As seen in Figure 2.6, a bleed device on the charge air pipe between the turbocharger and the engine inlet manifold is used to simulate gradually bleeding of air. This results in reduced air pressure to the engine combustion process. The cooling fault is used in papers V and VI, and both the air filter fault and the turbo fault are used in papers VI and VII. Table 2.4 summarizes the seven data sets collected from the hybrid power lab.

The work conducted in paper V can be considered as the initial experiment on the hybrid power lab, where only the cooling fault and profile 1 are used. As a consequence, fewer assumptions were made and the total duration of the ferry crossing was 22 minutes and 40 seconds, which equals 2,720 time steps. However, the logging system was subjected to several improvements in the transition between paper V and papers VI and VII. A more reliable logging frequency of 2 Hz was implemented, and hence, the total duration of the two engine load profiles was reduced to 22 minutes and 33 seconds, which equals 2,706 time steps. Besides, the number of decimal places was increased, which led to a change in the true f_t for the cooling system fault.

In papers VI and VII additional assumptions were made. First, the initial 360 time steps, that is, the initial three minutes, were removed from all data sets to acquire almost the same initial measurements for each sensor in each data set. This was performed because all data sets were collected at different dates and times, and therefore in conditions of different ambient temperatures, etc. Second, sensor measurements of both the cooling water temperature to the engine and the fuel consumption were removed from all data

Table 2.4: The seven data sets collected from the marine diesel engine [5, 13, 14].

Data set	Profile	Time steps	f_t
Normal operation	1	2,346 (2,720 in paper V)	–
Normal operation	2	2,346	–
Air filter degradation	1	2,346	1,670
Air filter degradation	2	2,346	1,433
Cooling system degradation	1	2,346 (2,720 in paper V)	1,713 (1,979 in paper V)
Turbo degradation	1	2,346	1,431
Turbo degradation	2	2,346	1,427

sets. The cooling water temperature to the engine is considered to be an unknown parameter. This feature is affected by the outdoor temperature, and hence, it varies when data sets are collected at different dates and seasons. The fuel consumption is an important feature for the combustion process in the engine. Nevertheless, the measurements obtained from the automation system were quite inaccurate. Finally, it is worth noting that real-life RTF data sets on ships are normally accumulated and collected through months, or perhaps even years. In this dissertation, however, the data sets are collected more rapidly due to time constraints. Even though the collected RTF data sets from the hybrid power lab only consist of 2,346 time steps, the real degradation patterns are assumed to remain.

3 Case study: the C-MAPSS data set

This chapter presents the research findings and important discussions concerning fault diagnostics and fault prognostics of the first data source used in this dissertation, namely, the C-MAPSS data set. As already mentioned, the C-MAPSS data set is considered to be the benchmark data set within the PHM domain. Such benchmark data provides the possibility to focus the research purely on DNNs since the data is ready to use. Additionally, the results can be compared against other researchers' work across the entire world. More importantly, the knowledge learned from benchmark data can be transferred to the maritime industry and autoships. This chapter is divided into three main sections: data pre-processing in Section 3.1, the results and discussions of both fault diagnostics in Section 3.2, and fault prognostics in Section 3.3. Section 3.3 is an initial attempt to respond to RO4 in this dissertation, that is, to increase the development and reliability of fault prognostics. Supplementary content related to this chapter can be found in papers II and IV in appendix B and D, respectively.

3.1 Data pre-processing

Advanced data pre-processing is rare for benchmark data because the data is already structured and divided into a training set and a test set. However, proper data normalization is necessary as the features in the C-MAPSS data set is subjected to different ranges. The z-score normalization method is used in both paper II and paper IV. For each feature in the training set, this method subtracts the mean and scales it to unit variance. Then, the normalization statistics obtained from the training set are applied to the test set. A signal-to-noise ratio (SNR) of 95% is also applied to the training set in paper IV to improve generalization.

3.2 Fault diagnostics

The C-MAPSS data set is mostly used for fault prognostics purposes, that is, predicting the RUL of the turbofan engines. Today, DNNs that aim to predict the RUL still require RTF targets to model the degradation process during supervised training. Previous studies have depended on the piece-wise linear (PwL) degradation model, which Heimes et al. [74] proposed in 2008, to construct RTF targets for the C-MAPSS data set [22, 32, 75]. This degradation model assumes the same constant initial RUL (R_i) value for all engines when they run in NOP. Then, the model degrades linearly until failure. This means that the constructed RTF targets ignore the entire fault diagnostics aspect because the degradation model only depends on the total number of time steps in each engine. However, the time step where the degradation starts is essential information to obtain to construct more reliable RTF targets for each engine in the training set. In the following subsection, the fault detection algorithm, proposed in paper III, is used to

detect the time step where the fault occurred for each engine in subset FD001.

3.2.1 Fault detection

The original fault detection algorithm is based on an unsupervised reconstruction-based training framework [19]. In other words, at each time step t the input vector $\boldsymbol{x}_t$ is also used as the target vector $\boldsymbol{y}_t$ for reconstruction, $\boldsymbol{y}_t = \boldsymbol{x}_t$. Then, at each t a reconstruction error is calculated by the mean squared error (MSE). Detailed information concerning the algorithm is found in appendix C. This training framework is also sometimes referred to as semi-supervised in the literature [76, 77].

For subset FD001, the initial 25% of each engine is considered NOP data, while the remaining is considered as FD data. In short, the algorithm is trained only on NOP data. Then, when FD data is fed to the algorithm, it outputs the reconstruction error as an anomaly score (AS). The AS is then smoothed before its acceleration is calculated. The original fault criterion of the algorithm was the maximum acceleration. However, in paper IV, it was discovered that the maximum acceleration was not suitable for FD001 as the degradation grows with increasing acceleration until failure for each engine. Hence, a dynamic acceleration threshold was created and used as the fault criterion for subset FD001. The predicted fault time steps $\hat{f}_t$ are detected once the acceleration exceeds the threshold. Table 3.1 shows the fault detection results in paper IV. Furthermore, it shows the total time step length T_t, $\hat{f}_t$, and the corresponding R_i for each engine in the training set of subset FD001. Thus, an optimized R_i value for each engine is obtained. These values provide information about the true degradation pattern in each engine and can be further used to construct reliable RTF targets for DNNs that aim to predict the RUL.

It is worth noting that subset FD001 only contains one fault-type and one operating condition, as shown in Table 2.2. If, however, several operating conditions were intro-

Table 3.1: Total time step length T_t, predicted fault time step $\hat{f}_t$, and corresponding initial RUL value R_i for each engine in the training set of subset FD001. 20% of the training set was randomly designated as the cross-validation set for hyper-parameter tuning of DNNs [17].

Training set																Cross-validation set			
Engine	T_t	$\hat{f}_t$	R_i	Engine	T_t	$\hat{f}_t$	R_i	Engine	T_t	$\hat{f}_t$	R_i	Engine	T_t	$\hat{f}_t$	R_i	Engine	T_t	$\hat{f}_t$	R_i
1	192	63	129	26	199	101	98	53	195	99	96	78	231	91	140	2	287	171	116
3	179	47	132	27	155	85	70	54	257	93	164	79	199	114	85	5	269	141	128
4	189	62	127	28	165	72	93	55	193	77	116	80	185	58	127	7	259	124	135
6	188	86	102	29	163	100	63	57	137	47	90	82	214	106	108	13	163	69	94
8	150	53	97	30	194	116	78	58	147	94	53	83	293	176	117	21	194	98	96
9	201	86	115	32	191	121	70	59	231	117	114	84	267	135	132	31	234	149	85
10	222	70	152	34	194	66	128	60	172	85	87	85	188	88	100	33	200	110	90
11	240	120	120	35	181	111	70	61	185	104	81	86	278	119	159	42	196	103	93
12	170	94	76	36	158	52	106	62	180	99	81	87	178	105	73	46	256	91	165
14	180	76	104	37	170	77	93	63	174	84	90	88	213	80	133	47	214	129	85
15	207	86	121	38	194	114	80	64	283	140	143	89	217	105	112	50	198	79	119
16	209	72	137	39	128	58	70	66	201	83	118	90	154	73	81	52	213	79	134
17	276	123	153	40	188	79	109	67	312	130	182	91	135	46	89	56	275	90	185
18	195	71	124	41	216	88	128	69	362	245	117	92	340	167	173	65	153	59	94
19	158	39	119	43	207	98	109	71	208	96	112	95	283	114	169	68	199	71	128
20	234	104	130	44	192	131	61	72	213	83	130	96	336	204	132	70	137	84	53
22	202	103	99	45	158	73	85	73	213	104	109	97	202	66	136	74	166	75	91
23	168	94	74	48	231	81	150	75	229	94	135	98	156	68	88	81	239	100	139
24	147	60	87	49	215	75	140	76	210	147	63	99	185	74	111	93	155	57	98
25	230	129	101	51	213	92	121	77	154	38	116	100	200	71	129	94	258	101	157

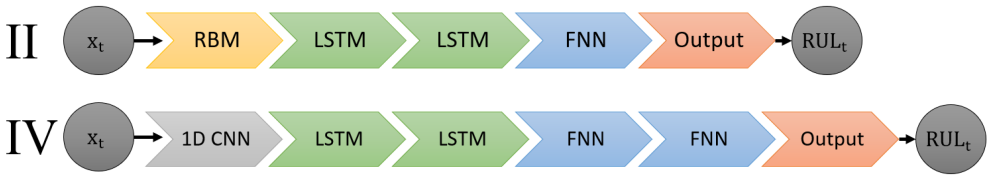


Figure 3.1: DNN structures proposed for the C-MAPSS data set [17, 22].

duced in the data set, the unsupervised reconstruction-based fault detection algorithm would face problems since the sensor measurements might differ strongly between different time steps with different operating conditions. Clearly, this is an issue concerning the various operating conditions in the maritime industry. This issue is further explored in Section 5.1.

3.3 Fault prognostics

Today, fault prognostics is an area of active research and inventions [34]. As a result, each week new DNN structures are proposed to predict the RUL in the literature. Thankfully, recent RUL prediction research studies on the C-MAPSS data set [32, 75, 78] can be used as a guide to determine which DNNs to include and which to omit. During this Ph.D. project, two DNN structures were proposed to predict the RUL on the C-MAPSS data set. Both structures are introduced in the following subsection.

3.3.1 Proposed deep neural networks

Figure 3.1 shows the DNN structures proposed in papers II and IV. Similar to [78], both structures include two LSTM layers in both the second and the third layer. These LSTM layers are included so that the DNNs learn statistical degradation patterns and long-term dependencies within the temporal information in the sensor data of the turbofan engines. Also, a feed-forward neural network (FNN) layer is attached in the fourth layer to map all extracted time-dependent features to a one-dimensional vector. A time-distributed fully connected output layer is, as well, attached in the final layer to handle error calculations and provide RUL predictions.

The major difference between the two DNN structures is the first layer. The DNN structure in paper II uses a restricted Boltzmann machine (RBM) layer, while the DNN structure in paper IV uses a one-dimensional convolutional neural network (1D CNN) layer. In paper II, the main goal was to improve generalization by performing an initial unsupervised pre-training stage. Therefore, the RBM layer was included to initialize the weights between the first and the second layer in a region near a good local minimum before supervised fine-tuning of the whole structure was conducted. The proposed structure was trained on both completely and reduced amounts of labeled training data, and showed promising generalization power towards the test set compared to purely supervised training. Large amounts of high-quality labeled training data might be both challenging and time-consuming to acquire, especially in the maritime industry. However, as mentioned in Section 3.2, this issue is solvable by utilizing the unsupervised

reconstruction-based fault detection algorithm to automatically construct RTF targets. Thus, the RBM layer was deprecated and replaced by a 1D CNN layer in paper IV. Similar to [32], the 1D CNN layer is included to extract and learn low-level temporal features from each sensor measurement individually. These features might contain important degradation information which are then used to form more complex patterns within the remaining layers.

Another difference between the two structures is that in paper IV a second FNN layer is included to act as a stand-alone dropout layer in the structure. This is an attempt to reduce the number of tune-able hyper-parameters compared to the structure in paper II, where dropout is applied to all layers. Dropout randomly drops 10-50% of the units during training, and hence, approximately connects an exponential number of different structures. This improves generalization since it prevents the structure from extracting the same degradation features repeatedly [79].

3.3.2 Validation of run-to-failure targets

The main goal in paper IV is to validate three different labeling approaches to construct RTF targets to be used in the supervised training procedure of DNNs that aim to predict the RUL. The three approaches are the PwL degradation model, descriptive statistics (DS) with polynomial regression, and the smooth AS function (ASF) obtained from the unsupervised reconstruction-based fault detection algorithm. All approaches utilize the optimized R_i values, as seen in Table 3.1. A detailed description of each approach is found in paper IV in appendix D. Figure 3.2 compares the three different RTF targets for engine one in subset FD001. Both DS targets and ASF targets are nonlinear, while the PwL targets are, as the name indicates, linear. During the experiments in paper IV, it was discovered that the PwL targets outperformed both DS targets and ASF targets in all performance aspects of RUL predictions. Consequently, when training in a supervised manner on RTF data, it seems that it is more convenient for DNNs to map inputs to a linear target compared to a nonlinear target. Linear targets are also beneficial if the

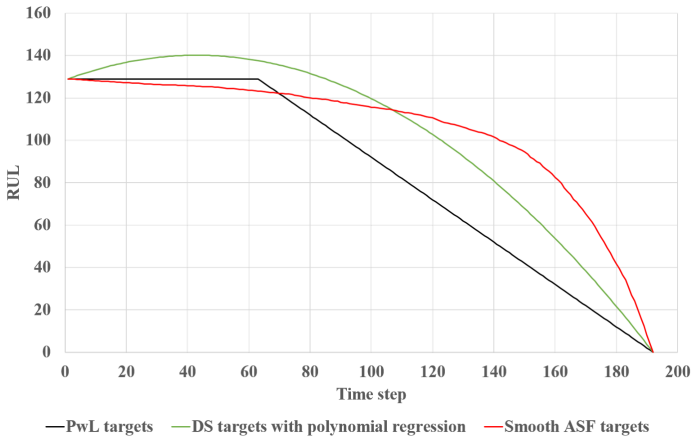


Figure 3.2: Comparison of different RTF targets [17].

RUL is to be considered as a time-based index, e.g., if the RUL decreases by one and the time step increases by one. This is highly relevant for autoships if future maintenance schedules are to be optimized by visualized RUL predictions.

As seen in Table 3.1, there is a high variance between the R_i values. This makes it difficult for the proposed DNN in paper IV to predict the RTF targets when the engines run in NOP. On the contrary, the proposed DNN is extremely accurate to predict the RTF targets in FD data close to the end of the engines lifetime. This also aids the generalization of the proposed DNN towards new field data, that is, the test set. Based on the research findings in paper IV, the PwL degradation model with optimized R_i values is also used in the case study for fault prognostics of the marine diesel engine in autonomous ferries in in Section 5.3.

3.3.3 Tuning of hyper-parameters

Within the DL domain, hyper-parameters can be defined as all parameters which can be set by a human or an algorithm before the training starts. DNNs do normally introduce a big number of such hyper-parameters, which can be both challenging and time-consuming to optimize during supervised training procedures. To ease this issue, a genetic algorithm (GA) approach was first proposed in paper II, and then further used in paper IV, to automatically tune hyper-parameters during this dissertation. See appendix B and D for a detailed description of the GA approach.

Importantly, before the GA approach starts, or any other hyper-parameter tuning approach for that matter, the complete training data has to be divided into training and cross-validation. The cross-validation set is used to tune the hyper-parameters and ensures that the DNN has learned most of the statistics in the training data correct. The performance on the cross-validation set also indicates if the DNN is overfitting or underfitting the training data. Then, when an acceptable performance on the cross-validation set is achieved, the trained DNN is applied on a stand-alone test set that it has never seen before. The performance on the test set specifies how well the DNN can generalize on new field data.

In both paper II and IV, the hold-out cross-validation method is used. This method selects a portion of the complete training data as the cross-validation set randomly. By this method, it is assumed that any random portion selected from the complete training data exhibits similar statistics. Consequently, to use this method, the total number of examples in the complete training sets, in all subsets of the C-MAPSS data set, have been considered large enough. For smaller data sets, the k-fold cross-validation method is usually used to reduce a potential bias of statistics.

The GA approach has proved to be an effective algorithm for finding a near-optimal solution in a selected search space of hyper-parameters. In paper II, the search space consisted of 8,748,000 possible combinations. However, in paper IV the search space was considerably reduced to 11,664 combinations to improve both efficiency and performance. The average training time was also considerably reduced from paper II, 60 hours, to paper IV, 13.33 hours. This reduction in training time was mainly due to GPU optimization of the LSTM layers, which was implemented in the DL4J library in the transition between paper II and IV. Nevertheless, the training time of the GA approach has not been critical during this Ph.D. project since it was mostly run during nights and weekends.

The following bullet-points present the knowledge learned based on the experience from the C-MAPSS data set concerning tuning of hyper-parameters of DNNs that aim to predict the RUL. This knowledge is transferred to the remaining case studies of this dissertation.

- It is a good idea to fix the random seed of weight initialization to ensure that the results are reproducible for other researchers.
- The total number of parameters (weights and biases) in a DNN should reflect both the number of examples and the complexity in the data set.
- To prevent overfitting and reduce the training time, early stopping is useful to use in supervised training procedures.
- The learning rate is the first hyper-parameter to tune, and it should be tuned roughly before any hyper-parameters tuning approach starts.
- To better maintain important low-level degradation features, the learning rate in the first layer can be a half order of magnitude higher than the learning rate in the remaining layers.
- Stochastic gradient descent as the optimization algorithm together with adaptive moment estimation (Adam) [80] as the learning rate method has proved excellent performance on the C-MAPSS data set.
- A $l2$ regularization coefficient between $1 \cdot 10^{-3}$ and $1 \cdot 10^{-6}$ is normally a good choice to reduce overfitting.
- Normally, it is sufficient to only apply dropout to the last layer before the output layer.
- To select the rectified linear unit (ReLU) activation function [81] in FNN layers and the tanh activation function in LSTM layers are usually a good choice.
- For most RUL applications, Xavier weight initialization [82] is a solid choice.

3.3.4 Remaining useful life predictions compared with the literature

To evaluate the RUL prediction results on test sets in the C-MAPSS data set, the scoring function (S), provided in [33], and the root mean square error (RMSE) are normally used:

$$\mathrm{S} = \begin{cases} \sum_{i=1}^{n} e^{-(\frac{d_i}{13})} - 1, & \text{for} \quad d_i < 0 \\ \sum_{i=1}^{n} e^{(\frac{d_i}{10})} - 1, & \text{for} \quad d_i \geq 0 \end{cases} \tag{3.1}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} d_i^2} \tag{3.2}$$

where n is the total number of true RUL targets in the test set and $d_i = RUL_{predicted,i} - RUL_{true,i}$. The RMSE gives equal penalty to early and late RUL predictions, namely,

Table 3.2: Recent results on the C-MAPSS data set.

Author & Refs.	Year	R_i	FD001		FD002		FD003		FD004	
			S	RMSE	S	RMSE	S	RMSE	S	RMSE
Ramasso [83]	2014	Optimized	216	13.27	2,796	22.89	317	16.00	3,132	24.33
Babu et al. [84]	2016	130	1,287	18.45	13,570	30.29	1,596	19.82	7,886	29.16
Malhotra et al. [85]	2016	125	256	12.81	n/a	n/a	n/a	n/a	n/a	n/a
Zheng et al. [78]	2017	130	338	16.14	4,450	24.49	852	16.18	5,550	28.17
Zhang et al. [75]	2017	n/a	334	15.04	5,585	25.05	422	12.51	6,558	28.66
Yoon et al. [86]	2017	140	419	14.80	n/a	n/a	n/a	n/a	n/a	n/a
Li et al. [32]	2018	125	274	12.61	10,412	22.36	284	12.64	12,466	23.31
Ellefsen et al. [22] (II)	2019	115,135,125,135	231	12.56	3,366	22.73	**251**	**12.10**	**2,840**	22.66
Ellefsen et al. [17] (IV)	2019	Optimized	**186**	**12.08**	n/a	n/a	n/a	n/a	n/a	n/a
Miao et al. [23]	2019	Optimized	n/a	12.29	n/a	17.87	n/a	14.34	n/a	21.81
da Costa et al. [87]	2020	125	300	13.67	**1,638**	**17.80**	267	12.57	2,904	**21.30**

when $d_i < 0$ and $d_i > 0$, respectively. In S, the penalty for late RUL predictions is larger. For example, if the predicted RUL is 100 and the true RUL is 90 in a real-life PHM system, the system is prone to dangerous situations as maintenance operations will be scheduled too late. On the contrary, early predictions pose less risk to system failures.

Table 3.2 presents recent results on the C-MAPSS data set in the literature. In the third column, an integer value indicates the same constant R_i for all engines in the entire C-MAPSS data set or for a specific subset, while the term “optimized" indicates some data-driven labeling approach to obtain optimized R_i values for each engine according to the actual engine health. It is worth noting that some studies are only using one out of four subsets in their experiments. Additionally, some studies are only using the RMSE for performance evaluation. Thus, in Table 3.2, “n/a" indicates not available information. The best results for each subset are highlighted in bold.

4

Case study: industrial company

This chapter presents the research results and important discussions concerning data pre-processing and fault diagnostics of the second data source in this dissertation. As explained in Section 2.4, this data source includes five RTF data sets, which are provided by an industrial company. Thus, as opposed to the previous case study, this chapter includes real operation data from a maritime component. This chapter is divided into two main sections; data pre-processing in Section 4.1 and the results and discussions of fault diagnostics in Section 4.2. Furthermore, Section 4.2 presents the initial development of the fault detection algorithm, as stated in RO3. Supplementary content related to this chapter is found in paper III in appendix C.

4.1 Data pre-processing

The RTF data sets were collected from a logging system only concerning the maritime component. Besides, the data sets do not face any problems related to real-life systems, such as varying logging frequencies between different sensors, missing and non-defined values, etc. The data sets are therefore structured and ready to use from start. All data sets start under different operational loads and corresponding sensor measurements. However, the starting operational load for each data set does not change drastically from NOP to failure. As a consequence, the issue of diverse operating conditions throughout a data set, as mentioned in Section 3.2, is disregarded in paper III.

Based on the facts about the data source in the paragraph above, only proper data normalization has to be applied to the RTF data sets. Similar to the data pre-processing of the C-MAPSS data set in Section 3.1, the z-score normalization method is used on the NOP data of each RTF data set. Then, the obtained normalization statistics are applied to the FD data. Sensor measurements of maritime components might be subjected to random amounts of noise when operated on ships. Thus, to increase the complexity of each NOP data set and create differentiated real-life maritime situations, different magnitudes of random white Gaussian noise, g, is added to each normalized sensor measurement at each time step t:

$$P_{signal} = \frac{1}{T_{NOP}} \sum_{t=1}^{T_{NOP}} \left(\sqrt{\frac{1}{n}\left(\hat{x}_1^2 + \cdots + \hat{x}_n^2\right)} \right) \tag{4.1}$$

$$P_{noise} = \frac{1}{T_{NOP}} \sum_{t=1}^{T_{NOP}} \left(\sqrt{\frac{1}{n}\left((\hat{x}_1 + g)^2 + \cdots + (\hat{x}_n + g)^2\right)} \right) \tag{4.2}$$

where T_{NOP} is the number of time steps in NOP data, $\hat{x}_n$ is the normalized measurement of sensor n, and P_{signal} and P_{noise} are the average power of the signal and the noise in

the NOP data, respectively. Then, the SNR can be defined as follows:

$$SNR(\%) = \frac{P_{signal}}{P_{noise}} \cdot 100 \tag{4.3}$$

Four different real-life situations are created by applying 100%, 90%, 80%, and 70% SNR to each of the five NOP data sets.

4.2 Fault diagnostics

The five RTF data sets do not include normal and fault labels. For example, a target column including “0" for normal data points, “1" for fault A data points, and “2" for fault B data points. In terms of fault diagnostics, the supervised learning principal involves training a supervised binary classifier, or multi-class classifier in this case, to differentiate between normal and fault labels in the target column. However, such fault labels are extremely rare to come by in the maritime industry. NOP data, on the other hand, is more easy to collect and define. Such data can be accumulated through the vast amount of sensors installed on both conventional ships and future autoships. When only NOP data is available, an unsupervised reconstruction-based training principal can be utilized to detect faults and create associated fault labels automatically for fault classification purposes. This is where the proposed unsupervised reconstruction-based fault detection algorithm for maritime components in paper III come in handy.

4.2.1 The initial development of the fault detection algorithm

In unsupervised reconstruction-based fault detection algorithms, also referred to as spectral anomaly detection in the literature [76, 77], the idea is to produce the lower dimensional embedding of the input data where NOP data and FD data are generally distinct. DNNs are normally used for this purpose as they allow dimension reduction through several hidden layers with non-linear transformations. First, a DNN is trained to reconstruct NOP data. This is done, in an unsupervised manner, such that the input data is also used as the target data for reconstruction. The DNN is trained in this manner until it provides a satisfying low reconstruction error. In other words, the compressed version of the input data supports the low dimensional reconstructions to extract information relevant to NOP data. Then, when FD data is fed to the trained DNN, it will output a larger reconstruction error since it cannot reconstruct the unexpected degradation patterns. At each time step between the input data and the low dimensional reconstructions, this reconstruction error is then used as an AS to detect faults. The AS is calculated by the MSE over all sensor measurements.

The AS needs a criterion on how to detect a fault. One approach is to set user-specified threshold values [88]. However, maritime components are subjected to harsh environmental conditions with varying operational loads, and hence, application-dependent threshold values are not suitable. In paper III, the proposed algorithm detects a fault automatically by estimating the time step with the maximum acceleration a_{max} in the AS. This is done by the sliding window operation, as seen in Figure 4.1. See appendix C for a complete description. a_{max} is chosen as the fault criterion to detect the fault time step $\hat{f}_t$ since this point indicates increasing velocity, and hence, a rapid increase in the AS. This increase in velocity indicates that one or several sensor measurements have started to deviate from NOP data. Due to latency in physical components, a_{max} is

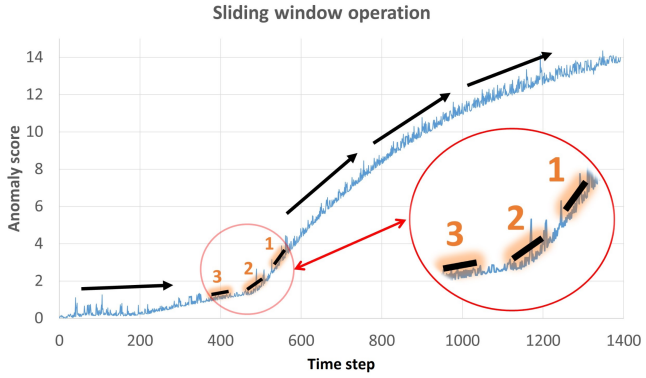


Figure 4.1: Illustration of the sliding window operation. Three windows (highlighted in orange) slide across AS through time [19].

a valid indication of a fault because there is an expected time delay before the fault will result in large sensor measurements deviations. Simply put, a_{max} provides early warnings.

During the experiments in paper III, three popular DNNs, AE, VAE, and LSTM, in addition to an FNN with one hidden layer (1FNN), are used with the proposed algorithm. All experiments are run five times and the average $\hat{f}_t$ is shown in Table 4.1. NOP data subjected to noise can be considered to be a regularization technique for DNNs. This is illustrated in Table 4.1 where the three DNNs provide consistent prediction performance along with reduced SNR. This is because DNNs do dimension reduction through several hidden layers to provide more abstract and robust features. Therefore, the low dimensional embedding is somewhat forced to filter the noise and generalize on the actual statistics in the NOP data. This is not the case, however, for the FNN with only one hidden layer. Consequently, it learns the statistics of noise, which disturb its capability to detect the fault in FD data. As seen in Table 4.1, reduced SNR strongly influences the 1FFN.

To further evaluate the proposed algorithm and select the best performing DNN, accuracy evaluations on the FD data in the four real-life situations are shown in Tables 4.2, 4.3, 4.4, and 4.5. The accuracy is defined as follows:

$$Acc\,(\%) = \left(1 - \frac{||\hat{f}_t - f_t||}{T_{FD}}\right) \cdot 100 \tag{4.4}$$

The accuracy evaluations can be considered to be indications of the distance between $\hat{f}_t$ and f_t. However, early and late predictions are not taken into account in this performance measurement, which would be of importance for a real-life data-driven PHM system for both conventional ships and autoships. All DNNs confirm robustness towards noisy real operation input data, but the VAE provides a slightly better overall accuracy performance compared to both AE and LSTM, as indicated in bold. Therefore, the VAE is the favored DNN to be used with the proposed algorithm when further explored and improved in papers IV, V, and VI.

When using a_{max} as the fault criterion, only offline fault detection is possible concerning a real-life data-driven PHM system. This is because one would need the FD data in advance to determine a_{max}. The utilization of thresholds based on the acceleration calculations can enable online fault detection. For maritime components, however, different fault-types might be subjected to different degradation patterns. Thus, dynamic and generic thresholds have to be created. These concerns are explored in the main case study of this dissertation in Chapter 5 in Section 5.2.

Table 4.1: Predicted fault time step $\hat{f}_t$ compared to true fault time step f_t on FD data [19].

Data set	Fault-type	T_{FD}	f_t	SNR(%)	$\hat{f}_t$			
					1FNN	AE	VAE	LSTM
1	A	665	157	100	148	151	154	158
				90	367	151	153	158
				80	412	154	152	158
				70	476	154	153	158
2	B	682	148	100	**148**	146	**148**	155
				90	152	150	147	**148**
				80	381	150	146	150
				70	463	149	146	161
3	B	1394	477	100	492	455	**477**	481
				90	480	479	479	481
				80	632	479	479	481
				70	791	481	482	481
4	A	1916	1306	100	1281	1281	1281	1281
				90	1278	1281	1281	1281
				80	1280	1282	1281	1281
				70	1282	1281	1281	1281
5	B	2732	787	100	807	752	807	732
				90	866	655	783	739
				80	932	728	796	740
				70	1043	732	800	742

100% SNR	Acc (%)			
Data set	1FNN	AE	VAE	LSTM
1	99.647	99.098	99.549	99.850
2	100	99.707	100	98.974
3	98.924	98.422	100	99.713
4	98.695	98.695	98.695	98.695
5	99.268	98.719	99.268	97.987
Avg. Acc	99.107	98.928	**99.502**	99.044

Table 4.2: Accuracy evaluation on FD data with 100% SNR applied to NOP data [19].

90% SNR	Acc (%)			
Data set	1FNN	AE	VAE	LSTM
1	68.421	99.098	99.398	99.850
2	99.413	99.707	99.853	100
3	99.785	99.857	99.857	99.713
4	98.434	98.695	98.695	98.695
5	97.108	95.168	99.854	98.243
Avg. Acc	92.632	98.505	**99.531**	99.300

Table 4.3: Accuracy evaluation on FD data with 90% SNR applied to NOP data [19].

80% SNR	Acc (%)			
Data set	1FNN	AE	VAE	LSTM
1	61.654	99.549	99.248	99.850
2	65.839	99.707	99.707	99.707
3	88.881	99.856	99.857	99.713
4	98.643	98.695	98.695	98.695
5	94.693	97.840	99.671	98.280
Avg. Acc	81.941	99.130	**99.435**	99.249

Table 4.4: Accuracy evaluation on FD data with 80% SNR applied to NOP data [19].

70% SNR	Acc (%)			
Data set	1FNN	AE	VAE	LSTM
1	52.030	99.549	99.399	99.850
2	53.812	99.853	99.707	98.094
3	77.475	99.713	99.641	99.713
4	98.747	98.695	98.695	98.695
5	90.629	97.987	99.524	98.353
Avg. Acc	74.539	99.159	**99.393**	98.941

Table 4.5: Accuracy evaluation on FD data with 70% SNR applied to NOP data [19].

5 Case study: marine diesel engines in autonomous ferries

This chapter presents the research findings and vital discussions concerning data pre-processing, fault diagnostics, and fault prognostics of the third data source in this dissertation. This data source is considered to be the main case study as it aims towards autoships, and more specifically, marine diesel engines in autonomous ferries. Today, enthusiasm for ship autonomy is flourishing in the maritime industry. Thus, such a case study is important to present convincing research for the industry in general. This chapter is divided into three main sections: data pre-processing in Section 5.1, fault diagnostics in Section 5.2, and fault prognostics in Section 5.3. Supplementary content related to this chapter is found in papers V, VI, and VII in appendix E, F, and G, respectively.

5.1 Data pre-processing

The marine diesel engine is considered to be the most critical component onboard autonomous ferries since it generates power for propulsion and auxiliary equipment. On the open sea, it is subjected to rapid variations in operational loads, which depends on both the task of operation and the harsh environment. In such complexity, the sensor measurements are highly connected to the operational loads. Therefore, a multi-regime normalization method has to be applied on the raw input data to present both normal operations and degradation patterns for DNNs [34]. As mentioned in Section 2.4, the raw data were first reduced to 47 time-variant features. However, features belonging to the battery system and the automation system are irrelevant for the component in focus. Thus, a feature selection process is necessary to consider for the fault detection task.

5.1.1 Feature selection

The research presented in paper V shows that feature selection improved the reconstruction process of the fault detection algorithm. Therefore, feature selection is also used in paper VI. Table 5.1 presents the selected features for the marine diesel engine. These are selected by first removing features with constant measurements because they actually provide no degradation information. Then, a Pearson correlation analysis is used to detect linear relationships between features. If two features have a high linear relationship, they likely contain redundant information. HDK is then used to determine which of the redundant features to keep and remove. In the end, nine features were considered relevant for the marine diesel engine.

Table 5.1: Feature selection for the marine diesel engine [5].

Index	Description	Unit
1	Boost pressure	bar
2	Engine load	kW
3	Engine cooling water temperature	°C
4	Engine exhaust gas temperature	°C
5	Cooling water temperature out of the engine	°C
6	Engine speed	rpm
7	Diesel generator cooling water flow	liter/min
8	Simulated propulsion load	kW
9	Cooling fan speed controller	rpm

5.1.2 Multi-regime normalization

As seen in Figure 2.7, it is obvious that both profiles fall into five distinct operating conditions based on the engine load. Fault patterns of typical fault-types associated with the marine diesel engine are highly connected to the operating conditions. In other words, the fault patterns can only be detected within a context, that is, a specific operating condition. The fault detection algorithm, however, is only able to detect single anomalous values if they differ from previous values. So, to present the fault patterns for the fault detection algorithm, multi-regime normalization has to be performed to merge the five operating conditions into one context. First, the NOP data sets in Table 2.4 are split into five data sets each based on the five operating conditions. Each feature in these data sets is then scaled with z-score normalization:

$$\overline{x}_n^o = \frac{x_n^o - \mu^o}{\sigma^o} \tag{5.1}$$

where x_n is the input feature, $n = 1, 2, ..., 9$, in operating condition $o = 1, 2, ..., 5$, and μ and σ is the population mean and population standard deviation of that feature. This results in five different normalization statistics, one for each operating condition [5]. To be able to train the fault detection algorithm on NOP data and to detect faults in FD data, these normalization statistics are then applied to the two NOP data sets and the five FD data sets in Table 2.4. To apply different normalization statistics, the engine load is monitored at each time step.

It is worth noting that the engine loads were divided into five operating conditions manually in paper VI. However, if new operating conditions are encountered in real-life data-driven PHM systems in autonomous ferries, which is likely to happen, this process has to be automated. This can be done, for instance, through unsupervised clustering algorithms, such as the K-Means algorithm.

5.2 Fault diagnostics

The proposed fault detection algorithm in paper III was the first attempt to answer R03 in Section 1.2 in this dissertation. However, as mentioned in Section 4.2, the algorithm utilized the maximum acceleration as the fault criterion, which only provided offline fault detection. Regarding the marine diesel engine in autonomous ferries, fault-types associated with the engine are subjected to different degradation patterns. To provide online fault detection in a real-life data-driven PHM system, one solution would be to create specific threshold limits for each fault-type. Alternatively, one could create

dynamic and generic threshold limits to consider all fault-types to make the algorithm fault-type independent, as stated in RO3. The main contributions of paper VI, to further improve the fault detection algorithm, are online and fault-independent fault detection by utilizing dynamic and generic threshold limits.

The learning framework is renamed from unsupervised in papers III, IV, and V to semi-supervised in paper VI. Also, the term reconstruction-based is renamed to spectral anomaly detection. This was done to follow the correct terminology used in recent anomaly detection studies in the literature [30, 76, 77]. It is worthwhile to be aware that semi-supervised learning has another meaning in fault prognostics, namely, the combination of unsupervised pre-training and supervised fine-tuning, as performed in paper II.

5.2.1 Dynamic and generic threshold limits

As mentioned in Section 4.2, the AS needs a criterion guiding the detection of a fault. Both AS or a smooth version of it can, of course, be used in addition to a threshold limit as the fault predictors. However, such predictors will vary between different fault-types since they reflect the degradation patterns. This contradicts the fact that the main goal of the improved fault detection algorithm in paper VI is to be fault-type independent. However, both velocity and acceleration calculations of the AS are considered to be more suitable fault predictors for the algorithm since such calculations are assumed to be similar between different fault-types. Thus, dynamic and generic threshold limits can be constructed.

In paper VI, the threshold limits are based on velocity v_n and acceleration a_n calculations of the smooth AS of NOP data for both engine load profiles. As seen in Figure 4.1, three sliding windows of length w determines the amount of smoothing performed on AS, $w = T/p$, where T is the total number of time steps in NOP data, as shown in Table 2.4, and p is an adjustable parameter. Careful tuning of p is necessary since excessive smoothing might obscure important data trends. Seven different p values, in the 30 to 90 range, are therefore used during the experiments in paper VI. To obtain the threshold limits, the minimum and maximum velocities of v_n, v_{min}, and v_{max}, and the minimum and maximum accelerations of a_n, a_{min} and a_{max}, are calculated for each p value in both profile 1 and profile 2. Then, a common set of upper and lower thresholds for both v_n and a_n are calculated based on the following experience-based formulas:

$$v_{upper} = \frac{|(v_{max,1} + v_{max,2}) - (v_{min,1} + v_{min,2})|}{2} \tag{5.2}$$

$$v_{lower} = -v_{upper} \tag{5.3}$$

$$a_{upper} = \frac{|(a_{max,1} + a_{max,2}) - (a_{min,1} + a_{min,2})|}{2} \tag{5.4}$$

$$a_{lower} = -a_{upper} \tag{5.5}$$

The common set of upper and lower thresholds for each p value are shown in Table 5.2. The upper and lower thresholds are added to v_n and a_n for each time step to construct the threshold limits used as fault criteria for FD data. When velocity v_d and acceleration

Table 5.2: A common set of upper and lower thresholds for both the velocity and the acceleration [5].

p	v_{lower}	v_{upper}	a_{lower}	a_{upper}
30	-2.63	2.63	-4.10	4.10
40	-3.40	3.40	-4.97	4.97
50	-3.81	3.81	-6.35	6.35
60	-4.40	4.40	-7.26	7.26
70	-5.10	5.10	-8.58	8.58
80	-5.74	5.74	-9.67	9.67
90	-6.32	6.32	-10.54	10.54

a_d calculations of FD data exceeds their respective limits, the velocity fault time step $\hat{f}_{t,v}$ and the acceleration fault time step $\hat{f}_{t,a}$ are detected. The complete procedure of how to construct the dynamic and generic threshold limits are elaborated in Algorithm 1 in appendix F.

5.2.2 Online fault detection

The air filter fault and the turbo fault in both profiles are used for validation in paper VI. This validation aims to discover the best performing fault predictor out of velocity and acceleration calculations in addition to the most suitable p value for the threshold limits. To validate both $\hat{f}_{t,v}$ and $\hat{f}_{t,a}$, the true fault time step f_t has to be determined. Since both faults-types where provoked gradually during the experiments, f_t could not be determined based on a recorded time step. The boost pressure, as seen in Table 5.1, is the key feature to monitor for both fault-types. In addition, both faults-types are highly connected to the engine loads and subjected to different degradation patterns. Therefore, f_t is determined where the deviation in boost pressure between NOP data and FD data is largest.

Table 5.3 shows $\hat{f}_{t,v}$ and $\hat{f}_{t,a}$ for each p value in both profiles for both fault-types. The accuracy evaluations, Acc_v and Acc_a, are equal to Eq. 4.4 in Section 4.2. Table 5.4 shows the average accuracy for each p value. When $p = 60$, the acceleration provides the highest average accuracy of 97.61%. Note that this is the only configuration that results in a satisfactory accuracy. This highly reflects the difficulty of constructing dynamic and generic threshold limits for two different fault-types associated with the marine diesel engine subjected to different environmental conditions, in terms of different engine load profiles, in autonomous ferries. Figure 5.1 illustrates the acceleration calculations and the corresponding threshold limits when $p = 60$ for both fault-types in both profiles. It is worth mentioning that the acceleration calculations and the threshold limits are not plotted before the entire sliding window operation is active. In other words, the initial 195 time steps are plotted as zeros ($w(60) \cdot 5 = 195$). As a consequence, such fault detections would have a time delay of 195 time steps, if applied in a real-life data-driven PHM system.

The cooling fault is subjected to a different degradation pattern compared to the two fault-types used for validation. Hence, it can be considered to be new field data that the algorithm has never seen before. The best algorithm configurations, as discovered in the validation, are further used for the cooling fault in profile 1 as the final test of the improved fault detection algorithm in paper VI. To evaluate the prediction, f_t

Table 5.3: The true fault time step f_t compared to the predicted fault time step $\hat{f}_t$ for air filter and turbo degradation data [5].

Fault-type	Profile	f_t	p	w	$\hat{f}_{t,v}$	$Acc_v(\%)$	$\hat{f}_{t,a}$	$Acc_a(\%)$
Air filter	1	1670	30	78	1255	82.31	1502	92.84
			40	58	1278	83.29	1609	97.40
			50	46	1289	83.76	1648	99.06
			60	39	1549	94.84	1660	99.57
			70	33	1566	95.57	1674	99.83
			80	29	1706	98.47	1680	99.57
			90	26	1709	98.34	1682	99.49
	2	1433	30	78	1362	96.97	1428	99.79
			40	58	1392	98.25	1445	99.49
			50	46	1404	98.76	1458	98.93
			60	39	1532	95.78	1483	97.87
			70	33	1540	95.44	0	38.92
			80	29	0	38.92	0	38.92
			90	26	0	38.92	0	38.92
Turbo	1	1431	30	78	731	70.16	693	68.54
			40	58	771	71.87	745	70.76
			50	46	786	72.51	752	71.06
			60	39	794	72.85	1347	96.42
			70	33	368	54.69	1362	97.06
			80	29	1395	98.47	1374	97.57
			90	26	1399	98.64	1381	97.87
	2	1427	30	78	951	79.71	892	77.20
			40	58	979	80.90	929	78.77
			50	46	991	81.42	1329	95.82
			60	39	1005	82.01	1347	96.59
			70	33	1387	98.29	1361	97.19
			80	29	1393	98.55	1371	97.61
			90	26	1397	98.72	1378	97.91

Table 5.4: The average accuracy for each p value [5].

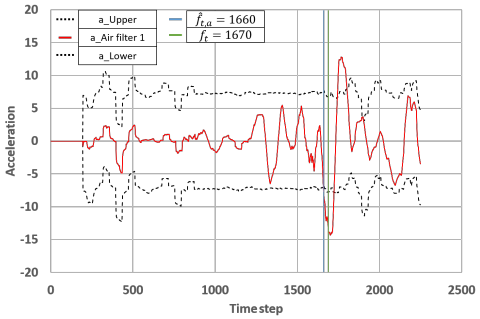
p	w	Avg. $Acc_v(\%)$	Avg. $Acc_a(\%)$
30	78	82.29	84.59
40	58	83.58	86.60
50	46	84.11	91.22
60	39	86.37	**97.61**
70	33	86.00	83.25
80	29	83.60	83.42
90	26	83.65	83.55

Table 5.5: The true fault time step f_t compared to the predicted fault time step $\hat{f}_t$ for cooling degradation data [5].

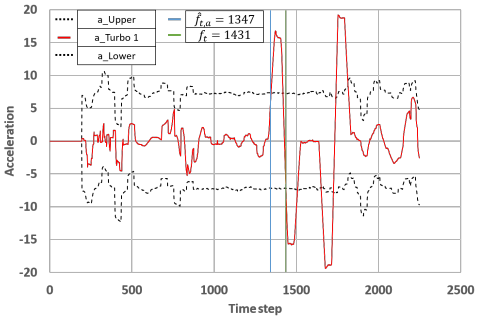
Fault-type	Profile	f_t	p	w	$\hat{f}_{t,a}$	$Acc_a(\%)$
Cooling	1	1713	60	39	1658	97.66

for the cooling fault is also chosen based on expert HDK. When the engine cooling water temperature, index 3 in Table 5.1, increases 85 °C, f_t is determined to be 1713. Table 5.5 shows that the algorithm predicts the cooling fault with an accuracy of 97.66%. Note that both in the validation and the final test the trend is that the acceleration provides early predictions, i.e. $\hat{f}_{t,a} < f_t$, when $p = 60$. However, early predictions with a corresponding high accuracy are considered as valid predictions since there is

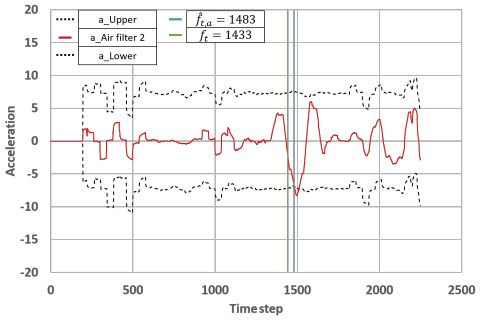
an expected time delay in the marine diesel engine before the faults will result in large sensor measurements deviations. Figure 5.2 shows the acceleration calculations and the corresponding threshold limits for the fault detection of the cooling degradation data.



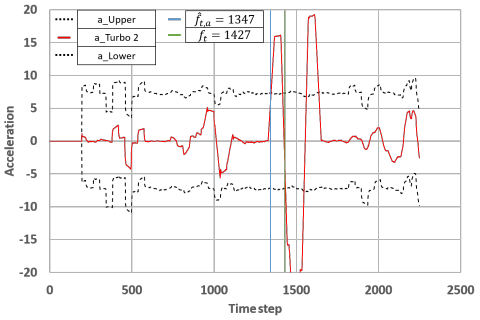
(a) Air filter fault in engine load profile 1.



(b) Turbo fault in engine load profile 1.



(c) Air filter fault in engine load profile 2.



(d) Turbo fault in engine load profile 2.

Figure 5.1: Online fault detection where $p = 60$ and the acceleration is used as the fault predictor for air filter and turbo degradation data [5].

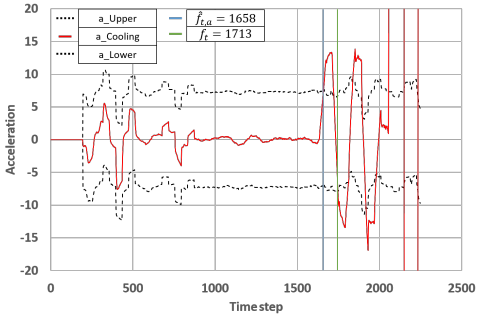


Figure 5.2: Online fault detection where $p = 60$ and the acceleration is used as the fault predictor for cooling degradation data [5].

The final test proves that the algorithm can be considered to be fault-type independent.

The resulting $\hat{f}_{t,a}$ obtained from paper VI are further used in paper VII to construct reliable RTF targets to do fault prognostics. This is further elaborated in the Section 5.3.

5.3 Fault prognostics

In the future, autonomous ferries are likely to be maintained, navigated, and operated without any crew involvement. Thus, accurate and reliable predictions of the progression of already detected fault-types, in terms of the RUL, are essential for such ferries. Such RUL predictions enable the optimization of maintenance schedules such that maintenance occurs at the ferry's next appropriate port of call. This increases both profitability and safety.

It is more difficult to achieve accurate and reliable RUL predictions based on real operation data than on the C-MAPSS data set, as done in papers II and IV. This is because real operation data is unstructured and lacks both RTF targets and a stand-alone test set from the start. However, the knowledge learned from benchmark data is valuable to transfer in order to do fault prognostics of the marine diesel engine in autonomous ferries. For example, both the idea of and the knowledge learned from the proposed DNNs in both paper II and IV are valuable, the PwL degradation model is directly transferred from paper IV to paper VII, and the experience gained from hyper-parameter tuning is beneficial.

5.3.1 Introducing the SkipRnet

Operational sensor data collected from the marine diesel engine in autonomous ferries will primarily involve time-series data. As seen in Figure 3.1 in Section 3.3, the number of hidden layers in the two proposed DNNs for the C-MAPSS data set is fixed. In other words, those DNNs utilize static structures. However, static DNNs have difficulty generalizing on real operational time-series data because the degree of complexity, in terms of both sensor noise and various operating conditions, might differ between training data and new field data. Static DNNs with few hidden layers and corresponding few parameters are only able to model time-series data with low complexity and vice versa. However, the utilization of skip connections, as successfully applied for image data in [37], enable dynamic DNNs, that is, the opportunity to automatically train different layers for different time steps. In this way, such DNNs will be trained at different rates based on how the error flows backward in different paths. Therefore, they should be able to handle time-series data in a wide range of complexities.

The proposed SkipRnet in paper VII is shown in Figure 5.3. As in papers II and IV, LSTMs and FNNs will act as the main building blocks. The LSTM layers are used to learn temporal and long-term dependencies within the features of degradation data. The FNN layers are then used to map all extracted features before a dropout layer is used to reduce overfitting. To keep it simple, both the RBM layer and the 1D CNN layer are dropped. By utilizing skip connections, however, the SkipRnet has the ability to skip both the second LSTM layer and the second FNN layer during the training procedure. This results in four different paths with differing numbers of parameters. In other words, the SkipRnet can be considered as an accumulation of four different DNNs. When the SkipRnet is trained and employed to predict the RUL on new field data, it has the potential to make use of the four paths with different numbers of parameters.

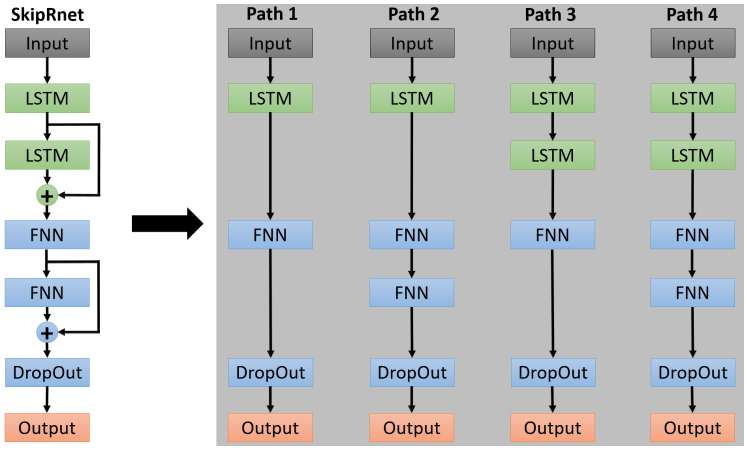


Figure 5.3: The SkipRnet and its four different paths [14].

The idea is that these path alternatives will handle the various operating conditions the marine diesel engine confronts. Thus, the SkipRnet is proposed to increase the reliability of data-driven fault prognostics, as stated in R04 in Section 1.2.

5.3.2 RTF targets for supervised training

As discovered in paper IV, the PwL degradation model, when incorporating diagnostics information, was the best performing and most convenient data-driven labeling approach. The time step where the degradation starts is essential information for this approach to construct reliable RTF targets for the SkipRnet. Therefore, the already detected fault time steps, as obtained in paper VI and described in Section 5.2, are directly used to construct RTF targets for air filter degradation data and turbo degradation data in both engine load profiles in paper VII. In this way, the SkipRnet can be trained with supervision since each time step in the data sets has a target value to map during the training process. As a consequence, both feature selection and multi-regime normalization will have a relatively low impact on the input-target mappings. By supervised training, DNNs are strong enough to both filter unnecessary features and cope with the complexity inherent in the different engine loads. Therefore, all 47 input features and conventional z-score normalization are used in paper VII.

5.3.3 Data split and data augmentation

High generalization power towards engine load profiles that the SkipRnet has never seen before is extremely important if the SkipRnet is to be employed in future data-driven PHM systems for autonomous ferries to provide real-time RUL predictions. Only the air filter and turbo degradation data in profile 1 are used as the training set for the SkipRnet. The degradation in the training set grows in magnitude until failure. Consequently, the last RUL target $= 0$. Profile 2 is subjected to different engine loads, and hence, it will be used as the test set. Therefore, the degradation in the test set has to end sometime prior to failure in order to verify that the SkipRnet is able to generalize to predict the

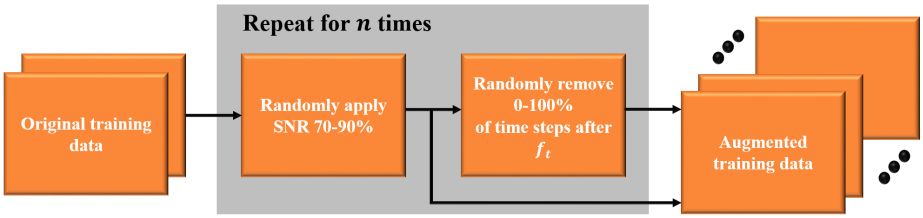


Figure 5.4: The proposed data augmentation technique for RTF time-series data [14].

RUL. Accordingly, a random interval of time steps before failure is removed in both the air filter and the turbo degradation data in profile 2. Table 5.6 summarizes the data split used in paper VII to do fault prognostics.

Problems accessing large amounts of RTF data are common in the maritime industry. This is unfortunate given that DNNs require large amounts of RTF data to do fault prognostics with satisfactory accuracy. In addition, a limitation of DNNs today is the danger that they will learn only exactly what we ask them to learn during supervised learning. An example is that the SkipRnet will only be trained on RTF data from profile 1. Thus, the danger is that the SkipRnet will only learn statistics from profile 1 and will not be able to generalize to profile 2. Therefore, a novel data augmentation technique for RTF time-series data is proposed in paper VII to increase the generalization power of the SkipRnet.

The aim of the proposed technique is to answer RO4, as stated in Section 1.2, to increase the reliability of fault prognostics. As seen in Figure 5.4, an SNR between 70 and 90% is first applied to the normalized original training set. The noise is applied, similar to Eq. 4.1, 4.2, and 4.3 in Section 4.1. The idea is that the resulting noisy data set will exhibit similar statistics to profile 1, but differ based on the SNR. This will increase the range of statistics that the SkipRnet will learn during the training procedure. Next, similar to [89], a random interval of time steps, in the range between 0-100% after f_t, is removed to also include some time-series that will end some time before failure. Thus, the SkipRnet is forced to learn distributions that are more similar to a real-life PHM system, where the actual goal is to predict the available time before operational failure. In paper VII, the proposed technique is repeated for 0, 10, 20, 30, 40, and 50 times for each fault-type in the training set. This results in six different scenarios of 0, 20, 40, 60, 80, and 100 augmented training data sets.

Table 5.6: Data split to do fault prognostics for the marine diesel engine [14].

Data set	Profile	Usage	f_t	Last RUL target	Time steps
Air filter degradation	1	Train/cross-val	1,660	0	2,346
Turbo degradation	1	Train/cross-val	1,347	0	2,346
Air filter degradation	2	Test	1,483	106	2,240
Turbo degradation	2	Test	1,347	490	1,856

5.3.4 Hyper-parameters and k-fold cross-validation

The bullet-points in Section 3.3, which is learned based on the experience from the C-MAPSS data set, are followed to select most of the hyper-parameters of the SkipRnet. However, the number of hidden units in each hidden layer, which relates to the total number of parameters in terms of weights and biases, is tuned through cross-validation. As opposed to the C-MAPSS data set, the data sets in Table 5.6 incorporate a relatively small number of examples. Thus, if a random portion is selected from the training sets to act as cross-validation, it might exhibit different statistics than the training sets. Consequently, by reducing the training data, important degradation patterns might be lost, which in turn increases error-induced bias. In this case, k-fold cross-validation is necessary. In paper VII, the six different training data scenarios were divided into seven folds or subsets. In other words, hold-out cross-validation with an 80% training and 20% cross-validation split was repeated seven times. Each time, one of the seven subsets is used as the cross-validation set and the remaining six subsets are used as the training set. The error, in terms of the RMSE on the cross-validation set, is estimated as the averaged of the seven trials.

The goal of the cross-validation is to acquire the most robust configuration of the SkipRnet. That is, to achieve the configuration that best reflects the degree of complexity in the cross-validation sets in all six scenarios. The first scenario includes zero augmented data sets, and hence, the SkipRnet is only trained and validated on the original training set. Thus, the first scenario is assumed to exhibit the lowest degree of complexity of the six scenarios. In contrast, the scenario with 100 augmented data sets is assumed to exhibit the highest degree of complexity. In paper VII, the SkipRnet with 128 hidden units in each hidden layer provides the lowest average cross-validation RMSE for all augmented data sets scenarios. Therefore, this configuration is further used to predict the RUL on profile 2.

5.3.5 Remaining useful life predictions for the marine diesel engine

In paper VII, the air filter and turbo degradation data in profile 2 are used as the stand-alone test set to verify the generalization power of the SkipRnet. The trained SkipRnet and Paths 1-4, which are used as baseline DNNs, are employed to predict the RUL at each time step. This can be considered a real-time test since this is how DNNs would potentially be employed in an actual data-driven PHM system for autonomous ferries. Similar to Eq. 3.1 and 3.2 in Section 3.3, both the RMSE and the scoring function (S) are used as the performance indicators:

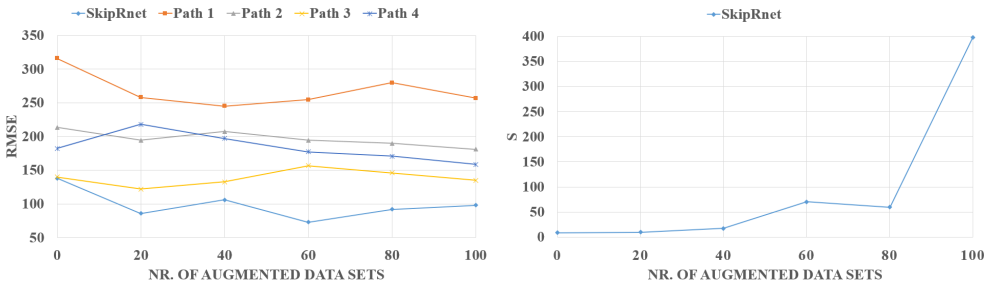
$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} d_i^2} \tag{5.6}$$

$$\text{S} = \begin{cases} \sum_{j=1}^{m} e^{-(\frac{d_j}{13})} - 1, & \text{for} \quad d_j < 0 \\ \sum_{j=1}^{m} e^{(\frac{d_j}{10})} - 1, & \text{for} \quad d_j \geq 0 \end{cases} \tag{5.7}$$

where n is the total number of constructed RUL targets, $d_i = RUL_{predicted,i} - RUL_{target,i}$, m is the total number of last RUL targets, and $d_j = RUL_{predicted,j} - RUL_{target\,last,j}$.

Figure 5.5a shows the RMSE on the test set when the SkipRnet and the four paths are trained on each augmented data set scenario. As expected, Path 1 provides the worst overall RMSE due to having the lowest number of parameters (122,753). Interestingly, Path 4 provides worse overall RMSE compared to the SkipRnet, even though Path 4 has the same number of parameters (270,849). A logical explanation for these findings is the advantage of the skip connections. For each time step in the test set, the SkipRnet has the ability to utilize the strengths and reduce the weaknesses of four DNNs. In other words, for each time step, the SkipRnet has the ability to utilize different numbers of parameters in the range between 122,753 and 270,849. Therefore, the SkipRnet is able to handle a wider range of complexities in new field data compared to DNNs without skip connections.

S is also important to consider in real-life data-driven PHM systems suitable for autonomous ferries. A reliable and low S performance close to the end of the marine diesel engine's lifetime has great significance, as the scheduling of maintenance operations in this period is critical. As seen in Figure 5.5b, the SkipRnet provides satisfactory S performance on the test set when trained on 0, 20, and 40 augmented data sets. However, when also considering the RMSE in Figure 5.5a, the SkipRnet provides the best overall RUL performance on the test set when trained on 20 augmented data sets. Therefore, Figure 5.6 compares the RUL predictions on the air filter fault and turbo fault in the test



(a) RMSE on the test set.

(b) S on the test set.

Figure 5.5: RUL performance evaluations on the test set [14].

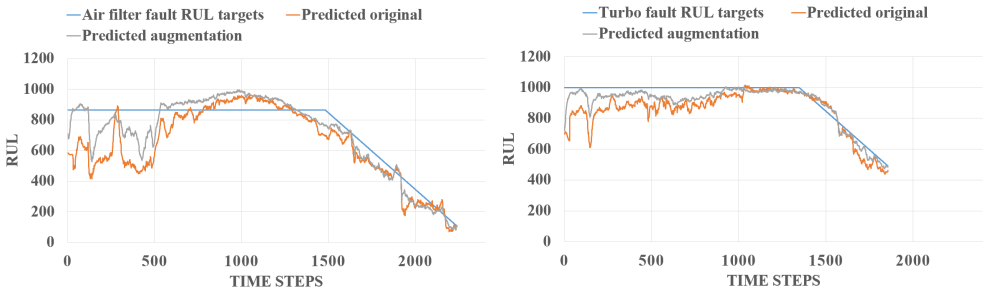


Figure 5.6: The prediction results on the air filter fault and the turbo fault in the test set when the SkipRnet is trained on 20 augmented data sets and the original training set [14].

set when the SkipRnet is trained on 20 augmented data sets and the original training set.

This comparison proves the advantage of the proposed data augmentation technique. It clearly increases the generalization power of the SkipRnet toward an engine profile it has never seen before. The predictions are kind of noisy, but this is expected due to the drastic changes in engine loads. As a consequence, confidence bounds have to be incorporated to increase the reliability of the RUL predictions in a real-life data-driven PHM system. Maintenance decisions based on prognostics information should be anchored in confidence bounds rather than a particular RUL prediction [41].

Based on the findings in paper VII, the more data you feed to DNNs, constructed for fault prognostics, the better they become at providing RUL predictions in new field data. Therefore, it is recommended that both component manufacturers and shipowners start saving and sharing their RTF data in order to gain the true benefits of data-driven PHM systems.

6 Conclusion

This dissertation has proposed and discussed a data-driven PHM system for autoships. Furthermore, it has presented important ideas and research findings concerning three case studies for data-driven algorithm development for such a system. Because data-driven PHM systems are in their infancy in the maritime industry in general, this dissertation has shown the possibility of knowledge transfer from benchmark data of airplane engines to a case study, involving the marine diesel engine in autonomous ferries. Additionally, this dissertation has presented clever solutions and novel DNNs to respond to the common lack of fault and failure data in the industry.

The importance of data pre-processing, fault diagnostics, and fault prognostics have been highlighted throughout this dissertation. The resulting analysis of such actions can be used to ensure both the operational availability and safety of critical components onboard autoships. This will, in turn, lead to trustworthy, efficient, and cost-beneficial autonomous operations on the open sea. All contributions in this dissertation aim to enhance these aspects.

6.1 Summary of contributions

The proposed data-driven PHM system, as stated in RO1, can be divided into four main actions: data pre-processing, fault diagnostics, fault prognostics, and decision support or automation. DNNs have appeared as extremely powerful in such actions – if sufficient fault and failure data is available. However, since both data-driven PHM systems and the utilization of DNNs have just begun to gain popularity in the maritime industry, a comprehensive literature survey was written, as stated in RO2. This survey investigates how DNNs have been applied to data-driven PHM systems in other domains in addition to present DNNs applicable to the maritime environment. In the proposed system, a fault detection algorithm, suitable for the harsh maritime environment, was developed, as stated in R03. The initial development and further improvements of this algorithm has been given high priority in this dissertation since it affects all subsequent actions. Autoships are assumed to be maintained, navigated, and operated without any crew involvement in the future. Thus, to predict the progression of already detected fault-types is essential. Such fault prognostics enables optimized maintenance schedules for the next appropriate port of call for autoships to avoid operational failure. However, fault prognostics are still under research and development. So, to increase the reliability of DNNs to do accurate and reliable fault prognostics, as stated in RO4, has been of high priority in this dissertation.

The main contributions of this thesis are as follows:

- ✓ Proposed the fundamentals of a data-driven PHM system suitable for autoships.
- ✓ Presented a comprehensive literature survey of data-driven PHM systems based on DNNs for autoships.
- ✓ Proposed and developed a fault-type independent fault detection algorithm for maritime components.
- ✓ Proposed solutions and novel DNNs to increase the reliability of fault prognostics.

6.2 Summary of publications

Paper I introduces and reviews four well-established DNNs recently applied to different practical PHM problems. The purpose of this paper is to support creativity and provide inspiration for PHM systems based on DNNs in autoships and the maritime industry. Furthermore, this paper discusses benefits, challenges, suggestions, existing problems, and future research opportunities with respect to this significant new technology. In this paper, the C-MAPSS data set is found to be the most-used benchmark data set for data-driven fault prognostics, and therefore this data set is further investigated during the experiments in paper II.

Paper II investigates the effect of unsupervised pre-training in RUL predictions utilizing a semi-supervised DNN structure. Additionally, a GA approach is applied to tune a selected search space of hyper-parameters in the training procedure. The advantages of the proposed semi-supervised setup are verified on the C-MAPSS data set. The experimental study compares this approach to purely supervised training, both when the training data is completely labeled with RTF targets and when the amount of RTF targets is reduced, and to the most robust results in the literature. The results suggest that unsupervised pre-training is a promising feature in RUL predictions subjected to multiple operating conditions and fault modes.

Paper III develops a fault detection algorithm for maritime components, which can be further used to construct RTF targets for supervised fault prognostics automatically. Thus, this algorithm aims to respond to the challenges of paper II, that high-quality RTF targets might be both challenging and time-consuming to acquire in PHM applications, and especially in the maritime industry. In paper III, the advantages of the proposed algorithm are verified on five different RTF data sets provided by an industrial company. Each data set is subject to a fault at an unknown time step. In addition, different magnitudes of random white Gaussian noise are applied to each data set to create several real-life situations. The results suggest that the algorithm is highly suitable to be included as part of fault diagnostics in future data-driven PHM systems.

Paper IV provides the PHM community an empirical study that validates the PwL degradation model against two other data-driven labeling approaches to construct RTF targets for subset FD001 in the C-MAPSS data set. The fault detection algorithm in paper III is used to automatically constructed RTF targets, which include the actual health of each engine. A DNN structure is proposed and trained on the three different

RTF targets to predict the RUL of each engine. During the training process, the GA approach in paper II is used to tune a selected search space of hyper-parameters. The results suggest that the DNN trained on PwL RTF targets performs the best and provides the most reliable RUL prediction accuracy. This DNN also outperforms the most robust results in the literature.

Paper V provides the initial experiments for the main case study in this dissertation, namely, the marine diesel engine in autonomous ferries. In this paper, the fault detection algorithm in paper III is used to detect faults automatically in a simulated autonomous ferry crossing operation. The benefits of the algorithm are confirmed on data sets of a cooling system fault collected from a marine diesel engine included in a hybrid power lab. To support the algorithm in the demanding reconstruction process, three different feature selection processes on the input data are compared. The results suggest that the algorithm achieves the highest fault prediction accuracy when the input data is subjected to feature selection based on sensitivity analysis.

Paper VI proposes a fault-type independent spectral anomaly detection algorithm for marine diesel engine degradation in autonomous ferries. This algorithm aims to improve the fault detection algorithm in paper III. The benefits of the algorithm are verified on data sets of three fault-types where the degradation pattern differs. These fault-types are a cooling system fault, an air filter fault, and a turbocharger fault. Both NOP data and FD data are collected from a marine diesel engine, using two different engine load profiles. These profiles aim to replicate real autonomous ferry crossing operations, environmental conditions the ferry may encounter. The proposed algorithm is trained to estimate velocity and acceleration calculations of the AS. Dynamic and generic threshold limits are simultaneously established to predict the fault time step online. The algorithm achieved an accuracy of 97.66% in the final test when the acceleration was used as the fault predictor. The results suggest that the algorithm is independent of fault-types with different degradation patterns related to the marine diesel engine.

Paper VII proposes a novel data augmentation technique and the SkipRnet for fault prognostics of marine diesel engines in autonomous ferries. The advantages are verified on RTF data of an air filter fault and a turbocharger fault in two different engine load profiles the ferries may face in real life. The first profile is used for training and validation, while the second is used for real-time testing. The proposed data augmentation technique is used to construct six different augmented data set scenarios based on the first profile. The SkipRnet requires high generalization power toward the second profile since harsh and variable environmental conditions will subject the marine diesel engine to unforeseeable operating conditions. Due to the presence of skip connections, the SkipRnet functions as an accumulation of four independent DNNs. Therefore, it has the ability to tackle a wider range of complexities in new field data than DNNs without skip connections. The advantage of both data augmentation and skip connections is clearly proven throughout this paper.

6.3 Important directions for future work

This dissertation has mainly focused on data-driven algorithm development for fault detection and fault prognostics. Consequently, fault isolation, fault classification, and

decision support or automation remain to be researched and developed to complete the proposed data-driven PHM system for autoships. Thus, important directions for future work are suggested as follows:

- In addition to fault detection, both fault isolation and classification are necessary for the completion of the fault diagnostics action in a data-driven PHM system. This is because fault detection only provides information on that a fault has occurred, but it lacks information concerning which fault-type it is and which component is faulty. Thus, a separate DNN can be trained to do fault classification, that is, to predict the probability of which fault-type detected faults belongs to in the current health state. However, such DNNs for this purpose are already developed [90, 91]. In a research perspective, techniques for handling imbalanced data, such as the focal loss [69], under- and oversampling [70], and weighted loss functions [71], are suggestions for future research. This is because the minority classes, that is, data points related to fault-types, are of high importance for a data-driven PHM system. If the system miss-classifies a fault condition as a normal condition, it could lead to downtime and a potential disaster for autoships. Additionally, fault isolation is important to guide maintenance personnel to the faulty component. For this action, the VAE [92] is worth consideration. Due to its generative characteristics, it can derive reconstructions of degradation data in the latent space. Such reconstructions can then be used to analyze the underlying cause of fault-types to pinpoint the faulty component.

- The final action of the proposed data-driven PHM system is to facilitate decision support or automation to recommend or direct ideal maintenance schedules. In the years to come, a human is still expected to make the final decisions. Therefore, the final action needs to use techniques that provide transparent explanations of the outputs from both fault diagnostics and fault prognostics. XAI uses methods for visualizing, explaining and interpreting DNNs [62, 63, 65, 66], and it has just begun to gain popularity. Therefore, XAI has strong potential for data-driven PHM systems, and future research and development should explore it. XAI has the potential to provide trust in the system since it aims to provide an understanding of how outputs are being made. Additionally, confidence bounds of RUL predictions should be incorporated into the final action because maintenance recommendations and its corresponding scheduling should not be entirely based on a particular RUL prediction.

- This dissertation has shown that DNNs perform extremely well – if sufficient RTF data is available. This belies the claim by most component manufacturers that their product range never fails. This attitude has to change drastically if the true benefits of data-driven PHM systems are to be realized. Thus, both component manufacturers and shipowners need to start saving and sharing their data to build benchmark data sets for academia. Such data sets would have been advantageous in data-driven algorithm development, which in turn, will benefit both engineering research and the maritime industry.

References

[1] E. Jokioinen, "Remote and autonomous ships - the next steps: Introduction," *Rolls-Royce, Buckingham Gate, London: The Advanced Autonomous Waterborne Applications (AAWA)*, pp. 4–14, 2016.

[2] A. L. Ellefsen, V. Æsøy, S. Ushakov, and H. Zhang, "A comprehensive survey of prognostics and health management based on deep learning for autonomous ships," *IEEE Transactions on Reliability*, vol. 68, no. 2, pp. 720–740, 2019.

[3] O. Levander, "Autonomous ships on the high seas," *IEEE Spectrum*, vol. 54, no. 2, pp. 26–31, 2017.

[4] L. Kretschmann, H.-C. Burmeister, and C. Jahn, "Analyzing the economic benefit of unmanned autonomous ships: An exploratory cost-comparison between an autonomous and a conventional bulk carrier," *Research in transportation business & management*, vol. 25, pp. 76–86, 2017.

[5] A. L. Ellefsen, P. Han, X. Cheng, F. T. Holmeset, V. Æsøy, and H. Zhang, "Online fault detection in autonomous ferries: Using fault-type independent spectral anomaly detection," *IEEE Transactions on Instrumentation and Measurement*, pp. 1–1, 2020.

[6] "Fleetbroadband." https://www.inmarsat.com/service/fleetbroadband/. Accessed: 2020-05-10.

[7] K. E. Knutsen, G. Manno, and B. J. Vartdal, "Beyond condition monitoring in the maritime industry," *DNV GL Strategic Research & Inovation Position Paper*, 2014.

[8] R. Kothamasu, S. H. Huang, and W. H. VerDuin, "System health monitoring and prognostics — a review of current paradigms and practices," *The International Journal of Advanced Manufacturing Technology*, vol. 28, no. 9, pp. 1012–1024, 2006.

[9] T. M. Allen, "Us navy analysis of submarine maintenance data and the development of age and reliability profiles," *Department of the Navy SUBMEPP*, 2001.

[10] P. L. Dussault, "Creating a closed loop evironment for condition based maintenance plus (cmb+) and prognostics health management," in *2007 IEEE Autotestcon*, pp. 327–331, Sept 2007.

[11] F. Camci, G. S. Valentine, and K. Navarra, "Methodologies for integration of phm systems with maintenance data," in *2007 IEEE Aerospace Conference*, pp. 1–9, March 2007.

[12] K. M. Janasak and R. R. Beshears, "Diagnostics to prognostics - a product availability technology evolution," in *2007 Annual Reliability and Maintainability Symposium*, pp. 113–118, Jan 2007.

[13] A. L. Ellefsen, X. Cheng, F. T. Holmeset, S. Ushakov, V. Æsøy, and H. Zhang, "Automatic fault detection for marine diesel engine degradation in autonomous ferry crossing operation," in *2019 IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 2195–2200, Aug 2019.

[14] A. L. Ellefsen, V. Æsøy, and H. Zhang, "Fault prognostics in autonomous ferries: The advantage of data augmentation and skip connections," *Submitted to IEEE Transactions on Reliability*, pp. 1–1, 2020.

[15] A. K. S. 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.

[16] P. W. Kalgren, C. S. Byington, M. J. Roemer, and M. J. Watson, "Defining phm, a lexical evolution of maintenance and logistics," in *2006 IEEE Autotestcon*, pp. 353–358, Sept 2006.

[17] 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. 71563–71575, 2019.

[18] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2017.

[19] A. L. Ellefsen, E. Bjørlykhaug, V. Æsøy, and H. Zhang, "An unsupervised reconstruction-based fault detection algorithm for maritime components," *IEEE Access*, vol. 7, pp. 16101–16109, 2019.

[20] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 387–395, ACM, 2018.

[21] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1544–1551, 2018.

[22] 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.

[23] H. Miao, B. Li, C. Sun, and J. Liu, "Joint learning of degradation assessment and rul prediction for aeroengines via dual-task deep lstm networks," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 9, pp. 5023–5032, 2019.

[24] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[25] M. Längkvist, L. Karlsson, and A. Loutfi, "A review of unsupervised feature learning and deep learning for time-series modeling," *Pattern Recognition Letters*, vol. 42, pp. 11–24, 2014.

[26] J. A. P. Rubio, F. Vera-García, J. H. Grau, J. M. Cámara, and D. A. Hernandez, "Marine diesel engine failure simulator based on thermodynamic model," *Applied Thermal Engineering*, vol. 144, pp. 982–995, 2018.

[27] A. S. Zymaris, Ø. Å. Alnes, K. E. Knutsen, and N. M. Kakalis, "Towards a model-based condition assessment of complex marine machinery systems using systems engineering," in *Proc. 3rd Eur. Conf. Prognostics Health Manage. Soc.*, pp. 1–15, 2016.

[28] J. Yu, "A selective deep stacked denoising autoencoders ensemble with negative correlation learning for gearbox fault diagnosis," *Computers in Industry*, vol. 108, pp. 62 – 72, 2019.

[29] R. Chen, X. Huang, L. Yang, X. Xu, X. Zhang, and Y. Zhang, "Intelligent fault diagnosis method of planetary gearboxes based on convolution neural network and discrete wavelet transform," *Computers in Industry*, vol. 106, pp. 48 – 59, 2019.

[30] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

[31] N. Jones, "Computer science: The learning machines," *Nature*, vol. 505, pp. 146–148, 01 2014.

[32] 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.

[33] 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*, pp. 1–9, IEEE, Oct 2008.

[34] O. Bektas, J. A. Jones, S. Sankararaman, I. Roychoudhury, and K. Goebel, "A neural network filtering approach for similarity-based remaining useful life estimation," *The International Journal of Advanced Manufacturing Technology*, vol. 101, no. 1-4, pp. 87–103, 2019.

[35] G. Vachtsevanos, F. Lewis, M. Roemer, A. Hess, and B. Wu, *Fault Prognosis*, pp. 284–354. John Wiley & Sons, Inc., 2007.

[36] E. Bjørlykhaug and O. Egeland, "Vision system for quality assessment of robotic cleaning of fish processing plants using cnn," *IEEE Access*, vol. 7, pp. 71675–71685, 2019.

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

[38] "Eclipse deeplearning4j development team, deeplearning4j: Open-source distributed deep learning for the jvm," *Apache Software Foundation License 2.0, http://deeplearning4j.org*, 2020.

[39] "Hannover messe." https://www.hannovermesse.de/en/. Accessed: 2019-12-05.

[40] J. Lee, F. Wu, W. Zhao, M. Ghaffari, L. Liao, and D. Siegel, "Prognostics and health management design for rotary machinery systems—reviews, methodology and applications," *Mechanical Systems and Signal Processing*, vol. 42, no. 1, pp. 314–334, 2014.

[41] J. Z. 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.

[42] D. McDonnell, N. Balfe, S. Al-Dahidi, and G. O'Donnell, "Designing for human-centred decision support systems in phm," in *European Conference of the Prognostics and Health Management Society*, pp. 1–16, IEEE, 2014.

[43] D. J. Power, "Web-based and model-driven decision support systems: concepts and issues," *AMCIS Proceedings*, p. 387, 2000.

[44] M. Tahan, E. Tsoutsanis, M. Muhammad, and Z. A. Abdul Karim, "Performance-based health monitoring, diagnostics and prognostics for condition-based maintenance of gas turbines: A review," *Applied Energy*, vol. 198, pp. 122–144, 2017.

[45] T. Sutharssan, S. Stoyanov, C. Bailey, and C. Yin, "Prognostic and health management for engineering systems: a review of the data-driven approach and algorithms," *The Journal of Engineering*, vol. 1, no. 1, 2015.

[46] 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.

[47] G. Vachtsevanos, F. Lewis, M. Roemer, A. Hess, and B. Wu, *Fault Diagnosis*, pp. 172–283. John Wiley & Sons, Inc., 2007.

[48] M. J. Roemer, C. S. Byington, G. J. Kacprzynski, and G. Vachtsevanos, "An overview of selected prognostic technologies with application to engine health management," *ASME Paper No. GT2006-90677*, 2006.

[49] S. Yin, S. X. Ding, X. Xie, and H. Luo, "A review on basic data-driven approaches for industrial process monitoring," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 11, pp. 6418–6428, 2014.

[50] J. Liu, W. Wang, F. Ma, Y. B. Yang, and C. S. Yang, "A data-model-fusion prognostic framework for dynamic system state forecasting," *Engineering Applications of Artificial Intelligence*, vol. 25, no. 4, pp. 814–823, 2012.

[51] "Nist big data interoperability framework: Volume 1, definitions," 2015.

[52] I. Arel, D. C. Rose, and T. P. Karnowski, "Deep machine learning - a new frontier in artificial intelligence research [research frontier]," *IEEE computational intelligence magazine*, vol. 5, no. 4, pp. 13–18, 2010.

[53] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, "A survey of machine learning for big data processing," *EURASIP Journal on Advances in Signal Processing*, vol. 2016, no. 1, p. 67, 2016.

[54] "Waymo open dataset." https://waymo.com/open/. Accessed: 2019-12-06.

[55] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25* (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), pp. 1097–1105, Curran Associates, Inc., 2012.

[56] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[57] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, (Cambridge, MA, USA), pp. 3104–3112, MIT Press, 2014.

[58] D. Nikolić, "Why deep neural nets cannot ever match biological intelligence and what to do about it?," *International Journal of Automation and Computing*, vol. 14, pp. 532–541, Oct 2017.

[59] O. Niculita, O. Nwora, and Z. Skaf, "Towards design of prognostics and health management solutions for maritime assets," *Procedia CIRP*, vol. 59, pp. 122–132, 2017.

[60] B.-M. Batalden, P. Leikanger, and P. Wide, "Towards autonomous maritime operations," in *2017 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, pp. 1–6, IEEE, 2017.

[61] R. Jalonen, R. Tuominen, and M. Wahlström, "Remote and autonomous ships - the next steps: Safety and security," *Rolls-Royce, Buckingham Gate, London: The Advanced Autonomous Waterborne Applications (AAWA)*, pp. 56–73, 2016.

[62] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (xai)," *IEEE Access*, vol. 6, pp. 52138–52160, 2018.

[63] J. Choo and S. Liu, "Visual analytics for explainable deep learning," *IEEE Computer Graphics and Applications*, vol. 38, pp. 84–92, Jul 2018.

[64] D. Wiljer and Z. Hakim, "Developing an artificial intelligence–enabled health care practice: Rewiring health care professions for better care," *Journal of Medical Imaging and Radiation Sciences*, 2019.

[65] W. Samek and K.-R. Müller, *Towards Explainable Artificial Intelligence*, pp. 5–22. Cham: Springer International Publishing, 2019.

[66] I. Lage, A. Ross, S. J. Gershman, B. Kim, and F. Doshi-Velez, "Human-in-the-loop interpretability prior," in *Advances in Neural Information Processing Systems*, pp. 10159–10168, 2018.

[67] G. Manno, K. Knutsen, and B. Vartdal, "An importance measure approach to system level condition monitoring of ship machinery systems," in *Proc. 11th Int. Conf. Condition Monit. Machinery Failure Prevention Technol.*, pp. 766–780, 2014.

[68] Y. Dong, "Implementing deep learning for comprehensive aircraft icing and actuator/sensor fault detection/identification," *Engineering Applications of Artificial Intelligence*, vol. 83, pp. 28 – 44, 2019.

[69] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

[70] Z. Wu, W. Lin, B. Fu, J. Guo, Y. Ji, and M. Pecht, "A local adaptive minority selection and oversampling method for class-imbalanced fault diagnostics in industrial systems," *IEEE Transactions on Reliability*, pp. 1–12, 2019.

[71] F. Jia, Y. Lei, N. Lu, and S. Xing, "Deep normalized convolutional neural network for imbalanced fault classification of machinery and its understanding via visualization," *Mechanical Systems and Signal Processing*, vol. 110, pp. 349–367, 2018.

[72] A. Saxena and K. Goebel, "Turbofan engine degradation simulation data set," *NASA Ames Prognostics Data Repository (https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/), NASA Ames Research Center, Moffett Field, CA*, 2008.

[73] X. Cheng, A. L. Ellefsen, F. T. Holmeset, G. Li, H. Zhang, and S. Chen, "A stepwise feature selection scheme for a prognostics and health management system in autonomous ferry crossing operation," in *2019 IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 1877–1882, Aug 2019.

[74] F. O. Heimes, "Recurrent neural networks for remaining useful life estimation," in *Prognostics and Health Management, 2008. PHM 2008. International Conference on*, pp. 1–6, IEEE, 2008.

[75] 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, pp. 2306–2318, Oct 2017.

[76] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009.

[77] J. An and S. Cho, "Variational autoencoder based anomaly detection using reconstruction probability," *SNU Data Mining Center - Special Lecture on IE*, vol. 2, no. 1, 2015.

[78] S. Zheng, K. Ristovski, A. Farahat, and C. Gupta, "Long short-term memory network for remaining useful life estimation," in *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pp. 88–95, IEEE, 2017.

[79] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, Jan. 2014.

[80] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[81] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, vol. 15, pp. 315–323, 2011.

[82] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256, 2010.

[83] E. Ramasso, "Investigating computational geometry for failure prognostics," *International Journal of Prognostics and Health Management*, vol. 5, no. 1, p. 005, 2014.

[84] G. S. Babu, P. Zhao, and X.-L. Li, "Deep convolutional neural network based regression approach for estimation of remaining useful life," in *International conference on database systems for advanced applications*, pp. 214–228, Springer, 2016.

[85] 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," in *2016 Workshop on Machine Learning for Prognostic and Health Management*, 2016.

[86] 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," *CoRR*, vol. abs/1709.00845, 2017.

[87] P. R. de Oliveira da Costa, A. Akçay, Y. Zhang, and U. Kaymak, "Remaining useful lifetime prediction via deep domain adaptation," *Reliability Engineering & System Safety*, vol. 195, p. 106682, 2020.

[88] A. Brandsæter, G. Manno, E. Vanem, and I. K. Glad, "An application of sensor-based anomaly detection in the maritime industry," in *2016 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pp. 1–8, IEEE, 2016.

[89] L. Jayasinghe, T. Samarasinghe, C. Yuen, J. C. N. Low, and S. S. Ge, "Temporal convolutional memory networks for remaining useful life estimation of industrial machinery," *arXiv preprint arXiv:1810.05644*, 2018.

[90] G. Xu, M. Liu, Z. Jiang, W. Shen, and C. Huang, "Online fault diagnosis method based on transfer convolutional neural networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, pp. 509–520, Feb 2020.

[91] J. Sun, C. Yan, and J. Wen, "Intelligent bearing fault diagnosis method combining compressed data acquisition and deep learning," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, pp. 185–195, Jan 2018.

[92] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

A

Paper I

A Comprehensive Survey of Prognostics and Health Management based on Deep Learning for Autonomous Ships

André Listou Ellefsen, Vilmar Æsøy, Sergey Ushakov, and Houxiang Zhang, *Senior Member, IEEE*

***Abstract*—The maritime industry widely expects to have autonomous and semi-autonomous ships (autoships) in the near future. In order to operate and maintain complex and integrated systems in a safe, efficient and cost-beneficial manner, autoships will require intelligent Prognostics and Health Management (PHM) systems. Deep learning (DL) is a potential area for this development, as it is rapidly finding applications in a variety of domains, including self-driving cars, smartphones, vision systems, and more recently in PHM applications. This paper introduces and reviews four well-established DL techniques recently applied to various practical PHM problems. The purpose is to support creativity and provide inspiration towards PHM based on DL (PHMDL) in autoships and the maritime industry. This paper discusses benefits, challenges, suggestions, existing problems, and future research opportunities with respect to this significant new technology.**

***Index Terms*—Autonomous ships, deep learning, maritime industry, prognostics and health management.**

ABBREVIATIONS AND ACRONYMS

AE	Autoencoder
BB-RBM	Bernoulli-Bernoulli RBM
BP-NN	Back-propagation Neural Network
C-MAPSS	Commercial Modular Aero-Propulsion System Simulation
CBM	Condition Based Maintenance
CD	Contrastive Divergence
CM	Condition Monitoring
CNN	Convolutional Neural Network
DAE	Denoising Autoencoder
DBN	Deep Belief Network
DL	Deep Learning
DM	Decision Maker
DSSs	Decision Support Systems
DT	Digital Twin
ETTF	Estimation Of Time To Failure
FFT	Fast Fourier Transform
FNN	Feed-forward Neural Network
GB-RBM	Gaussian-Bernoulli RBM
GG-RBM	Gaussian-Gaussian RBM
GRU-LSTM	Gated Recurrent Unit LSTM
HMI	Human Machine Interface
HMM	Hidden Markov Model
IoT	Internet Of Things
LR	Logistic Regression
LSTM	Long-Short Term Memory
MD	Mahalanobis Distance
MFCC	Mel-frequency Cepstrum Coefficient
MLP	Multilayer Perceptron
NIST	National Institute Of Standards And Technology
NN	Neural Network
PCA	Principal Component Analysis
PHM	Prognostics And Health Management
PHM08	The 1st International Conference On Prognostics And Health Management In 2008
PHMDL	Prognostics And Health Management Based On Deep Learning
PM	Preventive Maintenance
RBMs	Restricted Boltzmann Machines
RCM	Reliability Centered Maintenance
ReLU	Rectified-linear Unit
RF	Random Forest
RL	Reinforcement Learning
RM	Reactive Maintenance
RNNs	Recurrent Neural Networks
RUL	Remaining Useful Life
RVM	Relevance Vector Machine
SAE	Sparse Autoencoder
SOM	Self-organizing Maps
SVM	Support Vector Machine
TDNNs	Time-delay Neural Networks
TKEO	Teager-Kaiser Energy Operation
WPT	Wavelet Packet Transform

André Listou Ellefsen, Vilmar Æsøy, and Houxiang Zhang are with the Department of Ocean Operations and Civil Engineering, as part of the Mechatronics Laboratory, Norwegian University of Science and Technology, Aalesund, 6009 Norway, (e-mail: andre.ellefsen@ntnu.no; vilmar.aesoy@ntnu.no; hozh@ntnu.no).

Sergey Ushakov is with the Department of Marine Technology, Norwegian University of Science and Technology, Trondheim, 7491 Norway, (e-mail: sergey.ushakov@ntnu.no).

Manuscript received February 14, 2018; revised August 31, 2018 and January 25, 2019; accepted March 21, 2019. Date of publication April 30, 2019.

I. INTRODUCTION

AUTONOMOUS ships operate on the surface of the water entirely by themselves. Semi-autonomous ships require specialists and technicians who operate and monitor them from an onshore location through a satellite data link [1]. The

industry as well as academics widely expect that autoships, a term that encompasses both, will increase the performance of maritime operations, improving safety and profitability of industries that use them [2]. Many projects are undertaking to create such vessels [3]. Autoships will rely on complex and integrated systems to perform their main functions, and degradation of such systems during operation poses a serious threat to operations. Thus, they will require intelligent maintenance decision support systems (DSSs), which has begun to develop.

In general, maintenance in shipping follows either a reactive maintenance (RM) or preventive maintenance (PM) approach [4]. RM introduces high risks of unscheduled downtime, while PM provides relatively high reliability, but at unnecessary costs due to predetermined maintenance intervals [5]. PM also will not detect random failures, which are in fact the most common failure pattern in the maritime industry [6]. Thus, a more predictive maintenance approach is necessary in order to identify these kinds of failures. A predictive system will considerably increase the operation performance and drastically decrease unexpected system failures [7].

During the past decade, Prognostics and Health Management (PHM) has emerged as a promising engineering discipline for predictive maintenance decision support. It has enhanced potential to detect, isolate, and identify precursor and/or incipient faults of components and sub-components, monitors and predicts the progression of the fault, and provide decision-support or automation to develop maintenance schedules and asset management procedures [8]. Indeed, recent studies have confirmed that PHM is a positive alternative to traditional Condition Based Maintenance (CBM) and has therefore gained attention in both academia and the maritime industry [8]–[10]. However, DSSs with a high degree of decision automation have continue to fail frequently in industrial applications [11]. Accordingly, intelligent PHM systems require more precise and robust data-driven algorithms than systems to date have used.

PHM systems thus far have depended on traditional data-driven diagnostics and prognostics approaches [12]–[15] and signal processing techniques [16]. With the development of internet of things (IoT) and rise with big data, the traditional approaches confront several challenges when processing the increased volumes of data. Typically, they exploit human-engineered feature extraction methods, supervised machine learning algorithms, and shallow architectures. Thus, the traditional approaches are highly application-dependent, require large quantities of labeled training data, and are simply not designed for complex and large data sets in real-world applications [17], [18].

However, during the past decade with increased processing power and great progress in graphics processors [19], DL techniques have seen rapid developments. The areas of signal and information processing [20], speech recognition [21]–[23], images [24], [25], natural language processing [26], [27], and visual tracking [28] have seen significant improvements. DL techniques consist of several layers of non-linear processing stages. They utilize supervised or unsupervised learning strategies to automatically extract feature representations from raw input data. As a result, they are able to capture complicated, hierarchically statistical patterns in more complex, high dimensional and noisy real-world data [29]. For this reason, DL techniques are the most promising area of research to overcome the limitations of traditional diagnostics and prognostics approaches [30]. Nonetheless, issues remain that make it difficult to apply DL techniques to practical PHM problems.

Autoships requires intelligent PHM systems that must be capable of providing reliable diagnostics and prognostics information in varying operating environments [31]. Additionally, lack of onboard crew members and the introduction of highly automated systems necessitate an end-to-end solution. DL techniques are less application-dependent than traditional machine learning algorithms because they are able to process raw and varying sorts of input data. Consequently, human-engineered feature extraction methods are not necessary. DL techniques therefore require minimal human input in the data processing stage and can be considered an end-to-end solution. Nevertheless, DL techniques are still normally applied to perform supervised classification and/or regression tasks within the PHM domain [32]–[34]. With respect to autoships and the maritime industry generally, the lack of fault labels and run-to-failure data of components and sub-components are major issues towards successful implementation of PHM systems based on current DL techniques [35].

This paper reviews and discusses both theoretical and practical issues regarding DL techniques. The broad PHM applications and extensive literature make it impossible for one review to embrace all the work in the field. This review aims to provide a summary of the most important advances in DL techniques recently applied to PHM suitable for autoships and the maritime industry. The important advances introduced in this paper mainly took place from 2013 to 2018. The current research status and issues, benefits, challenges, and future research opportunities will be discussed. Although many DL techniques can be used for PHM purposes, the focus nonetheless is on Autoencoder (AE), Convolutional Neural Network (CNN), Deep Belief Network (DBN) and Long-Short Term Memory (LSTM). This is primarily because they are well-established and show great promise for future work.

The overall organization of the paper is as follows. Section II introduces the necessary background on PHM and DL. Section III considers the main benefits in applying PHM based on DL in autoships, as well as the most important challenges that arise in the field. Section IV reviews DL applied to PHM in other applications suitable for autoships and the maritime industry. This section elaborates strengths and weaknesses in a more theoretical and practical understanding. To the best of the authors' knowledge, the use of intelligent PHM systems based on DL techniques in autoships have not yet been studied comprehensively. Thus, section IV will provide inspiration to obtain both knowledge and understanding. Section V provides discussions regarding suitable solutions for autoships, consisting of important open questions, existing problems, and future research opportunities. Finally, Section VI concludes the survey paper.

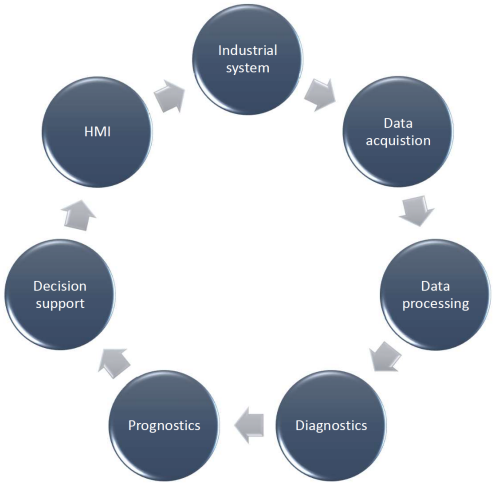


Fig. 1: CBM flowchart adopted from [44].

II. Background: Prognostics and Health Management and Deep Learning

In this section, the necessary background on PHM and DL will be introduced. First, PHM is defined in general. Next, each step of PHM is explained and discussed. Finally, DL is presented with its promising aspects.

A. Prognostics and Health Management

PHM is an emerging engineering discipline that strives to decrease and ultimately eliminate inspections and time-based maintenance intervals [36]. This will be achieved through accurate condition monitoring (CM), precursor and/or incipient fault-detection, -isolation and -identification, and prediction of approaching failures. PHM amplifies and integrates the principles of both CBM and Reliability Centered Maintenance (RCM). It is designed to predict and protect the integrity of complex systems, components, and sub-components by avoiding unforeseen operational problems [37]. This creates a robust system to optimize maintenance decision making in order to increase the reliability and expected lifetime of industrial systems. Industrial systems such as the automotive industry [38], [39], the U.S Department of Defence [40], the aerospace and aviation industries [41], [42], and manufacturing systems [43] have recently integrated PHM with success.

PHM consists of seven steps initially defined from CBM [44]. Figure 1 illustrates the steps. The following subsections briefly discuss each step.

1) Data acquisition and processing: Data acquisition is the process of accumulating and storing raw sensor data related to the system condition. The data collected is usually categorized as CM data and event-data. CM data is the sensor measurements associated with the system health, while event-data is the knowledge obtained from an event (e.g. what kind of failure did occur, when and where did the failure take place, who performed the maintenance procedure) [45]. Event-data provides useful information as to the performance of current features, as well as feedback in redesign or enhancement of features [44]. Thus, it is as important as CM data, although humans generally enter it manually, making it more fallible. An optimal maintenance system should automatically collect the event-data.

Data processing includes data cleaning and data analysis. Cleaning isolates potential human and/or sensor faults and eliminates data that reflects these errors. The data can be a value type, a waveform type, or a multidimensional type [44]. Waveform and multidimensional data may contain noise. Therefore, cleaning also generally includes methods like amplification, data compression, data validation, denoising, and filtering to enhance the signal-to-noise ratio [46]. Data analysis extracts condition indicators that represents incipient and/or precursor failures or faults. The main purpose of those features is to maximize diagnostics and prognostics accuracy in order to decrease false alarms. The literature has described processing techniques like wavelet transform, data denoising, and data smoothing [47], [48]. [46], [49] describes signal processing and feature extraction.

2) Diagnostics and prognostics: An effective PHM system includes diagnostics and prognostics approaches in order to provide ample and efficient decision support or automation. Diagnostics identify, localize, and determine the severity of an evolving fault condition [36]. It involves fault detection, fault isolation, and fault identification [50]. Fault detection, also called health/condition assessment [37], [45], compares sensor data with expected operational performance, that is, expected values of system parameters such as pressure, temperature, and vibration, to identify irregular operating conditions. Fault isolation involves pinpointing the component or sub-component that is degraded. Fault identification determines fault- type and dimension according to classes associated with specific values of measured signals [51]. Normally, this classification process uses a supervised classifier (e.g. machine learning algorithm) to classify various faults.

Prognostics predict the progression of faults, and hence, estimate the available time before a component or sub-component loses its operational ability, namely, before a failure [52]. Because the large uncertainties involved, researchers have called prognostics "the Achilles' heel" of PHM [53], [54]. According to [55], the technical definition of prognostics is the estimation of time to failure (ETTF). However, in line with common usage in the literature, this paper uses the technical term remaining useful life (RUL). Any RUL estimation should include associated confidence intervals, which will indicate the window in which maintenance or repair must be conducted [53]. Such intervals add assurance of continuous operation in spite of the inherent uncertainty associated with the degradation process, human errors, and flaws in both the diagnostics and prognostics approach applied in the PHM system [56]. Maintenance decisions based on prognostics information should be grounded in confidence intervals instead of a particular RUL value. The

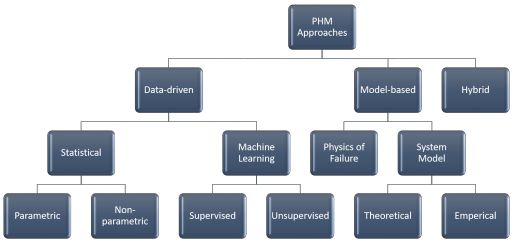


Fig. 2: A hierarchy of the three main diagnostics and prognostics approaches.

TABLE I: Recent PHM/CBM reviews based on traditional diagnostics and prognostics approaches (the years between 2006 and 2017).

Author	Refs.	Year	PHM application	Approaches
Tahan et al.	[60]	2017	Diagnostics and prognostics: Gas turbines	Data-driven Model-based Hybrid
Bailey et al.	[61]	2015	Diagnostics and prognostics: Engineering systems	Data-driven
An et al.	[62]	2015	Prognostics: Fatigue crack growth	Data-driven Model-based
Lee et al.	[8]	2014	Diagnostics and prognostics: Rotary machinery systems	Data-driven Model-based
Sikorska et al.	[56]	2010	Prognostics: Selection of RUL models	Data-driven Model-based
Vachtsevanos et al.	[50]	2006	Diagnostics: Book chapter	Data-driven Model-based
Vachtsevanos et al.	[53]	2006	Prognostics: Book chapter	Data-driven Model-based
Roemer et al.	[63]	2006	Prognostics: Engines	Data-driven Model-based
Jardine et al.	[44]	2006	Diagnostics and prognostics: Machinery systems	Data-driven

confidence intervals increase the reliability of the PHM system.

Successful prognostics depend on accurate diagnostics [11], [56], [57]. Diagnostics is necessary when prognostics fails and can prevent future failures of similar characteristics [44]. Even so, prognostics are considered more important than diagnostics to the ultimate goal of zero-downtime performance. This is because prognostics has the potential to prevent failures before they occur. Nevertheless, several challenges to successful implementation still exist. Thus the challenges, which [58] describes, should be addressed.

No common accepted prognostics methodology exists [43]. Figure 2 illustrates the three most common, however, which are Data-driven, Model-based, and Hybrid. The hybrid approach is a combination of data-driven and model-based approaches, aiming to utilize strengths of both approaches while avoiding their weaknesses [59]. Table I provides a summary of the most comprehensive PHM/CBM reviews regarding traditional diagnostics and prognostics approaches considered in this survey paper.

3) Decision support and HMI: The final object of a PHM system is to provide reliable decision support or automation in order to enable effective maintenance scheduling. Decision support should assist a decision maker (DM), while decision automation uses software to provide entirely autonomous decisions [64]. However, according to [11], the output of today's industrial PHM systems usually constitutes decision support, as decision automation has not been integrated globally. Normally, inputs from human experts (application-dependent domain knowledge) and a DM who interprets the outputs compose the system [65]. Nevertheless, the human expert-generated input will fail when it encounters new conditions, that the knowledge base does not define. In addition, the most proficient DM has an insufficient cognitive capacity to analyze and understand large quantities of information [66]. Hence, decision support in the age of big data is subjected to uncertainties and does not always ensure good quality decisions. The literature provides several excellent reviews discussing this issue [11], [65], [66].

The advantages of a PHM system are highly connected to the decision-making based on the accumulation and understanding of diagnostics and prognostics information. Making the best decisions based on complex and large quantities of information is difficult [66]. However, advanced and deep signal processing and machine learning techniques are evolving rapidly [18]. These techniques provide automatic feature extraction and unsupervised learning procedures. Thus, such techniques minimize the human expert-generated input and have the potential to contribute to more intelligent PHM systems.

As [11] propose, the reliability of a fully autonomous (intelligent) PHM system needs to be greater than 99%. This level of reliability makes it possible for the PHM system to provide directions for maintenance procedures transferring directly from the system to the maintenance personnel, without the involvement of a DM. Another important aspect of autonomous PHM systems is that they must prove reliability in order to utilize the "black-box" approach. PHM systems with low reliability should enable user-access to the source code in order to promote understanding and trust in the system [11].

The human machine interface (HMI) is also an important perspective regarding understanding and trust since the screens and displays heavily affect how a DM or the maintenance personnel understand the PHM system. A web-based PHM system, in which the user interacts using a thin-client web browser, has several advantages. According to [67], this system is powerful to retrieve, analyze, and visualize structured data from high-dimensional databases. It can provide access to unstructured data and promote communication and decision making in distributed teams [67].

B. Deep Learning

In recent years, DL has turned into an extremely active subfield of machine learning. DL and big data are probably the most significant trends in the fast-growing digital world [19]. According to the National Institute of Standards and Technology (NIST) [68], big data is "the large amount of data in the networked, digitized, sensor-laden, information-driven world." NIST goes on to note that big data "can overwhelm traditional

technical approaches and the growth of data is outpacing scientific and technological advances in data analytics." Big data forces a dramatic paradigm shift towards data-driven approaches and discoveries within scientific research. [18], [19] provide exceptional reviews regarding the relationship between big data and DL. In addition, machine learning is now a major technical field of the signal processing society [69].

The expansion of big data and IoT tends to make traditional machine learning algorithms like Hidden Markov Model (HMM) [70], Support Vector Machine (SVM) [71], and Neural Network (NN) with one hidden layer [61] vague, creating several challenges. First, traditional algorithms utilize shallow architectures, with only two stages of data-dependent computation elements. This means that shallow architectures contain only a small number of non-linear processing transformations. Previous analyzes of the boolean circuit complexity theory literature [72], [73], have revealed that shallow circuits require exponentially more elements than deeper circuits [74]. According to [75], this applies also to shallow and deep architectures in machine learning algorithms when they are required to process highly non-linear and varying functions. Consider the parity function with d inputs. Gaussian SVM requires 2^d parameters, NN with one hidden layer requires d^2 parameters, while a deep architecture requires d parameters with $log_2\, d$ layers. As a result, shallow architectures are inefficient due to the increased number of computational elements (e.g. hidden units), which require many examples [75]. Consequently, Gaussian SVM and NN with one hidden layer suffer from a decreased capacity to process more complex and high-dimensional real-world data with accuracy [16], [29]. Second, most traditional machine learning algorithms use supervised learning procedures. This means they require large quantities of high-quality labeled training data. However, in real-world applications large amounts of the data are unlabeled, and according to [76], most data collected in the age of big data is heterogeneous and unstructured. Finally, traditional machine learning algorithms lack the ability to extract and organize the discriminative information from the data [77].

Over 60 years ago, Richard Bellman declared that learning complexity grows exponentially with the linear increase in the dimensionality of the data [17]. He named this phenomenon "The curse of dimensionality" [78]. During the last decades, researchers have applied human-engineered feature extraction methods to the data processing stage to reduce the dimensionality of the data so that traditional machine learning algorithms can process it [17]. As a consequence, much of the actual work in using traditional machine learning algorithms goes into the design of the features because the performance of the algorithms relies heavily on the chosen method [77]. Hence, human-engineered feature extraction methods require precise engineering and substantial domain expertise, and the applied algorithm becomes highly application-dependent [79].

Recent discoveries in neuroscience, increases in computing power and an explosion of digital data have been the central motivational factors for the emergence of DL. The discoveries in [80], [81] clarify that the neocortex allows signals to propagate through a complex hierarchy of units. In time, these units will learn to represent observations based on the regularities they express [17]. DL focus on similar characteristics as the neocortex. Actually, DL is a three-decade-old technique and a renewal of the even older NNs [82].

Great advances and innovations have been achieved in DL since 2006 [75], [83]–[85]. At that time researchers, gathered by the Canadian Institute for Advanced Research, introduced unsupervised learning strategies that could extract features without requiring labeled training data, that is, capture statistical patterns in the observed data [79], [86]. Unsupervised DL techniques introduce hierarchical structures to automatically extract important features, from low-level input observations to high-level abstractions, using unsupervised pre-training where all layers are initialized. After precise fine-tuning, the highest level abstract features will normally be the input to a supervised classifier or regressor, minimizing the global training requirement [87].

More specifically, a DL technique is a multilayer stack of non-linear processing stages to compactly (with few parameters) represent highly non-linear and varying functions [75]. Most of the stages are subjected to supervised or unsupervised learning and compute non-linear input-output mappings. Each stage modifies its input in order to increase both the invariance and selectivity of the representation [79]. Consequently, DL techniques can often capture complex, hierarchically statistical patterns in unstructured, high dimensional, and noisy real-world data [29]. With multiple non-linear layers, DL techniques make possible extremely involved functions of its inputs that, at the same, are time-sensitive to small details and insensitive to large irrelevant variations [79].

In the past decade, DL techniques have shown fast advancements with notable impacts on signal and information processing [20], beaten records in image recognition [25] and speech recognition [22], and outperformed traditional machine learning algorithms in natural language understanding [27], and diagnostics and prognostics purposes [32], [88]. In addition, as [19] state, DL is going to play an important role in prediction tasks due to increased processing power and the advances in graphics processors. A great historical survey of DL is given in [89]. It summarizes both current work and work from the previous millennium, including the history of supervised learning and back-propagation.

III. Autonomous Ships: Benefits and Challenges in Applying PHM based on DL

Only three years ago, most people considered autoships as a futuristic fantasy [3]. Today, however, this preconception has changed drastically. In fact, autoships will almost certainly be in commercial use by the end of this decade [3]. The first vessels will require a few crew members, however, at least to operate in challenging maritime areas. The transition to totally human-free autoships will likely take place gradually over a period of a few decades [3].

According to [3] and [31], securing regulatory approval, support from the industry, and public acceptance for autoships requires evidence they are at least as safe as traditional ships used for similar operational tasks. As they will ultimately, have

no maintenance personnel on board ready to perform unsystematic maintenance, safety critical systems and components must be more reliable than on traditional ships.

Autoships will transfer real-time diagnostics and prognostics information to shore to permit analysis and prioritization of issues of critical systems and components. Todays maritime maintenance procedures, by contrast, typically follow an RM or PM approach [4]. RM can be described as post-failure repair of components or sub-components, while PM involves predetermined maintenance intervals based on constant intervals, age-based or imperfect maintenance [5]. Traditional ships tend to rely heavily on onboard maintenance personnel since it is less costly to conduct RM and/or PM approaches while still at sea [31].

RM would create large and unnecessary costs when critical system/component failures occur during operation of autoships. Both the process of dispatching maintenance personnel while the autoship is still at sea and the process of guiding the vessel back to shore in order to perform repairs would create random and unplanned downtime, compromising efficiency. On the other hand, the constant and experience based maintenance intervals utilized in PM could be scheduled around predetermined port of calls. This will, of course, provide high reliability, but it involves unneeded maintenance inspections and procedures of completely functional systems. It also might not prevent the random need for maintenance involved in RM, since random failures are the most common type in the maritime environment [6]. The need for predictive maintenance approaches, such as intelligent PHM systems, is clear.

Based on the background information and brief discussion in Section II, it is obvious that DL techniques have the potential to overcome the limitations of traditional machine learning algorithms applied to diagnostics and prognostics purposes. For that reason, DL techniques are highly suitable to be applied in intelligent PHM systems. The next step in this survey paper is to introduce and discuss benefits and challenges in applying Prognostics and Health Management based on deep learning (PHMDL) in autoships.

A. Benefits

- Normally, critical systems on traditional ships are over-engineered by built-in redundancy. In this way, traditional ships complete their operational tasks even if a serious functional failure occurs. This design philosophy is related to historical inaccessibility to shore [90]. However, Inmarsat and Telenor have recently launched the data transfer satellites Inmarsat-5 and Thor 7, respectively, which will provide high-speed broadband connections to ships at sea [1]. This will enable new design philosophies, including online PHMDL systems, as alternatives to the legacy redundancy policy. Real-time diagnostics and prognostics of components and sub-components in which online PHMDL systems are referred to an onboard system that links to shore will make it possible to contribute the most efficient operating conditions possible, and enable future autoships without onboard maintenance personnel [31].
- The ultimate goal of a PHMDL system is to achieve zero-downtime performance. Real-time and reliable RUL estimations, with associated confidence intervals, of different components and sub-components, will have an enormous impact on the maintenance procedure and safety concept on autoships. When the RUL of a faulty component is estimated, the maintenance procedure can be scheduled to the next appropriate port of call, or if necessary, dispatching maintenance personnel before a failure occurs when the autoship is still in operation [1]. This will significantly increase the operational performance, and at the same time, drastically decrease unexpected system failures. In addition, reliable estimations will provide trust in safe behavior in offshore activities [7].
- According to [1], the insurance company Allianz reported in 2012 that between 75% and 96% of marine accidents are a result of human errors. This is mainly a result of human exhaustion, but also because today's maritime activities require humans both to manage planned operational activities and make complicated decisions based on the overall system conditions [7]. Autoships will reduce both the number of crew members and the influence of human DMs due to increased autonomous and intelligent operational planning and decision making. In this way, autoships will have the potential to decrease human errors and the risk of injury to crew members [31]. PHMDL systems have great potential to contribute to this human error reduction since these systems are less dependent on prior knowledge and human influence.

B. Challenges

- Autoships requires adaptation and integration within the functioning of a business of an organization, and hence, significant changes in the organizational culture [4]. The introduction of autoships also involves confidence and trust in "black-box" systems, such as a PHMDL system. These systems are intelligent in that they transfer directions for future maintenance procedures directly from the autoship to the maintenance team on shore. In order to act as a fully autonomous and intelligent system, the PHMDL system must adapt to the varying operational and environmental conditions that occur in the harsh maritime environment [35].
- A further concern is the continuous flow of data to shore. Autoships depend on heavily integrated and complex systems to deliver their main functions. As a result, the associated flow of sensor data becomes massive, high-dimensional, heterogeneous, and unstructured. The PHMDL system will have to provide automatic preprocessing and dimensionality reduction schemes. This massive flow of data also presents a cybersecurity challenge, as hackers would threaten safe maritime operations [1].
- A great challenge in the maritime industry is the lack of run-to-failure data of components and sub-components [35]. Traditional ships are often application-designed and unique, or batch-produced in two to ten

vessel series [4]. These short series creates a slow accumulation of relevant failure data compared to, for instance, the aviation industry that produces hundreds of the same aircraft in series [4]. In addition, traditional ships are typically equipped with components from several different manufacturers [91]. The resulting diversity of uncoordinated monitoring systems increases the complexity of the failure data. With respect to the introduction of autoships and PHMDL systems, it would be advantageous to build extensive databases regarding run-to-failure data of critical and relevant components and sub-components. This could be realized if stakeholders agreed to cooperate to share data.

C. Summary

Reliable and real-time diagnostics and prognostics in autoships have the potential to improve efficiency, maintenance procedures, and safety aspects. Based on the above-mentioned challenges, such as varying operational and environmental conditions and massive data flows, DL techniques will be superior to the combination of human-engineered feature extraction methods and traditional machine learning algorithms. This is because DL techniques utilize unsupervised learning procedures to automatically extract key features and reduce the dimensionality of raw unlabeled input data. Accordingly, DL techniques do not require human-engineered feature extraction methods, such as Mel-frequency Cepstrum Coefficient (MFCC) or wavelet transform, in the data processing stage. This means that the diagnostics and prognostics accuracy of a PHMDL system is less application-dependent. For that reason, PHMDL systems will have the potential to perform diagnostics and prognostics under different environmental and operational conditions. However, DL techniques are usually used to perform supervised classification and/or regression tasks. For that reason, available run-to-failure databases would be advantageous. The next section reviews recent PHMDL applications. This is to fully elaborate strengths and weaknesses in a more theoretical and practical understanding.

IV. Applications of Deep Learning to Prognostics and Health Management

In recent years, DL has emerged as an innovative and encouraging research field for PHM [30]. This section introduces and reviews well-established DL techniques like Autoencoder (AE), Convolutional Neural Network (CNN), Deep Belief Network (DBN) and Long-Short Term Memory (LSTM) based on applications to PHM in the recent five years. This information will support the need for creativity and inspiration in producing PHMDL possibilities for autoships.

A. Deep Belief Network

1) Introduction: In 2006, Hinton et al. [83], introduced a greedy layer-wise unsupervised learning algorithm. This was the first valid algorithm for training fully-connected deep architectures, and hence, marked the starting point for notable progress in DL. The algorithm was originally introduced for DBNs and improved previous optimization problems of training deep architectures by initializing the weights in a region near a good local minimum [75]. The algorithm makes it possible to automatically learn internal representations of data. These internal representations are high-level abstractions of the input and allow a network to produce complex input-output mappings directly from data [87]. In this way, the algorithm is, in theory, not dependent on human-engineered features in the data processing stage.

The fundamental ideas of the algorithm are as follow [75], [87];

1) Pre-train one layer at a time in a greedy way. In other words, layer n is kept fixed while the $n-th$ layer is trained using the output of n as the input.
2) Perform unsupervised learning at each layer in order to maintain information from the input.
3) Fine-tune the whole network with respect to the global training requirement.

DBNs consists of several layers of Restricted Boltzmann Machines (RBMs) [92], and normally some additional layers to conduct e.g. classification or regression tasks.

RBMs [29], [75], [86], [93], [94] are probabilistic generative models that learn a joint probability distribution from unlabeled training data. They are a special type of Markov random fields, typically with Bernoulli or Gaussian stochastic visible units, v, in a single input layer and Bernoulli stochastic hidden units, h, in a single hidden layer. Normally, as shown in Figure 3, the visible and hidden units are fully connected with bias vectors, b and c, respectively, and weight matrix, w. In addition, units in the same layer have zero connections. Consequently, RBMs can be defined as symmetrical bipartite graphs. The hidden layer in the first RBM will serve as the input layer for the second RBM.

The Bernoulli-Bernoulli RBM (BB-RBM) is the binary version of RBMs. It is an energy-based model with the joint probability distribution specified by its energy function [93]:

$$P(v,h) = \frac{1}{Z} e^{-E(v,h)} \tag{1}$$

The energy function is given by:

$$E(v,h) = -\sum_{i=1}^{V} b_i v_i - \sum_{j=1}^{H} c_j h_j - \sum_{i=1}^{V}\sum_{j=1}^{H} w_{ij} v_i h_j \tag{2}$$

where w_{ij} represents the weight between the binary states of visible unit v_i and hidden unit h_j, b_i and c_j denotes the bias terms, while V and H indicates the numbers of visible and hidden units, respectively. The partition function, Z, is given by summing all possible combinations of visible and hidden vectors. It ensures that the distribution is normalized:

$$Z = \sum_{v}\sum_{h} e^{-E(v,h)} \tag{3}$$

Due to the fact that RBMs are symmetrical bipartite graphs, the conditional probabilities $p(v|h)$ and $p(h|v)$ are factorial, and can be efficiently calculated as (see full derivation in [86],

[93]):

$$P(v_i = 1|h) = \sigma\Big(b_i + \sum_{j=1}^{H} w_{ij} h_j\Big) \quad (4)$$

$$P(h_j = 1|v) = \sigma\Big(c_j + \sum_{i=1}^{V} w_{ij} v_i\Big) \quad (5)$$

where σ is the activation function. The logistic sigmoid function $\frac{1}{1+e^{-x}}$ is a usual choice [29].

For real-value data applications, Gaussian-Bernoulli RBM (GB-RBM) is normally used as the initial RBM to convert real-valued stochastic variables to binary stochastic variables [95], [96]. The second RBM can then be a BB-RBM with a rectified-linear unit (ReLU) [97] transformation as the activation function for further processing. The energy function for GB-RBM is given by [93]:

$$E(v,h) = \sum_{i=1}^{V} \frac{(v_i - b_i)^2}{2\gamma_i^2} - \sum_{j=1}^{H} c_j h_j - \sum_{i=1}^{V}\sum_{j=1}^{H} \frac{v_i}{\gamma_i} h_j w_{ij} \quad (6)$$

where γ_i is the standard deviation of visible unit v_i. The corresponding conditional probabilities are expressed by:

$$P(v_i = x|h) = \frac{1}{\gamma_i\sqrt{2\pi}} exp\left(- \frac{(x - b_i - \gamma_i \sum_j h_j w_{ij})^2}{2\gamma_i^2} \right) \quad (7)$$

$$P(h_j = 1|v) = \sigma\Big(c_j + \sum_{i=1}^{V} w_{ij} \frac{v_i}{\gamma_i}\Big) \quad (8)$$

where x is a real number. In practice, to make the model implementation of GB-RBM more simple, the input data should be normalized to have zero mean and unit variance [93]. It should be noted that a study conducted in 2010 has shown that noisy ReLUs works better than Bernoulli stochastic units in RBMs hidden layer [98].

The contrastive divergence (CD) [99] update rule is used to train RBMs:

$$\Delta w_{ij} = \epsilon\Big(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{recon}\Big) \quad (9)$$

where ϵ is the learning rate and $\langle * \rangle$ denotes expectations under the distribution. The first expectation is with respect to the data distribution and samples visible units based on hidden units (Equation 7). The second expectation has to do with the reconstructed input data distribution, generated by Gibbs sampling, which samples hidden units based on visible units (Equation 8). The reconstruction part of RBM training makes it a generative model since it guesses the probability distribution of the original input. The weights between the input layer and the hidden layer are then updated using Equation 9. This process will repeat until the parameters converge, that is, the hidden layer is able to approximate the input layer. Thus, RBMs model data distribution using hidden units without the use of label knowledge. After the RBM training process, the parameters are presented to the DBN. In the end, the whole DBN architecture is fine-tuned using supervised back-propagation with a much smaller data set of labeled training data [100]. It should be noted that the training process of RBMs is crucial in applying DBNs successfully to practical problems. [93] includes a practical training guide by the machine learning group at the University of Toronto.

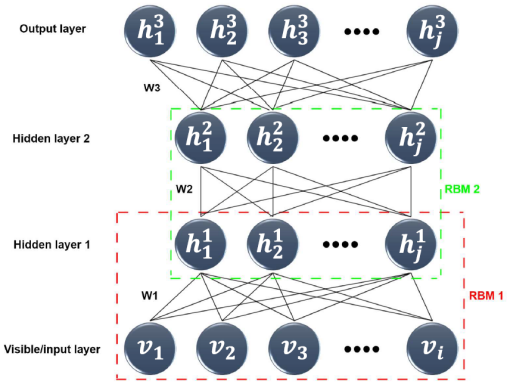


Fig. 3: A simple DBN representation with two hidden layers. Each visible and hidden unit are essentially nodes where calculations take place.

2) Recent applications to PHM: DBNs are capable of providing automatic feature extraction from unlabeled training data and of performing supervised classification or regression tasks by adding one or more additional layers. These properties are well suited for PHM systems. The paragraphs below review applications of DBNs to PHM in the years between 2013 and 2017.

Regardless of the well-proven applicability of traditional data-driven diagnostic approaches, CM through multiple sensors remains one of the major difficulties to be addressed in the areas of classification and health diagnostics [14]. The reason for this is that the complexity of the classification model increases with multiple sensors and heterogeneity of sensor signals, and hence, the data becomes highly dimensional. Tamilselvan et al. [32] proposed a novel DBN approach for use in multi-sensor health diagnostics state classification. The proposed approach was demonstrated with the publicly available data set from the competition held at the 1st international conference on Prognostics and Health Management in 2008 (PHM08) [101]. The data set was produced by the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS), provided by NASA [102]. In addition, two case studies were conducted for further demonstration. The DBN provided better classification performance compared with four traditional data-driven diagnostic algorithms; SVM, back-propagation Neural Network (BP-NN), self-organizing maps (SOM), and Mahalanobis distance (MD). However, in this study, labeled training data was used for different health states. Thus, this study did not investigate DBNs full potential for automatic feature extraction of unlabeled training data.

Tran et al. [103] also utilized DBN as the diagnostics

approach. The proposed approach was validated with signals from a two-stage reciprocating air compressor under different valve conditions. The DBN was used to classify faults and showed superior performance compared to traditional data-driven diagnostic algorithms, such as Relevance Vector Machine (RVM) and BP-NN. However, in this study, the DBN approach was only used for classification, and hence, it did not examine the automatic feature extraction of unlabeled training data aspect. On the other hand, the Teager-Kaiser energy operation (TKEO) and wavelet transform were used as the feature extraction methods.

Nevertheless, the automatic feature extraction aspect was heavily explored in [100] and [104]. Yang Fu et al. [100] demonstrated that the performance of the traditional data-driven diagnostics algorithms SVM, Multilayer perceptron (MLP) and k-means, strongly depends on the human-engineered feature method selected. Three kinds of features are included in this comparison: raw vibration data with normalization, MFCC, and wavelet method. In this study, the DBN consistently presented wonderful classification performance in all three features. This shows that DBN is a promising automatic feature extraction tool to be used on raw signals without too much data preparation. [104] is a similar study. Li et al. utilized a DBN as a statistical feature learning tool for bearing and gearbox systems in time, frequency, and time-frequency domains. The proposed approach indicated better classification results compared to SVM and a single layer of GB-RBM.

Various traditional data-driven prognostics approaches have been proposed for different applications. Normally, they involve human-engineered feature extraction methods in combination with a single traditional machine learning algorithm. As a consequence, these traditional approaches can hardly maintain good generalization performance and adapt to different prognostics applications. However, Zhang et al. [105] proposed a multiobjective DBN ensemble (MODBNE) method. MODBNE applies a multiobjective evolutionary ensemble learning framework combined with the DBN training process. In this way, the proposed method is able to create multiple DBNs of varying accuracy and diversity, which in fact are two conflicting objectives. The evolved DBNs are then combined to perform RUL estimations. The proposed method was evaluated by the publicly available C-MAPSS data set, the turbofan engine degradation simulation data set [106] produced by the C-MAPSS and provided by NASA. The big difference between the PHM08 data set and the C-MAPSS data set is that only the latter provides true RUL targets. The proposed approach was compared with several traditional data-driven algorithms.

Deutsch et al. [107] introduced a deep architecture for RUL estimations of rotating components using vibration sensors. The proposed approach combines the automatic feature learning ability of DBN, and the predictive power of feed-forward Neural Network (FNN). The approach is termed DBN-FNN and has the opportunity to either utilize processed vibration features or extract features from the vibration data to estimate RUL. The RUL estimation includes confidence boundaries obtained by the re-sampling technique jackknife. The proposed approach overcomes the limitations of traditional data-driven approaches by performing automatic feature extraction and RUL estimations without human interference or prior knowledge. Thus, the DBN-FNN approach confirms potential towards the application of autoships.

To enable accurate RUL estimations, feature extraction is a vital step. Liao et al. [108] proposed an enhanced single layer RBM with a novel regularization term to automatically generate features that are suitable for RUL estimations. The main advantage of the regularization term is that it tries to maximize the trend of the output features. Consequently, it has the potential to make better representations of the degradation patterns in the system. The proposed approach is compared with traditional RBM and principal component analysis (PCA). This method has the opportunity to be extended to a DBN by stacking multiple enhanced RBMs. However, the proposed approach is based on a Gaussian-Gaussian RBM (GG-RBM). According to [75], DBNs containing only Gaussian units will only be able to model Gaussian data. In addition, the mean-field propagation through a Gaussian unit gives rise to a purely linear transformation. Hence, the internal representations would be completely linear. In other words, Gaussian transformations do not work well on RBMs' hidden layers.

Jiang et al. [109] proposed a deep architecture involving a DBN and a non-linear kernel-based parallel evolutionary SVM. The objective was to predict evolution states of complex systems in classification tasks. The goal of the algorithm is to predict class labels of test data without any label information. In two case studies, the proposed approach outperformed both SVM and the traditional DBN.

DBNs have also been successfully and heavily applied in time series forecasting [110]–[112].

B. Autoencoder

1) Introduction: The greedy layer-wise unsupervised learning algorithm introduced by Hinton et al. [83] and further analyzed by Bengio et al. [75], can be applied not only to RBMs but also to AEs. An original AE [29], [75], [77], [86], [94] is an FNN, normally with one hidden layer, trained to reproduce its input to its output by forcing the computations to flow through a "bottleneck" representation [74], namely, dimensionality reduction. The hidden layer, h, describes a code used to represent the input, x. The network consists of two parts: an encoder function $h = f_{\theta_e}(x)$ and a decoder function that produces a reconstruction $r = g_{\theta_d}(h)$. If the AE learns the identity function, $g_{\theta_d}(f_{\theta_e}(x)) = x$, it will not be effective to extract meaningful features [113]. However, modern variations of the original AE are normally restricted to only copy input that is similar to the training data. Consequently, the AE is forced to prioritize which characteristics of the input it should copy. Thus, it often learns useful features of the data, and at the same time, filters useless information [94]. In addition, since the input vector is transformed into a lower dimension, the efficiency of the learning process can be increased [20]. Figure 4 shows a simple AE. It should be noted that AE is also called autoassociator in the literature.

The visible units, x, in the input layer, the hidden units, h, in the hidden layer, and the reconstruction units, r, in the

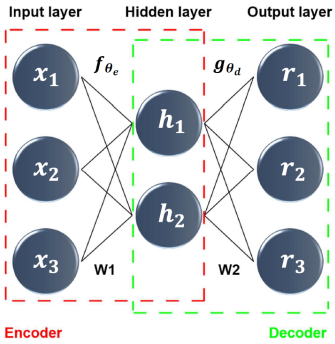


Fig. 4: Simple structure of an autoencoder. Three nodes in the input and output layer and two nodes in the hidden layer (bottleneck).

output layer are connected with weight matrices, w^1 and w^2. The hidden layer and the output layer have bias vectors b and c, respectively. As opposed to the parameterization of RBMs (single weight matrix), the AE framework permits a different matrix in the encoder, θ_e =$\{w^1, b\}$, and in the decoder, θ_d =$\{w^2, c\}$. Nevertheless, in practice it is common to use tied weights, $w^2 = (w^1)^T$, [77]. This provides the parameterizations identical and serves as a regularizer since it constrains the parameter space [29]. θ_e and θ_d are learned concurrently on the task of reconstruction and compared to the original input in order to obtain the lowest possible reconstruction error $L(x, r)$ [77]:

$$J_{AE}(\theta_e, \theta_d) = \sum L(x, g_{\theta_d}(f_{\theta_e}(x))) \tag{10}$$

where L is a loss function such as the squared error $L(x, r) = ||x - r||^2$. Basic AE training consists in finding values of the weights and biases in order to minimize $L(x, r)$. The most normal encoder and decoder function are affine (feed-forward) mappings, optionally followed by a non-linearity [77]:

$$f_{\theta_e}(x) = \sigma_f(b_j + \sum_i w^1_{ji} x_i) \tag{11}$$

$$g_{\theta_d}(h) = \sigma_g(c_i + \sum_j w^2_{ij} h_j) \tag{12}$$

where σ_f and σ_g are the encoder and decoder activation functions. It should be noted that the choice of activation and loss function depends on the input domain range and character. AEs can be stacked, like the RBM, to form a deep architecture. Thus, the training procedure is equivalent to the one introduced for DBNs [83], but using AEs rather than RBMs. [74] presents a comparative study regarding AEs and RBMs. This study suggests that DBNs have a slight edge over stacked AEs. According to [86], this is probably because CD is closer to the log-likelihood gradient than the reconstruction error gradient. There exist several modern variations of the original AE in the literature. In the following subsections, the Denoising Autoencoder (DAE) and the Sparse Autoencoder (SAE) will be introduced in relation to recent applications to PHM.

2) Denoising Autoencoder: Vincent et al. [114], [115] proposed the DAE in 2008. This extension of the original AE was designed to learn more robust representations in a deep architecture. DAEs are trained with corrupted data, $\tilde{x}$, by adding noise into the training data through the stochastic corruption process, $\tilde{x} \sim q(\tilde{x}|x)$. The robustness is achieved when the DAE reconstructs the clean version of the training data through the training process. The objective function for optimization in the DAE is given by:

$$J_{DAE}(\theta_e, \theta_d) = \sum IE_{q(\tilde{x}|x)}[L(x, g_{\theta_d}(f_{\theta_e}(\tilde{x})))] \tag{13}$$

where $IE_{q(\tilde{x}|x)}[*]$ represents the average value over $\tilde{x}$ drawn from the stochastic corruption process $\tilde{x} \sim q(\tilde{x}|x)$ [77]. The major difference between AE and DAE is that, r is a deterministic function of $\tilde{x}$ rather than of x. Hence, DAE must undo the corruption instead of simply copy the input [94]. DAEs can also be stacked to form a deep architecture. The greedy layer-wise training strategy is identical to the strategy for the original AE and RBM. It should be noted that the stochastic input corruption process is only applied in the training procedure in order to learn more robust and valuable representations [116]. Thereafter, the reconstructed clean version is used as the input to the next layer. Various corruption processes like additive Gaussian noise, salt and pepper noise, and masking noise can be considered [115].

3) Sparse Autoencoder: In 2006, Ranzato et al. [117] proposed the learning algorithm for sparse representations. SAE is also an extension of the original AE that aims to use sparse representations in order to produce a simple understanding of the input data by extracting the hidden structure of the data [20]. The training criterion involves a sparsity penalty term, $\Omega(h)$, on the hidden layer, h, in addition to the reconstruction error $L(x, r)$ [94]:

$$J_{SAE}(\theta_e, \theta_d) = \sum L(x, g_{\theta_d}(f_{\theta_e}(x))) + \Omega(h) \tag{14}$$

The sparsity penalty term $\Omega(h)$ is added to the objective function of the original AE (Equation 10) in order to constrain the learned features. It controls the number of active neurons in the hidden layer, h. A neuron is considered active if the output is close to 1, and inactive otherwise [113]. The sparsity penalty term is defined as:

$$\Omega(h) = \beta \sum_{j=1}^{H} KL(\rho|\rho_j) \tag{15}$$

where β controls the weight, H is the number of neurons in the hidden layer and $KL[*]$ is the Kullback-Leibler divergence [118]:

$$KL(\rho|\rho_j) = \rho \, log\frac{\rho}{\rho_j} + (1 - \rho) \, log\frac{1 - \rho}{1 - \rho_j} \tag{16}$$

where ρ is a hyperparameter (typically close to zero, e.g. $\rho = 0.05$ [77]) and ρ_j is the average activation of the hidden unit j. As seen from Equation 16, the sparsity penalty term is zero if $\rho_j = \rho$. Thus, the sparsity penalty term will penalize ρ_j if it deviates considerably from ρ. In other words, it promotes partial activations of each hidden unit as specified by ρ [29]. By only activating a few hidden nodes at the same time, the system robustness is improved. According to [94], SAEs are typically used to learn features for classification tasks due to its enhanced performance. After stacking several SAEs to form a deep architecture, the greedy layer-wise training procedure is also here identical as for the DAE, original AE, and RBM.

4) Recent Applications to PHM: AEs are, as with DBN, capable of providing automatic feature extraction from unlabeled training data, and in addition, performing supervised classification or regression tasks by adding one or more additional layers. The modern versions of AE, DAE, and SAE seem particularly promising for PHM applications and autoships. DAE is robust to noise and SAE has the potential to increase the robustness of the system and the performance of classification tasks. In the paragraphs below, applications of AEs, DAEs, and SAEs to PHM are reviewed in the years between 2014 and 2017.

Feature extraction is a crucial part of a PHM system because it determines the performance of both diagnostics and prognostics. Lu et al. [119] proposed a stacked AE, containing two hidden layers, as the feature extraction method for rolling bearing fault diagnostics. The results indicated that the second hidden layer provided more precise and identifiable features than the first hidden layer and the raw features in the visible layer. Thus, a stacked AE is a promising tool to extract features from bearing signal data.

Typically, in large industrial systems, the data is derived from several platforms that could potentially involve different data types. Based on this, Ma et al. [120] proposed an architecture with multiple input modalities applied to fault diagnostics. The proposed approach is using RBMs to obtain a unified representation for both images and structured data. Then, the unified representation is the input to a stacked AE in order to reconstruct the images and the structured data to obtain abstract features and remove useless information. In the final layer, a supervised linear classifier is added to classify the learned features and fine-tune the whole network. Comparing the proposed approach with BP-NN showed lower misjudgment rate for both normal and fault conditions.

Jia et al. [121] proposed a novel intelligent fault diagnostics method for rotary machinery in order to overcome the limitations of traditional diagnostic approaches. The main limitations highlighted in this study are shallow architectures and the requirement of application-dependent human-engineered feature extraction methods in the data processing stage. To overcome these limitations, the proposed method utilized a stacked AE to adaptively extract fault characteristics (features) from measured signals in the frequency domain, and automatically classify machinery health conditions. The proposed method was validated using rolling element bearing- and planetary gearbox data sets, and finally, compared with the traditional BP-NN. The results indicated that the proposed method overcomes the above-mentioned limitations.

Xia et al. [116] also addresses the limitations of traditional diagnostics approaches, specifically the need for prior knowledge of features and the requirement of large quantities of labeled condition data as the main limitations. In addition, most traditional approaches need to be rebuilt or retrained in order to diagnose new conditions. This procedure is both computationally expensive and time-consuming. To overcome these limitations, the proposed method in this study utilized a stacked DAE with a softmax regression classifier in the output layer. The results indicated that the proposed approach is robust to noise, capable of automatically learning representative features from unlabeled data, and achieves high performance in fault classification. In addition, the proposed method is capable of classifying new conditions by fine-tuning the trained architecture applying small amounts of labeled data from that new condition. This proves suitability towards autoships which are subjected to varying environmental and operating conditions. The proposed method was verified with a standard data set of bearing faults and compared to SVM and k-nearest neighbor algorithm.

Thirukovalluru et al. [122] also pointed out the importance of the feature extraction process in diagnostics systems. This study compares the classification performance of traditionally human-engineered features and stacked denoising SAE generated features. The human-engineered features extraction methods used in this analysis are Fast Fourier Transform (FFT) and Wavelet Packet Transform (WPT), and SVM and Random Forest (RF) are used as the classifiers. The stacked denoising SAE is a variation of the original AE that both utilize the strengths from DAE and SAE, namely, the input corruption process and the sparsity penalty term. The results of the experiments showed that the stacked denoising SAE generated features achieved higher classification performance than the human-engineered features methods at least once. The results were validated using five different data sets: air compressor monitoring, drill bit monitoring, steel plate monitoring, and two data sets of bearing fault-monitoring data.

High-quality labeled training data and expert knowledge are not easily obtained regarding induction motors due to environmental interference and inherent motor structure complexity. For that reason, Sun et al. [113] also proposed a stacked denoising SAE in order to improve induction motor fault classification by reducing the dependency of labeled data and expert knowledge. The input corruption process enhances the robustness of the automatically extracted features and the stability of the proposed architecture. The extracted features are then used to train a classifier. Both SVM and logistic regression (LR) are considered as the classifiers. The "dropout" technique [123] is also introduced in this study. This is a regularization technique invented in 2014, and it was integrated into the whole architecture to reduce overfitting in the training process. For verification, the effectiveness of proposed architecture was compared with three different BP-NNs.

C. Long-Short Term Memory

1) Introduction: Recurrent Neural Networks (RNNs) [26],

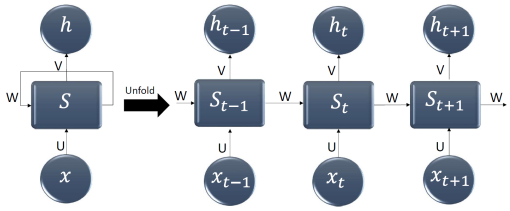


Fig. 5: RNN unfolded in time, adopted from [79]. The same weight matrices (U,V,W) are used at each time step.

[79], [94] are a group of neural networks used for tasks that involve sequential data. The popularity of RNNs emerged with the idea of connecting past information to the current task. In order to do so, traditional RNNs share the same weights (U,V,W) across several time steps, and this is the main difference compared to FNN. Weight sharing is important because a specific piece of information can occur at several positions within the sequential data [94]. RNNs are usually trained with the back-propagation algorithm to calculate the derivative of a total error with respect to all states, S_t, and all the parameters [79]. Figure 5 illustrates a simple model of this.

However, during the early 1990s, [124], [125] discovered a vanishing and exploding gradient problem. That is, when the shared (fixed) weight, W, is multiplied by itself several times, depend on magnitude, the product, W^t, will either vanish or explode [94]. Consequently, when the gap between previous relevant information and the present task becomes large, the information will be lost, and hence, the traditional RNN have difficulties of learning long-term dependencies.

One of the most popular approaches to reduce the difficulty of learning long-term dependencies is the LSTM. The original LSTM is a special kind of RNN that was first introduced by [126]. The initial idea of the LSTM architecture is to introduce a memory cell. This memory cell contains non-linear gating units in order to regulate the information flow in and out of the cell. By this, the memory cell is able to maintain its state over long durations, and the weights are conditioned on the context and not fixed. Thus, the time scale of integration can vary dynamically [94]. The literature provides several modifications and variations of the original LSTM; see [127] for a thorough review. Regarding recent applications to PHM, the vanilla LSTM with no peephole connections, originally described by [128], [129], is the most common choice. For that reason, the paragraphs below will discuss vanilla LSTM (hereinafter referred to as LSTM).

By introducing the memory cell, LSTMs are explicitly designed to learn long-term dependencies. Inside the memory cell, as illustrated in Figure 6, three non-linear gating units protect and regulate the cell state, $\boldsymbol{S}_t$. The gating units introduce a sigmoid layer, σ, in order to obtain an output value between 0 and 1, input weights $\boldsymbol{W}$, recurrent weights $\boldsymbol{R}$, and bias weights $\boldsymbol{b}$. The paragraphs below are based on a comprehensive blog post regarding the understanding of LSTM networks, [130].

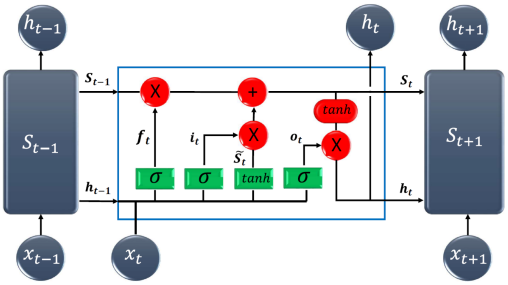


Fig. 6: Vanilla LSTM, adopted from [130]. The blue rectangle represents the memory cell.

The first gating unit is called the forget layer, and is defined as:

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_f\, \boldsymbol{x}_t + \boldsymbol{R}_f\, \boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \tag{17}$$

The forget layer determines which historical information the memory cell removes from the cell state. In this layer, an output value of 0 means to completely remove it, while an output value of 1 means to completely keep it. The second gating unit consists of two parts. The first part is called the input layer:

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_i\, \boldsymbol{x}_t + \boldsymbol{R}_i\, \boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \tag{18}$$

The input layer decides which values the memory cell will updated. The second part, is a tanh layer who creates a vector of new candidate state values, $\tilde{\boldsymbol{S}}_t$:

$$\tilde{\boldsymbol{S}}_t = tanh(\boldsymbol{W}_s\, \boldsymbol{x}_t + \boldsymbol{R}_s\, \boldsymbol{h}_{t-1} + \boldsymbol{b}_s) \tag{19}$$

In this way, the second gating unit determines what new information the memory cell is going to store in the cell state. Obviously, the next step is to update the previous cell state, $\boldsymbol{S}_{t-1}$, into the new cell state, $\boldsymbol{S}_t$:

$$\boldsymbol{S}_t = \boldsymbol{f}_t \otimes \boldsymbol{S}_{t-1} + \boldsymbol{i}_t \otimes \tilde{\boldsymbol{S}}_t \tag{20}$$

where, $\otimes$, denotes element-wise multiplication of two vectors. First, the previous state is multiplied by the output from forget layer, and then the new candidate state values are added, scaled by the output from the input layer, that is, how much each new candidate state value will be updated. The third and final gating unit decides the output. This also consists of two parts. The first part is the output layer:

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_o\, \boldsymbol{x}_t + \boldsymbol{R}_o\, \boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \tag{21}$$

The output layer determines which parts of the cell state the memory cell is going to output. Then, the second part will create a filtered version of the cell state in order to push the values between -1 and 1, and finally multiply it by the scaled output value from the output layer:

$$\boldsymbol{h}_t = \boldsymbol{o}_t \otimes tanh\,(\boldsymbol{S}_t) \tag{22}$$

Through this procedure, LSTMs have the ability to remove or add information to the cell state. The traditional RNN lack this ability, and hence, it will completely override cell states.

TABLE II: The C-MAPSS data set [106].

Data set	FD001	FD002	FD003	FD004
Timeseries training set	100	260	100	249
Timeseries test set	100	259	100	248
Operating conditions	1	6	1	6
Fault conditions	1	1	2	2

2) Recent Applications to PHM: LSTMs are highly capable of learning long-term dependencies and specially designed for sequential data. With respect to PHM applications, sequential data is a standard format of the input data, e.g temperature and vibration measurements [131]. For that reason, LSTMs are good candidates for RUL estimations because LSTMs might reveal hidden information in the data, as well as the past dependencies that may influence future events. The paragraphs below review applications of LSTM to PHM in the recent three years.

Chen et al. [132] applied LSTM in mechanical state predictions. The proposed method was divided into two steps. The first step, which had two parts, involved feature extraction methods to obtain the mechanical state characteristics. The empirical mode decomposition method was used to decompose bearing data into stationary signals. Then, the intrinsic mode function energy entropy was calculated based on the decomposition. This calculation was to characterize the mechanical state. The second step applied the LSTM network in order to make predictions. The results indicated that the mean square error index of the LSTM network was lower compared to SVM.

Accurate and reliable RUL estimations play a vital role for a PHM system. Traditional data-driven approaches, such as HMMs and traditional RNNs, encounter difficulties when modeling sequential data. Both approaches have issues with long-term dependencies. In addition, CNNs do not fully account for sequence information because of their segmented input. Based on this, Zheng et al. [33] proposed an LSTM approach to provide RUL estimations. The proposed architecture consists of multiple layers of LSTMs combined with multiple layers of FNN. The LSTM layers are good for temporal modeling and can reveal hidden patterns in the sequential input data. The FNN layers are then applied in order to map the LSTM features and predict the RUL. Multiple layers are used to discover the complex relationship within the sensor data. This study used both the C-MAPSS data set [106] and the PHM08 data set [102]. The results indicated that the proposed architecture outperformed the above-mentioned approaches. The proposed method was trained in a supervised manner by engineering and utilizing piece-wise linear RUL targets, as recommended by [133], for the training data sets.

A similar and more comprehensive study on the C-MAPSS data set [106] was conducted in [134]. In this work Wu. et al also proposed an LSTM approach for RUL estimations. The C-MAPSS data set consists of four subsets as shown in Table II. Both subset FD002 and FD004 involves several operating conditions. Therefore, a dynamic difference method was proposed in order to extract new features from inter-frame dynamic changes before the training procedure. These changes contain valuable degradation information, and hence, enable the LSTM approach to better control the underlying physical processes. The proposed method indicated improved performance compared to traditional RNN and gated recurrent unit LSTM (GRU-LSTM) [135].

Yuan et al. [136] proposed another LSTM approach to providing RUL estimations. However, the motivational factor in this study was to utilize the LSTM approach to build a common model for several different faults. In addition, the proposed approach was able to get RUL estimations and the probability of each fault at the same time. This feature makes it easy to design confidence intervals. The proposed approach was compared with traditional RNN and two variations of LSTM: GRU-LSTM and AdaBoost-LSTM. However, in all cases, the LSTM showed enhanced performance. The comparison used the C-MAPSS data set [106]. In this study, an SVM was used as an anomaly detector to create labels.

The majority of recent PHM applications based on DL have been focusing on either automatic feature extraction, classification, or regression. For that reason, Liao et al. [131] proposed a novel end-to-end deep architecture by stacking LSTM, FNN, and survival analysis. In this way, the proposed method integrates feature extraction and prediction as a single optimization task. This study utilized the LSTM as the first feature extraction layer. The reason for this is that the LSTM layer is able to handle the raw sequential input, and potentially discover past information that may influence future events. The extracted features will then be the input for the FNN layer, which makes it possible to further improve learning of the feature representation. Finally, the survival model learns the features and predicts the failure probability to indicate health conditions. Stochastic gradient descent is used as the learning optimization method for all the parameters. The proposed method showed promising results and was validated by a small data set of fleet mining haul trucks and by a large open source data set.

It should be noted that the proposed methods in the above LSTM-studies are based on supervised learning. In other words, trained on labeled training data. Nevertheless, in real-world applications, e.g. the maritime industry, high-quality labeled training data is hard to acquire, and large amounts of the data are unlabeled. For that reason, the first feature extraction layer, in any architecture, would have the advantage of utilizing unsupervised learning strategies, like the above mentioned DBNs or AEs. Gensler et al. [137] proposed an interesting approach that combines AE and LSTM. Specifically, the proposed approach combines automatic feature extraction from unlabelled training data with the temporal context utilization of the LSTM. The main idea is that after pre-training the AE in an unsupervised manner, the network architecture will be cut after the encoding side (bottleneck), and then the learned encoding will act as the input for the LSTM. Finally, the AE-LSTM architecture will be fine-tuned, where only the LSTM weights are trained, to produce the desired output. The proposed approach showed enhanced prediction performance compared to MLP, LSTM, and DBN. Although this study is targeting solar power forecasting, the proposed method has the potential to provide inspiration towards future intelligent PHM

systems applied to autoships.

Malhotra et al. [138] provides another interesting approach. Prognostic approaches for RUL estimations are generally based on the assumption that the health degradation curve follows a specific shape, e.g. exponential or linear. In this study it was observed that such assumptions are not well-suited for real-world applications. In addition, most prognostic approaches are application-dependent, meaning they are not robust towards new conditions. Based on these observed limitations, this study proposed an unsupervised approach, by utilizing an LSTM encoder-decoder, to obtain a health index (HI) for a multi-sensor time-series data system. The HI is then used to learn a regression model for RUL estimations. Briefly explained, the LSTM encoder learns a representation of the input time-series. Then, the LSTM decoder applies this representation to reconstruct the time-series using the current hidden state and the predicted value of the previous time-step. See [27] for a deeper understanding of the LSTM encoder-decoder method. The study used the C-MAPSS data set [106] and a milling machine data set, as well as a case study on real-world data, for validation. Surprisingly, the proposed approach showed improved performance compared to approaches that rely on assumptions about health degradation.

D. Convolutional Neural Network

1) Introduction: CNNs [20], [79], [94], [139] are designed for processing multiple arrays of 1D, 2D, or 3D grid-structured topology data. Examples of 1D, 2D, and 3D grid are: time-series data taking samples at systematic time intervals, pixels in image data, and video or volumetric images, respectively. CNNs have been inspired by earlier work on time-delay neural networks (TDNNs) [140]. Primarily, TDNNs are one-dimensional CNNs applied to time-series and use shared weights in a temporal dimension in order to reduce learning computation requirements. In addition to shared weights, convolution, pooling, and multi-layer architectures are the important ideas of CNNs. In fact, CNNs are the first truly DL technique to successfully train multiple layers [17].

A CNN can briefly be defined as a neural network that uses the mathematical operation convolution instead of general matrix multiplication in at least one of its layers [94]. With respect to mathematical understanding, convolution is an operation to combine two functions of a real-valued argument by measuring the overlap of two functions when one proceeds over the other. The discrete 1D convolution operation can be defined as:

$$\boldsymbol{S}(t) = (\boldsymbol{I} \cdot \boldsymbol{K})(t) = \sum_{a} \boldsymbol{I}(a)\boldsymbol{K}(t-a) \tag{23}$$

where t is the discretized time index, $\boldsymbol{I}$ is the input, $\boldsymbol{K}$ is a kernel (filter), and a is the finite number of array elements. The output, $\boldsymbol{S}(t)$, is usually referred to as the feature map. Expanding Equation 23, the discrete 2D convolution operation can be defined as:

$$\boldsymbol{S}(i,j) = (\boldsymbol{I} \cdot \boldsymbol{K})(i,j) = \sum_{m}\sum_{n} \boldsymbol{I}(m,n)\boldsymbol{K}(i-m, j-n) \tag{24}$$

where i and j are the discretized time indexes, and m and n are the finite number of array elements in each of the two dimensions. However, in the context of CNNs in practice, the standard discrete convolutional operations are moderately different. The reason for this is that the operation consists of several convolutions in parallel in order to extract many types of features at several locations in the input data. In addition, the input is normally a grid of vector-valued observations, and not only a grid of real values. Full derivations of the above equations, as well as practical variations of the standard discrete convolutional operation, appear in [94].

The convolution operation exploits three prime features of CNNs in the learning process [20], [94]. First, CNNs have sparse interactions. This is realized by making the kernel smaller than the input, and hence, CNNs needs to store fewer parameters compared with traditional FNN. This is because traditional FNN uses general matrix multiplication between layers, that is, every output unit interacts with every input unit. The sparse interaction feature reduces computational and memory requirements, as well as increasing statistical efficiency. Second, shared weights between several functions in the architecture further reduce the memory requirement and the complexity of the network. Finally, shared weights result in equivariance in the layers. That is, the output will change according to the input.

Researchers generally use one of two sets of terminology for describing the conceptual structure of CNNs. The first is the complex layer terminology [94], which this survey paper employs. The second is the simple layer terminology where every processing step is considered to be a separate layer. [17], [20] further describes simple layer terminology.

Complex layer terminology understands each layer in a CNN as having three processing steps, as illustrated in Figure 7. The first step, $C1$, executes several convolution operations in parallel in order to produce feature maps with linear activations. These activations are then processed by non-linear activation functions, σ, in the second step. The sigmoid activation function or the ReLU are common choices. In the third step (usually called sub-sampling), a pooling function, $S1$, is used to further adjust the output by calculating summary statistics of the nearby outputs. Regular choices are max-pooling [141] and average-pooling. Max-pooling calculates the maximum output in a rectangular neighborhood, while average-pooling calculates the average output. All pooling functions further reduce the dimensionality of the representations, and at the same time, generates an invariance to small translations of the input, namely, if the input to the pooling function changes, most of the pooled outputs do not change [79], [94]. This layer procedure repeats itself in the next layers. Finally, the outputs from the last layer are rasterized and presented as a single input vector to a traditional FNN in order to perform functions such as classification or regression.

The training procedure of CNNs is introduced in [139]. It is similar to standard back-propagation training performed on FNN, but the reduced number of parameters and shared weights in CNNs improve the training efficiency. In addition, CNNs are capable of handling raw input data, and hence, pre-processing is rare. This means that CNNs are less dependent

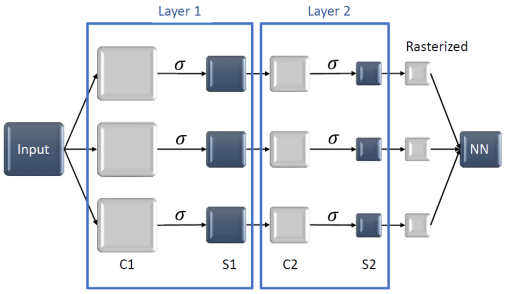


Fig. 7: Conceptual structure of CNN.

on prior knowledge and human-engineered feature extraction methods which appears to be suitable towards future intelligent PHM systems applied to autoships. The convolutional operations also enable CNNs to process inputs with varying spatial extents [94].

2) Recent Applications to PHM: According to [94], CNNs have been most successful on 2D and 3D grid-structured topology data, like object recognition [142] and face recognition [143], respectively. Nevertheless, recent applications to PHM have applied 1D grid-structured topology for sequential data and shown enhanced performance compared with traditional machine learning algorithms. For that reason, CNN applications to PHM in the years between 2016 and 2018 will be reviewed in the following paragraphs.

In order for CNNs to act as an effective feature extraction method for raw industrial signals in PHM applications, the successful applications of CNNs to 2D and 3D grid topology data have to be modified. The reason for this is that raw industrial signals are usually 1D time-series with hidden information behind strong intervals and deep correlations amid various time points. Thus, to fully exploit all the information in the signal, it is necessary to consider the relationship between signals in diverse locations. Liu et al. [144] proposed a novel dislocated time-series CNN (DTS-CNN) diagnostics approach. The proposed approach utilizes a dislocated layer as the initial layer in order to extract the relation between periodic vibration signals with varying intervals. After the initial dislocated layer, there are two layers with convolution- and max-pooling steps, then a fully connected softmax classifier is used for classification. For verification, an electric machine fault simulator with different operating conditions was used in two experiments. The DTS-CNN showed improved performance, compared to traditional CNN and wavelet packet SVM, due to its robustness under different non-stationary operating conditions. In addition, the proposed approach was capable of automatically extracting features from raw input data, and hence, less dependent on human-engineered feature extraction methods and prior knowledge.

Jing et al. [145] proposed a CNN approach to provide automatic feature extraction and fault diagnosis. The motivational factor in this study was three limitations of traditional diagnostic algorithms: First, their inability to handle raw input data. This requires human-engineered feature extraction methods, which means the diagnostics accuracy heavily depends on domain expertise and prior knowledge. Second, feature extraction methods are application-dependent. In this way, the diagnostics results are sensitive to changes in the mechanical system. Third and finally, traditional diagnostics algorithms lack the ability to mine new features. In this study, CNN was applied to automatically learn features from raw vibration data in the time domain, frequency domain, and time-frequency domain, and then conduct diagnostics of gearboxes. 1D segments are collected from the raw vibration data as the input for the CNN. This study uses one layer with convolution and pooling steps because this approach showed higher accuracy and more stable results than other configurations. In the end, a fully connected softmax layer was used for classification. The proposed approach was validated with a publicly available data set for gearboxes. The comparison uses manual feature extraction methods from each domain and traditional diagnostics approaches such as FNN, SVM, and RF. The results indicated that the proposed approach outperformed the comparative methods.

Traditional prognostics algorithms are usually based on shallow architectures. Consequently, they lack the ability to capture more complex relationships between the sensory data and RUL estimations. In addition, traditional algorithms do not have the ability to automatically learn salient features. Based on these restrictions, Babu et al. [88] proposed a novel CNN-based regression approach for RUL estimations from multi-variate time-series sensor signals. However, applying CNN to multiple channels of time-series signals has two main challenges that apply to the processing steps: they need to be applied along temporal dimensions, and they need to be shared among multiple sensors. To cope with these challenges, this study adopted a sliding window strategy in order to create segmented collections of the time-series signals. Each segment is then fed into two layers with convolution- and average-pooling steps. The first convolution step is two-dimensional, while the second and the two pooling steps are one-dimensional. These processing steps automatically capture salient features of the sensor signals at different time scales, and hence, the features extracted are task dependent and not human-engineered. Finally, all salient features are systematically unified and mapped into the RUL estimation of a traditional FNN regressor. Both the PHM08 data set [101] and the C-MAPSS data set [106] were used to validate the results. The proposed approach showed enhanced performance compared to MLP, SVM, and RVM. It should be noted that a piece-wise linear RUL target function has been used in this study. This means a supervised training procedure with target run-to-failure data. However, Zheng et al. [33] claim that LSTM outperformed the CNN approach in their study because the RUL estimations in the CNN approach are built based on independent sliding windows. Sliding windows are in fact time-dependent with respect to RUL estimations, and hence, the CNN approach does not fully consider sequence information.

In 2018, Li et al. [34] proposed a new CNN approach

that has shown improved RUL prediction performance on the C-MAPSS data set [106] compared to both the CNN approach in [88] and the LSTM approach in [33]. Similar to [88], this CNN approach also prepares the input data in two dimensions (sequence length $\times$ selected features) and utilize a time-window strategy. However, in contrast to [88], this CNN approach performs all the convolution steps in one dimension. In this way, the CNN is able to learn high-level representations of each raw feature from the very start rather than learning the spatial relationship of several features and then extracting information. Additionally, the proposed approach employs the advanced regularization technique "dropout" [123] and the adaptive learning rate method "Adam" [146], both invented in 2014.

E. Discussion and Summary

Table III provides a structured summary of the recent PHM applications based on DL reviewed in this survey paper. This representative set of applications has been selected to provide insight into the current state-of-the-art and to encourage important directions for future research towards intelligent PHM systems suitable for autoships. DBNs, AEs, DAEs, SAE, LSTMs, and CNNs have been explicitly chosen as the DL techniques primarily because they are well-established and show great promise for future developments.

With respect to autoships and the maritime industry more generally, future intelligent PHM systems will have to adapt to highly varying operational and environmental conditions, as the maritime environment is harsh and uncertain. Additionally, such systems needs to provide automatic pre-processing and dimensionality reduction schemes in order to effectively process massive data flows of high-dimensional and unstructured data. Based on the reviews this paper has discussed, DBNs, AEs, DAEs, and SAEs seem like promising ways to address these challenges since these DL techniques provide an unsupervised learning procedure as an initial pre-training step. This procedure will automatically capture abstract, important statistical structures and reduce dimensionality of raw unlabeled input data. As a result, DL techniques minimize the need for human-crafted feature extraction methods in the data processing stage, reduce the need for large amounts of high-quality labeled training data, and have the potential to be applied to new conditions by fine-tuning the trained final architecture using a much smaller labeled data set from that new condition. [104], [107], [113], [116], [137] intensively investigates the effectiveness of these qualities.

Another important concern is the fact that sensor data is going to be the most common data type format for future intelligent PHM systems used in autoships. LSTMs are highly capable of learning long-term dependencies that may influence future events, they are specially designed for sequential data, and might discover hidden data information. [33], [134] suggests the strengths of the LSTM. Furthermore, the CNN approach in [34] seems to be highly suitable for sequential data. Actually, it outperformed both an equivalent CNN approach in [88] and the LSTM approach in [33] on the C-MAPSS data set [106].

TABLE III: Recent PHM applications based on DL (the years between 2013 and 2018).

DL technique	Author. Reference	Year	PHM application
DBN	Deutsch et al. [107]	2017	Automatic feature extraction and failure prognostics: Rotating components
	Li et al. [104]	2016	Automatic feature extraction and fault diagnostics: Rotary machinery
	Zhang et al. [105]	2016	Automatic feature extraction and failure prognostics: C-MAPSS data set [106]
	Liao et al. [108]	2016	Automatic feature extraction and failure prognostics: Rotating systems
	Jiang et al. [109]	2016	Automatic feature extraction and Time-series prediction: Complex systems
	Yang Fu et al. [100]	2015	Automatic feature extraction: Cutting state monitoring
	Tamilselvan et al. [32]	2013	Fault diagnostics: PHM08 data set [101] Electric power transformer
	Tran et al. [103]	2013	Fault diagnostics: Reciprocating compressor valves
DAE	Xia et al. [116]	2017	Automatic feature extraction and fault diagnostics: Motor bearings
SAE	Sun et al. [113]	2016	Automatic feature extraction and fault diagnostics: Induction motor
DAE/SAE	Thirukovalluru et al. [122]	2016	Automatic feature extraction and fault diagnostics: Air compressor monitoring Drill bit monitoring Steel plate monitoring Bearing fault monitoring
AE	Lu et al. [119]	2015	Automatic feature extraction and fault diagnostics: Rolling bearing data
AE	Jia et al. [121]	2015	Automatic feature extraction and fault diagnostics: Rolling element bearing Planetary gearbox
RBM/AE	Ma et al. [120]	2014	Automatic feature extraction and fault diagnostics: Power transformers Circuit breakers
LSTM	Wu et al. [134]	2018	Failure prognostics: C-MAPSS data set [106]
	Chen et al. [132]	2017	Failure prognostics: Bearings
	Zheng et al. [33]	2017	Failure prognostics: PHM08 data set [101] C-MAPSS data set [106]
	Yuan et al. [136]	2016	Failure prognostics: C-MAPSS data set [106]
	Liao et al. [131]	2016	Feature extraction and failure prognostics: Mining haul trucks
AE/LSTM	Gensler et al [137]	2016	Automatic feature extraction and failure prognostics: Solar power plants
LSTM	Malhotra et al. [138]	2016	Automatic feature extraction and failure prognostics: C-MAPSS data set [106] Milling machine
CNN	Li et al. [34]	2018	Automatic feature extraction and failure prognostics: C-MAPSS data set [106]
	Liu et al. [144]	2017	Automatic feature extraction and fault diagnostics: Electric machine fault simulator
	Jing et al. [145]	2017	Automatic feature extraction and fault diagnostics: Gearbox
	Babu et al. [88]	2016	Automatic feature extraction and failure prognostics: PHM08 data set [101] C-MAPSS data set [106]

V. Discussion

The previous sections discuss the benefits and challenges of applying PHMDL in autoships in addition to specific introductions and reviews of well-established DL techniques with respect to recent applications to PHM. These sections are intended to enlighten the reader with both opportunities and a more theoretical and practical understanding on how PHMDL can be applied to autoships. This section gives more general discussions concerning suitable deep architectures in order to address the challenges and exploit the benefits. Finally, existing problems and future research opportunities are introduced.

A. Suitable Deep Architectures

The majority of the reviewed studies in Section IV, either utilize DBN, AEs, LSTM, or CNN as the selected DL technique. However, different DL techniques can be stacked in order to further exploit the advantages and reduce the drawbacks of each DL technique. Gensler et al. [137] combines AE and LSTM to both make use of the automatic feature extraction from unlabeled input data and the temporal context utilization of LSTM. This approach has served as inspiration for this paper's proposed deep architecture for intelligent PHM systems for autoships.

PHMDL systems in autoships demands automatic preprocessing and dimensionality reduction schemes due to varying operational and environmental conditions and massive data flows of high-dimensional and unstructured data. Thus, the unsupervised DL techniques, DBN and the modern variations of the original AE, DAE, and SAE, have great potential to be applied as the first layer of a deep architecture. More specifically, the stacked denoising SAE [113], [122] is an encouraging opportunity. The reason for this is that both the stochastic corruption process, $\tilde{x} \sim q(\tilde{x}|x)$, in DAEs and the sparsity penalty term, $\Omega(h)$, in SAEs appears particularly suitable for autoships and the maritime environment. This combination will give the first layer the potential to provide robustness towards noisy sensor data and to control the number of active hidden neurons. This will increase both the robustness of the system and the efficiency of the automatic feature extraction process. The increased efficiency reflects the reduced number of active neurons in the hidden layers, as it costs energy to activate neurons and to send signals between them. Actually, human brains seem to minimize such computational costs in a similar manner [89].

The lack of onboard crew members in autoships creates higher demands for scheduling maintenance procedures to the next appropriate port of call [31]. Consequently, PHMDL systems have to provide reliable RUL estimations of relevant components and sub-components. Additionally, sensor data will be the most common data type format for PHMDL systems in autoships. Thus, LSTM is a quite promising candidate to act as the following layer or layers of a deep architecture. LSTM is especially designed for sensor data and was cpable of revealing hidden information and learning long-term dependencies within sensor data with multiple operating and fault conditions in [33]. This proves the potential to enhance RUL estimations. Finally, a traditional FNN layer can be applied in order to map all extracted features and provide RUL estimations. However, it should be noted that any RUL estimation should include associated confidence intervals in order to provide reliable and trustworthy outputs. This is necessary to help autoships to optimize maintenance planning.

The well-proven regularization technique, "dropout" [123], extends the idea of the DAE. "Dropout" randomly drop units during training, and hence, regularize a network by adding noise to its hidden units. In this way, the network learns to make generalized representations of the input data, which enhances the feature extraction ability. "Dropout" can be applied to all hidden layers in both LSTMs and FNNs. Therefore, it should be considered as part of the proposed deep architecture. It should be noted that "dropout" should only be applied to the non-recurrent connections in LSTMs.

B. Existing Problems and Future Research Opportunities

Even if unsupervised DL techniques are applied in the first layer, the deep architecture will still require a reduced amount of labeled training data in order to perform supervised classification or regression in the final layer. The labeled training data is necessary to fine-tune the whole architecture with respect to the final classification or regression task. As a consequence, the supervised learning procedure assumes that input events are independent of earlier output events [89]. As a result, the DL techniques reviewed in this survey paper do not involve learning to act in totally unknown environments. We assume that this feature will be extremely useful in future intelligent PHM systems applied to autoships due to the lack of fault labels and run-to-failure data of components and sub-components [35].

Additionally, according to [147], current DL techniques are insufficient for fixed network architectures. This is because recently introduced tri-traversal theory proves that DL techniques will need to adapt at three levels of organization (T3-structure), that is, be equipped with intelligent procedures that rapidly adjust their network architectures, in order to deal with the complexity of the maritime environment as well as other contexts.

Solving this problem through a combination of DL and reinforcement learning (RL) where there is no supervised teacher is an exciting research objective. Briefly explained, RL enables learning from feedback received through interactions with an external environment [18]. In the application of autoships, we propose that this environment could involve several environmental and operating conditions. Assuming some typical values of the sensor data in each condition, RL is able to search for possible inputs and outputs in order to maximize a reward [148], e.g. the performance of the deep architecture. Based on the observed rewards, RL is able to obtain the optimal, or nearly optimal, deep architecture structure for each condition. Consequently, RL adaptively adjusts the hyper-parameters in the deep architecture, e.g. learning rate, number of hidden layers and nodes, "dropout" rate, etc. Its policy requires RL to decide whether to use the hyper-parameters that gave the highest reward last time for a specific condition or to try out different hyper-parameters in hope of providing even better performance [148].

Another interesting research objective is the digital twin (DT). Today, the concept of the DT is one of the most-sought research objectives in the maritime industry [149]. A DT will consist of a series of simulation models that are continuously updated to mirror their real-life twins [150], e.g. an autoship. In this way, a DT differs from a generic model because it is specific to its physical counterpart. The DT model must also reflect changes involving the physical autoship. A DT will include a system and data information model, simulation models and data analytics, and dependability and performance models [35]. The DT concept allows new design paradigms where different stakeholders will be able to contribute to the creation of a DT with specific models and evaluate in advance how the autoship will operate in different scenarios [90]. In this way, the DT is able to collect highly important prior knowledge, in addition to the knowledge acquired through sensors etc. in the current state. The new design paradigms enable DTs to build extensive databases regarding run-to-failure data of critical and relevant components and sub-components. This will be highly beneficial and necessary for successful implementations of future intelligent PHM systems on autoships.

According to [82], DL has the property that it gets better as it receives more data. This property also applies to DTs because, over time, they will be increasingly more detailed and continuously updated with sensor data. This quality accelerates developments towards data-driven approaches suitable for industrial big data, such as DL techniques.

VI. Conclusion

Autoships are a provident and rapidly expanding research field. In order to operate and maintain complex and integrated systems in a safe, efficient, and cost-beneficial manner, autoships are expected to include intelligent PHMDL systems. PHMDL have the potential to reduce built-in redundancy and to provide more robust and reliable maintenance scheduling.

In this survey paper, well-established DL techniques have been introduced and reviewed with respect to recent PHM applications. In these reviews, DL techniques have demonstrated that they are a superior alternative to human-crafted feature extraction methods combined with traditional machine learning algorithms in many practical PHM problems. Hence, DL techniques are highly suitable to be applied to diagnostics and prognostics tasks in future intelligent PHM systems in the age of big data. The main intention of this survey paper is to support creativity and inspiration in explorations of PHMDL possibilities in the maritime industry, particularly autoships. To guide future researchers, this paper introduced and discussed the benefits and challenges of implementation of PHMDL in autoships, and the maritime industry in general. In highly varying operational and environmental conditions and massive data flow, DL techniques will be advantageous due to unsupervised learning procedures that automatically extract high-level abstract features and, at the same time, reduce the dimensionality of raw unlabeled input data. In this way, PHMDL is less application-dependent than traditional approaches, and hence, has the potential to operate in different conditions.

This paper has also provided more general discussions, concerning suitable deep architectures, existing problems, and future research opportunities It appears that DL, RL, and DT all have the potential to push the development of autoships and intelligent PHM systems to the next level.

Acknowledgment

This work was supported by the Norwegian University of Science and Technology within the Department of Ocean Operations and Civil Engineering under project no. 90329106. The authors would like to thank Digital Twins For Vessel Life Cycle Service (DigiTwin) and the Research Council of Norway, grant no. 280703.

References

[1] O. Levander, "Autonomous ships on the high seas," *IEEE Spectrum*, vol. 54, no. 2, pp. 26–31, February 2017.

[2] L. Kretschmann, H.-C. Burmeister, and C. Jahn, "Analyzing the economic benefit of unmanned autonomous ships: An exploratory cost-comparison between an autonomous and a conventional bulk carrier," *Research in Transportation Business & Management*, 2017.

[3] E. Jokioinen, "Remote and autonomous ships - the next steps: Introduction," *Rolls-Royce, Buckingham Gate, London: The Advanced Autonomous Waterborne Applications (AAWA)*, pp. 4–14, 2016.

[4] K. E. Knutsen, G. Manno, and B. J. Vartdal, "Beyond condition monitoring in the maritime industry," *DNV GL Strategic Research & Inovation Position Paper*, 2014.

[5] R. Kothamasu, S. H. Huang, and W. H. VerDuin, "System health monitoring and prognostics – a review of current paradigms and practices," *The International Journal of Advanced Manufacturing Technology*, vol. 28, no. 9, pp. 1012–1024, 2006.

[6] T. M. Allen, "Us navy analysis of submarine maintenance data and the development of age and reliability profiles," *Department of the Navy SUBMEPP*, 2001.

[7] B.-M. Batalden, P. Leikanger, and P. Wide, "Towards autonomous maritime operations," in *2017 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*. IEEE, 2017, pp. 1–6.

[8] J. Lee, F. Wu, W. Zhao, M. Ghaffari, L. Liao, and D. Siegel, "Prognostics and health management design for rotary machinery systemsreviews, methodology and applications," *Mechanical Systems and Signal Processing*, vol. 42, no. 1, pp. 314–334, 2014.

[9] A. Brandsæter, G. Manno, E. Vanem, and I. K. Glad, "An application of sensor-based anomaly detection in the maritime industry," in *2016 IEEE International Conference on Prognostics and Health Management (ICPHM)*. IEEE, 2016, pp. 1–8.

[10] X. Xu, S. Fan, H. Huang, H. Zhu, and Q. Wen, "Research on phm technology application of ship maintenance program optimization," in *2013 IEEE Conference on Prognostics and Health Management (PHM)*, June 2013, pp. 1–6.

[11] D. McDonnell, N. Balfe, S. Al-Dahidi, and G. O'Donnell, "Designing for human-centred decision support systems in phm," in *European Conference of the Prognostics and Health Management Society*. IEEE, 2014.

[12] S. Yin, S. X. Ding, X. Xie, and H. Luo, "A review on basic data-driven approaches for industrial process monitoring," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 11, pp. 6418–6428, 2014.

[13] P. Baraldi, F. D. Maio, D. Genini, and E. Zio, "Comparison of data-driven reconstruction methods for fault detection," *IEEE Transactions on Reliability*, vol. 64, no. 3, pp. 852–860, 2015.

[14] O. Geramifard, J.-X. Xu, C. K. Pang, J. Zhou, and X. Li, "Data-driven approaches in health condition monitoringa comparative study," in *2010 8th IEEE International Conference on Control and Automation (ICCA)*. IEEE, 2010, pp. 1618–1622.

[15] 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.

[16] D. Yu and L. Deng, "Deep learning and its applications to signal and information processing [exploratory dsp]," *IEEE Signal Processing Magazine*, vol. 28, no. 1, pp. 145–154, 2011.

[17] I. Arel, D. C. Rose, and T. P. Karnowski, "Deep machine learning - a new frontier in artificial intelligence research [research frontier]," *IEEE computational intelligence magazine*, vol. 5, no. 4, pp. 13–18, 2010.

[18] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, "A survey of machine learning for big data processing," *EURASIP Journal on Advances in Signal Processing*, vol. 2016, no. 1, p. 67, 2016.

[19] X. W. Chen and X. Lin, "Big data deep learning: Challenges and perspectives," *IEEE Access*, vol. 2, pp. 514–525, 2014.

[20] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, no. Supplement C, pp. 11–26, 2017.

[21] Z. H. Ling, S. Y. Kang, H. Zen, A. Senior, M. Schuster, X. J. Qian, H. M. Meng, and L. Deng, "Deep learning for acoustic modeling in parametric speech generation: A systematic review of existing techniques and future trends," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 35–52, May 2015.

[22] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[23] K. Noda, Y. Yamaguchi, K. Nakadai, H. G. Okuno, and T. Ogata, "Audio-visual speech recognition using deep learning," *Applied Intelligence*, vol. 42, no. 4, pp. 722–737, 2015.

[24] Q. V. Le, "Building high-level features using large scale unsupervised learning," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2013, pp. 8595–8598.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[26] I. Sutskever, J. Martens, and G. E. Hinton, "Generating text with recurrent neural networks," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 1017–1024.

[27] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'14. Cambridge, MA, USA: MIT Press, 2014, pp. 3104–3112.

[28] G. Wu, W. Lu, G. Gao, C. Zhao, and J. Liu, "Regional deep learning model for visual tracking," *Neurocomputing*, vol. 175, no. Part A, pp. 310–323, 2016.

[29] M. Längkvist, L. Karlsson, and A. Loutfi, "A review of unsupervised feature learning and deep learning for time-series modeling," *Pattern Recognition Letters*, vol. 42, pp. 11–24, 2014.

[30] G. Zhao, G. Zhang, Q. Ge, and X. Liu, "Research advances in fault diagnosis and prognostic based on deep learning," in *Prognostics and System Health Management Conference (PHM-Chengdu)*. IEEE, 2016, pp. 1–6.

[31] R. Jalonen, R. Tuominen, and M. Wahlström, "Remote and autonomous ships - the next steps: Safety and security in autonomous shipping - challenges for research and development," *Rolls-Royce, Buckingham Gate, London: The Advanced Autonomous Waterborne Applications (AAWA)*, pp. 56–73, 2016.

[32] P. Tamilselvan and P. Wang, "Failure diagnosis using deep belief learning based health state classification," *Reliability Engineering & System Safety*, vol. 115, no. Supplement C, pp. 124–135, 2013.

[33] S. Zheng, K. Ristovski, A. Farahat, and C. Gupta, "Long short-term memory network for remaining useful life estimation," in *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*. IEEE, 2017, pp. 88–95.

[34] 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.

[35] A. S. Zymaris, Ø. Å. Alnes, K. E. Knutsen, and N. M. Kakalis, "Towards a model-based condition assessment of complex marine machinery systems using systems engineering," in *PHM Europe, Bilbao, Spain*, 2016, Conference Proceedings.

[36] P. W. Kalgren, C. S. Byington, M. J. Roemer, and M. J. Watson, "Defining phm, a lexical evolution of maintenance and logistics," in *2006 IEEE Autotestcon*, Sept 2006, pp. 353–358.

[37] M. Jouin, R. Gouriveau, D. Hissel, M.-C. Pra, and N. Zerhouni, "Prognostics and health management of pemfc state of the art and remaining challenges," *International Journal of Hydrogen Energy*, vol. 38, no. 35, pp. 15 307–15 317, 2013.

[38] M. A. Taie, M. Diab, and M. ElHelw, "Remote prognosis, diagnosis and maintenance for automotive architecture based on least squares support vector machine and multiple classifiers," in *2012 IV International Congress on Ultra Modern Telecommunications and Control Systems*, Oct 2012, pp. 128–134.

[39] A. Ismail and W. Jung, "Recent development of automotive prognostics," *Korean Reliability Society*, pp. 147–153, 2015.

[40] P. L. Dussault, "Creating a closed loop evironment for condition based maintenance plus (cmb+) and prognostics health management," in *2007 IEEE Autotestcon*, Sept 2007, pp. 327–331.

[41] F. Camci, G. S. Valentine, and K. Navarra, "Methodologies for integration of phm systems with maintenance data," in *2007 IEEE Aerospace Conference*, March 2007, pp. 1–9.

[42] K. M. Janasak and R. R. Beshears, "Diagnostics to prognostics - a product availability technology evolution," in *2007 Annual Reliability and Maintainability Symposium*, Jan 2007, pp. 113–118.

[43] J. Lee, M. Ghaffari, and S. Elmeligy, "Self-maintenance and engineering immune systems: Towards smarter machines and manufacturing systems," *Annual Reviews in Control*, vol. 35, no. 1, pp. 111–122, 2011.

[44] A. K. S. 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.

[45] W. Elghazel, J. Bahi, C. Guyeux, M. Hakem, K. Medjaher, and N. Zerhouni, "Dependability of wireless sensor networks for industrial prognostics and health management," *Computers in Industry*, vol. 68, no. Supplement C, pp. 1–15, 2015.

[46] G. Vachtsevanos, F. Lewis, M. Roemer, A. Hess, and B. Wu, "Signal processing and database management systems," in *Intelligent Fault Diagnosis and Prognosis for Engineering Systems*. John Wiley & Sons, Inc., 2007, ch. 4, pp. 95–171. [Online]. Available: http://dx.doi.org/10.1002/9780470117842.ch4

[47] Z. K. Peng and F. L. Chu, "Application of the wavelet transform in machine condition monitoring and fault diagnostics: a review with bibliography," *Mechanical Systems and Signal Processing*, vol. 18, no. 2, pp. 199–221, 2004.

[48] G. Niu and B.-S. Yang, "Intelligent condition monitoring and prognostics system based on data-fusion strategy," *Expert Systems with Applications*, vol. 37, no. 12, pp. 8831–8840, 2010.

[49] Y. Lei, "Signal processing and feature extraction," in *Intelligent*

Fault Diagnosis and Remaining Useful Life Prediction of Rotating Machinery. Butterworth-Heinemann, 2017, ch. 2, pp. 17–66.

[50] G. Vachtsevanos, F. Lewis, M. Roemer, A. Hess, and B. Wu, "Fault diagnosis," in *Intelligent Fault Diagnosis and Prognosis for Engineering Systems*. John Wiley & Sons, Inc., 2007, ch. 5, pp. 172–283.

[51] E. Zio and G. Gola, "Neuro-fuzzy pattern classification for fault diagnosis in nuclear components," *Annals of Nuclear Energy*, vol. 33, no. 5, pp. 415–426, 2006.

[52] Y. Lei, "Introduction and background," in *Intelligent Fault Diagnosis and Remaining Useful Life Prediction of Rotating Machinery*, Y. Lei, Ed. Butterworth-Heinemann, 2017, ch. 1, pp. 1 – 16.

[53] G. Vachtsevanos, F. Lewis, M. Roemer, A. Hess, and B. Wu, "Fault prognosis," in *Intelligent Fault Diagnosis and Prognosis for Engineering Systems*. John Wiley & Sons, Inc., 2007, ch. 6, pp. 284–354.

[54] T. Khawaja, G. Vachtsevanos, and B. Wu, "Reasoning about uncertainty in prognosis: a confidence prediction neural network approach," in *NAFIPS 2005. Annual Meeting of the North American Fuzzy Information Processing Society*. IEEE, 2005, pp. 7–12.

[55] D. A. Tobon-Mejia, K. Medjaher, N. Zerhouni, and G. Tripot, "A data-driven failure prognostics method based on mixture of gaussians hidden markov models," *IEEE Transactions on Reliability*, vol. 61, no. 2, pp. 491–503, June 2012.

[56] J. Z. 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.

[57] G. W. Vogl, B. A. Weiss, and M. Helu, "A review of diagnostic and prognostic capabilities and best practices for manufacturing," *Journal of Intelligent Manufacturing*, 2016.

[58] B. Sun, S. Zeng, R. Kang, and M. G. Pecht, "Benefits and challenges of system prognostics," *IEEE Transactions on Reliability*, vol. 61, no. 2, pp. 323–335, 2012.

[59] J. Liu, W. Wang, F. Ma, Y. B. Yang, and C. S. Yang, "A data-model-fusion prognostic framework for dynamic system state forecasting," *Engineering Applications of Artificial Intelligence*, vol. 25, no. 4, pp. 814–823, 2012.

[60] M. Tahan, E. Tsoutsanis, M. Muhammad, and Z. A. Abdul Karim, "Performance-based health monitoring, diagnostics and prognostics for condition-based maintenance of gas turbines: A review," *Applied Energy*, vol. 198, pp. 122–144, 2017.

[61] T. Sutharssan, S. Stoyanov, C. Bailey, and C. Yin, "Prognostic and health management for engineering systems: a review of the data-driven approach and algorithms," *The Journal of Engineering*, vol. 1, no. 1, 2015.

[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] M. J. Roemer, C. S. Byington, G. J. Kacprzynski, and G. Vachtsevanos, "An overview of selected prognostic technologies with application to engine health management," *ASME Paper No. GT2006-90677*, 2006.

[64] D. J. Power and R. Sharda, "Decision support systems," in *Springer Handbook of Automation*, S. Y. Nof, Ed. Springer Berlin Heidelberg, 2009, ch. 87, pp. 1539–1548.

[65] N. Iyer, K. Goebel, and P. Bonissone, "Phm decision support under uncertainty," in *Annual conference of the prognostics and health management society*, 2016, Conference Proceedings.

[66] ——, "Framework for post-prognostic decision support," in *2006 IEEE Aerospace Conference*. IEEE, 2006, pp. 10–pp.

[67] D. J. Power, "Web-based and model-driven decision support systems: concepts and issues," *AMCIS Proceedings*, p. 387, 2000.

[68] "Nist big data interoperability framework: Volume 1, definitions," 2015. [Online]. Available: http://dx.doi.org/10.6028/NIST.SP.1500-1

[69] L. Deng, "Expanding the scope of signal processing [from the editor]," *IEEE Signal Processing Magazine*, vol. 25, no. 3, pp. 2–4, May 2008.

[70] L. Rabiner and B. Juang, "An introduction to hidden markov models," *IEEE ASSP Magazine*, vol. 3, no. 1, pp. 4–16, Jan 1986.

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

[72] J. Håstad, *Computational Limitations of Small-depth Circuits*. Cambridge, MA, USA: MIT Press, 1987.

[73] E. Allender, "Circuit complexity before the dawn of the new millennium," in *Foundations of Software Technology and Theoretical Computer Science: 16th Conference Hyderabad, India, December 18–20, 1996 Proceedings*, V. Chandru and V. Vinay, Eds. Springer Berlin Heidelberg, 1996, pp. 1–18.

[74] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation," in *Proceedings of the 24th International Conference on Machine Learning*, ser. ICML '07. New York, NY, USA: ACM, 2007, pp. 473–480.

[75] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in Neural Information Processing Systems 19*, P. B. Schölkopf, J. C. Platt, and T. Hoffman, Eds. MIT Press, 2007, pp. 153–160.

[76] C. Petrou and M. Paraskevas, "Signal processing techniques restructure the big data era," in *Proceedings of the 20th Pan-Hellenic Conference on Informatics*. ACM, 2016, p. 52.

[77] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[78] R. Bellman and R. Bellman, *Dynamic Programming*. Princeton University Press, 1957.

[79] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[80] T. S. Lee and D. Mumford, "Hierarchical bayesian inference in the visual cortex," *Journal of the Optical Society of America A*, vol. 20, no. 7, pp. 1434–1448, 2003.

[81] T. S. Lee, D. Mumford, R. Romero, and V. A. F. Lamme, "The role of the primary visual cortex in higher level vision," *Vision Research*, vol. 38, no. 15, pp. 2429–2454, 1998.

[82] N. Jones, "Computer science: The learning machines," *Nature*, vol. 505, pp. 146–148, 01 2014.

[83] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[84] G. E. Hinton, "What kind of a graphical model is the brain?" in *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, ser. IJCAI'05. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2005, pp. 1765–1775.

[85] M. aurelio Ranzato, C. Poultney, S. Chopra, and Y. L. Cun, "Efficient learning of sparse representations with an energy-based model," in *Advances in Neural Information Processing Systems 19*, P. B. Schölkopf, J. C. Platt, and T. Hoffman, Eds. MIT Press, 2007, pp. 1137–1144.

[86] Y. Bengio, "Learning deep architectures for ai," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[87] D. Erhan, P.-A. Manzagol, Y. Bengio, S. Bengio, and P. Vincent, "The difficulty of training deep architectures and the effect of unsupervised pre-training," pp. 153–160, 2009/04/15 2009.

[88] G. S. Babu, P. Zhao, and X.-L. Li, "Deep convolutional neural network based regression approach for estimation of remaining useful life," in *International conference on database systems for advanced applications*. Springer, 2016, pp. 214–228.

[89] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[90] O. Niculita, O. Nwora, and Z. Skaf, "Towards design of prognostics and health management solutions for maritime assets," *Procedia CIRP*, vol. 59, pp. 122–132, 2017.

[91] G. Manno, K. E. Knutsen, and B. J. Vartdal, "An importance measure approach to system level condition monitoring of ship machinery systems," in *11th International Conference on Condition Monitoring and Machinery Failure Prevention Technologies, CM 2014 and MFPT 2014, At Manchester*, 2014, Conference Proceedings.

[92] Y. Freund and D. Haussler, "Unsupervised learning of distributions on binary vectors using two layer networks," in *Advances in Neural Information Processing Systems 4*, J. E. Moody, S. J. Hanson, and R. P. Lippmann, Eds. Morgan-Kaufmann, 1992, pp. 912–919.

[93] G. E. Hinton, "A practical guide to training restricted boltzmann machines," in *Neural Networks: Tricks of the Trade: Second Edition*, G. Montavon, G. B. Orr, and K.-R. Mller, Eds. Springer Berlin Heidelberg, 2012, pp. 599–619.

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

[95] L. Deng, D. Yu *et al.*, "Deep learning: methods and applications," *Foundations and Trends® in Signal Processing*, vol. 7, no. 3–4, pp. 197–387, 2014.

[96] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[97] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, G. Gordon, D. Dunson, and M. Dudk, Eds., vol. 15. Fort Lauderdale, FL, USA: PMLR, 11–13 Apr 2011, pp. 315–323.

[98] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[99] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Comput.*, vol. 14, no. 8, pp. 1771–1800, 2002.

[100] Y. Fu, Y. Zhang, H. Qiao, D. Li, H. Zhou, and J. Leopold, "Analysis of feature extracting ability for cutting state monitoring using deep belief networks," *Procedia CIRP*, vol. 31, no. Supplement C, pp. 29–34, 2015.

[101] A. Saxena and K. Goebel, "Phm08 challenge data set," *NASA Ames Prognostics Data Repository (https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/), NASA Ames Research Center, Moffett Field, CA*, 2008.

[102] 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*, Oct 2008, pp. 1–9.

[103] V. T. Tran, F. AlThobiani, and A. Ball, "An approach to fault diagnosis of reciprocating compressor valves using teagerkaiser energy operator and deep belief networks," *Expert Systems with Applications*, vol. 41, no. 9, pp. 4113–4122, 2014.

[104] C. Li, R.-V. Sánchez, G. Zurita, M. Cerrada, and D. Cabrera, "Fault diagnosis for rotating machinery using vibration measurement deep statistical feature learning," *Sensors*, vol. 16, no. 6, p. 895, 2016.

[105] 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. PP, no. 99, pp. 1–13, 2017.

[106] A. Saxena and K. Goebel, "Turbofan engine degradation simulation data set," *NASA Ames Prognostics Data Repository (https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/), NASA Ames Research Center, Moffett Field, CA*, 2008.

[107] J. Deutsch and D. He, "Using deep learning-based approach to predict remaining useful life of rotating components," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. PP, no. 99, pp. 1–10, 2017.

[108] L. Liao, W. Jin, and R. Pavel, "Enhanced restricted boltzmann machine with prognosability regularization for prognostics and health assessment," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 11, pp. 7076–7083, 2016.

[109] P. Jiang, C. Chen, and X. Liu, "Time series prediction for evolutions of complex systems: A deep learning approach," in *2016 IEEE International Conference on Control and Robotics Engineering (ICCRE)*. IEEE, 2016, pp. 1–6.

[110] X. Qiu, L. Zhang, Y. Ren, P. N. Suganthan, and G. Amaratunga, "Ensemble deep learning for regression and time series forecasting," in *2014 IEEE Symposium on Computational Intelligence in Ensemble Learning (CIEL)*. IEEE, 2014, pp. 1–6.

[111] R. Hrasko, A. G. C. Pacheco, and R. A. Krohling, "Time series prediction using restricted boltzmann machines and backpropagation," *Procedia Computer Science*, vol. 55, no. Supplement C, pp. 990–999, 2015.

[112] T. Kuremoto, S. Kimura, K. Kobayashi, and M. Obayashi, "Time series forecasting using a deep belief network with restricted boltzmann machines," *Neurocomputing*, vol. 137, no. Supplement C, pp. 47–56, 2014.

[113] W. Sun, S. Shao, R. Zhao, R. Yan, X. Zhang, and X. Chen, "A sparse auto-encoder-based deep neural network approach for induction motor faults classification," *Measurement*, vol. 89, no. Supplement C, pp. 171–178, 2016.

[114] 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*. ACM, 2008, pp. 1096–1103.

[115] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, Dec. 2010.

[116] M. Xia, T. Li, L. Liu, L. Xu, and C. W. d. Silva, "Intelligent fault diagnosis approach with unsupervised feature learning by stacked denoising autoencoder," *IET Science, Measurement & Technology*, vol. 11, no. 6, pp. 687–695, 2017.

[117] M. Ranzato, C. Poultney, S. Chopra, and Y. Lecun, "Efficient learning of sparse representations with an energy-based model," in *Advances in Neural Information Processing Systems 19*, P. B. Schölkopf, J. C. Platt, and T. Hoffman, Eds. MIT Press, 2007, pp. 1137–1144.

[118] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 03 1951.

[119] W. Lu, X. Wang, C. Yang, and T. Zhang, "A novel feature extraction method using deep neural network for rolling bearing fault diagnosis," in *The 27th Chinese Control and Decision Conference (2015 CCDC)*, May 2015, pp. 2427–2431.

[120] Y. Ma, Z. Guo, J. Su, Y. Chen, X. Du, Y. Yang, C. Li, Y. Lin, and Y. Geng, "Deep learning for fault diagnosis based on multi-sourced heterogeneous data," in *2014 International Conference on Power System Technology*, Oct 2014, pp. 740–745.

[121] F. Jia, Y. Lei, J. Lin, X. Zhou, and N. Lu, "Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data," *Mechanical Systems and Signal Processing*, vol. 72-73, no. Supplement C, pp. 303–315, 2016.

[122] R. Thirukovalluru, S. Dixit, R. K. Sevakula, N. K. Verma, and A. Salour, "Generating feature sets for fault diagnosis using denoising stacked auto-encoder," in *2016 IEEE International Conference on Prognostics and Health Management (ICPHM)*. IEEE, 2016, pp. 1–7.

[123] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.

[124] S. Hochreiter, "Untersuchungen zu dynamischen neuronalen netzen," *Diploma thesis, Technische Universität München*, vol. 91, 1991.

[125] 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, Mar 1994.

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

[127] K. Greff, R. K. Srivastava, J. Koutnk, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. PP, no. 99, pp. 1–11, 2017.

[128] F. A. Gers, J. A. Schmidhuber, and F. A. Cummins, "Learning to forget: Continual prediction with lstm," *Neural Comput.*, vol. 12, no. 10, pp. 2451–2471, Oct. 2000.

[129] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm networks," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 4, July 2005, pp. 2047–2052 vol. 4.

[130] C. Olah, "Understanding lstm networks. 2015," *URL http://colah. github. io/posts/2015-08-Understanding-LSTMs/img/LSTM3-chain. png*, 2015.

[131] L. Liao and H.-i. Ahn, "Combining deep learning and survival analysis for asset health management," *International Journal of Prognostics and Health Management*, vol. 16, 2016.

[132] Z. Chen, Y. Liu, and S. Liu, "Mechanical state prediction based on lstm neural netwok," in *Control Conference (CCC), 2017 36th Chinese*. IEEE, 2017, pp. 3876–3881.

[133] F. O. Heimes, "Recurrent neural networks for remaining useful life estimation," in *Prognostics and Health Management, 2008. PHM 2008. International Conference on*. IEEE, 2008, pp. 1–6.

[134] 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.

[135] 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.

[136] M. Yuan, Y. Wu, and L. Lin, "Fault diagnosis and remaining useful life estimation of aero engine using lstm neural network," in *IEEE International Conference on Aircraft Utility Systems (AUS)*. IEEE, 2016, pp. 135–140.

[137] A. Gensler, J. Henze, B. Sick, and N. Raabe, "Deep learning for solar power forecastingan approach using autoencoder and lstm neural networks," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2016, pp. 002 858–002 865.

[138] 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," in *2016 Workshop on Machine Learning for Prognostic and Health Management*, 2016, Conference Proceedings.

[139] 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, Nov 1998.

[140] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 3, pp. 328–339, Mar 1989.

[141] Y. T. Zhou and R. Chellappa, "Computation of optical flow using a neural network," in *IEEE 1988 International Conference on Neural Networks*, July 1988, pp. 71–78 vol.2.

[142] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[143] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 1701–1708.

[144] R. Liu, G. Meng, B. Yang, C. Sun, and X. Chen, "Dislocated time series convolutional neural architecture: An intelligent fault diagnosis approach for electric machine," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1310–1320, 2017.

[145] L. Jing, M. Zhao, P. Li, and X. Xu, "A convolutional neural network based feature learning and fault diagnosis method for the condition monitoring of gearbox," *Measurement*, vol. 111, no. Supplement C, pp. 1 – 10, 2017.

[146] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[147] D. Nikoli, "Why deep neural nets cannot ever match biological intelligence and what to do about it?" *International Journal of Automation and Computing*, 2017.

[148] S. Marsland, "Reinforcement learning," in *Machine learning: an algorithmic perspective*, R. Herbrich and T. Graepel, Eds. Chapman & Hall/CRC, 2015, ch. 11, pp. 231–247.

[149] K. B. Ludvigsen, L. K. Jamt, N. Husteli, and . Smogeli, "Digital twins for design, testing and verification throughout a vessel's life cycle," in *15th International Conference on Computer and IT Applications in the Maritime Industries (COMPIT), Lecce, Italy*, 2016, Conference Proceedings, pp. 448–457.

[150] R. Rosen, G. von Wichert, G. Lo, and K. D. Bettenhausen, "About the importance of autonomy and digital twins for the future of manufacturing," *IFAC-PapersOnLine*, vol. 48, no. 3, pp. 567–572, 2015.

André Listou Ellefsen received his Master degree in Subsea Technology from the Norwegian University of Science and Technology (NTNU), Trondheim, Norway, in 2016. He is currently pursuing the Ph.D. degree with NTNU, alesund, Norway, as part of the Mechatronics Laboratory, Department of Ocean Operations and Civil Engineering. His current research interests include artificial intelligence, deep learning, decision support, predictive maintenance, prognostics and health management, and digital twins.

Prof. Dr. Vilmar Æsøy graduated from NTNU in 1989, and continued his research on natural gas fueled marine engines at NTNU/MARINTEK to 1997. In 1996 he received his PhD degree for his research on natural gas ignition and combustion through experimental investigations and numerical simulations. During the research period 1989-1997 he was engaged in several large R&D projects developing gas fueled engines and fuel injection systems for the diesel engine manufacturers, Wärtsilä and Bergen Diesel (Roll-Royce). From 1998 to 2002, he worked as R&D manager for Rolls-Royce Marine Deck Machinery. Since 2002 he has been employed in teaching at Aalesund University College, developing and teaching courses in marine product and systems design on bachelor and master level. From January 2010 he received the green ship machinery professorship. His special research interest is within the field of energy and environmental technology, with focus on combustion engines and the need for more environmental friendly and energy efficient systems.

Prof. Dr. Sergey Ushakov received his PhD degree in 2012 from the Department of Marine Technology at NTNU for the work on measurement and characterization of particulate matter emissions from marine diesel engines. Before re-joining the Department in 2016 as professor in marine machinery, for several years worked in MARINTEK (currently SINTEF Ocean) within the fields of marine diesel engine emission characterization and emission reduction technologies covering both volatile and non-volatile exhaust emissions. During this work he was involved in a number of bigger and smaller research projects where accumulated substantial experience with experimental work both in laboratory and on board of different vessels. The current research focus is environmentally friendly shipping as well as improvement of marine diesel engines efficiency especially emphasizing the experimental part of this work.

Prof. Dr. Houxiang Zhang (M04-SM12) received Ph.D. degree on Mechanical and Electronic Engineering in 2003. From 2004, he worked as Postdoctoral fellow, senior researcher at the Institute of Technical Aspects of Multimodal Systems (TAMS), Department of Informatics, Faculty of Mathematics, Informatics and Natural Sciences, University of Hamburg, Germany. In Feb. 2011, he finished the Habilitation on Informatics at University of Hamburg. Dr. Zhang joined the NTNU (before 2016, Aalesund University College), Norway in April 2011 where he is a Professor on Mechatronics. Dr. Zhang has engaged into two main research areas: 1) Biological robots and modular robotics, especially on biological locomotion control, 2) Virtual prototyping in demanding marine operation. He has applied for and coordinated more than 20 projects supported by Norwegian Research Council (NFR), German Research Council (DFG), and industry. In these areas, he has published over 160 journal and conference papers as author or co-author. Dr. Zhang has received four best paper awards, and four finalist awards for best conference paper at International conference on Robotics and Automation.

B

Paper II

Reliability Engineering and System Safety 183 (2019) 240–251

Contents lists available at ScienceDirect

Reliability Engineering and System Safety

journal homepage: www.elsevier.com/locate/ress

Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture

André Listou Ellefsen[*,a], Emil Bjørlykhaug[a], Vilmar Æsøy[a], Sergey Ushakov[b], Houxiang Zhang[a]

[a] Department of Ocean Operations and Civil Engineering, Norwegian University of Science and Technology, Aalesund 6009, Norway
[b] Department of Marine Technology, Norwegian University of Science and Technology, Trondheim 7491, Norway

ARTICLE INFO

Keywords:
C-MAPSS
Deep learning
Genetic algorithm
Prognostics and health management
Remaining useful life
Semi-supervised learning

ABSTRACT

In recent years, research has proposed several deep learning (DL) approaches to providing reliable remaining useful life (RUL) predictions in Prognostics and Health Management (PHM) applications. Although supervised DL techniques, such as Convolutional Neural Network and Long-Short Term Memory, have outperformed traditional prognosis algorithms, they are still dependent on large labeled training datasets. With respect to real-life PHM applications, high-quality labeled training data might be both challenging and time-consuming to acquire. Alternatively, unsupervised DL techniques introduce an initial pre-training stage to extract degradation related features from raw unlabeled training data automatically. Thus, the combination of unsupervised and supervised (semi-supervised) learning has the potential to provide high RUL prediction accuracy even with reduced amounts of labeled training data. This paper investigates the effect of unsupervised pre-training in RUL predictions utilizing a semi-supervised setup. Additionally, a Genetic Algorithm (GA) approach is applied in order to tune the diverse amount of hyper-parameters in the training procedure. The advantages of the proposed semi-supervised setup have been verified on the popular C-MAPSS dataset. The experimental study, compares this approach to purely supervised training, both when the training data is completely labeled and when the labeled training data is reduced, and to the most robust results in the literature. The results suggest that unsupervised pre-training is a promising feature in RUL predictions subjected to multiple operating conditions and fault modes.

1. Introduction

The remaining useful life (RUL) is a technical term used to describe the progression of faults in Prognostics and Health Management (PHM) applications [1]. Prognosis algorithms tend ideally to achieve the ideal maintenance policy through predictions of the available time before a failure occurs within a component or sub-component, that is RUL [2]. In this way, RUL predictions have the potential to prevent critical failures, and hence, becomes an important measurement to achieve the ultimate goal of zero-downtime performance in industrial systems. However, traditional prognosis algorithms suffer from a decreased capacity to process the increased complexity in today's sequential data with accuracy.

Recently, deep learning (DL) has emerged as a potent area to process highly non-linear and varying sequential data with minimal human input within the PHM domain [3]. Today, DL is an extremely active sub-field of machine learning. With increased processing power and continuous developments in graphics processors, DL has the potential to improve prediction tasks as the computational burden reduces significantly [4]. However, deep architectures introduce many diverse hyper-parameters, which are challenging to optimize in the training process. Thus, this study proposes a Genetic Algorithm (GA) approach in order to optimize the hyper-parameters in an efficient manner.

DL techniques, such as Convolutional Neural Network (CNN) and Long-Short Term Memory (LSTM), have shown rapid developments and outperformed traditional prognosis algorithms in RUL predictions for turbofan engine degradation [5–7]. DL techniques predict the RUL without any prior knowledge of engine degradation mechanics. Thus, data analysts today apply their knowledge about the RUL prediction problem to the selection and design of DL techniques, rather than to feature engineering. However, both CNN and LSTM depend on purely supervised learning. In other words, they require large labeled training datasets in the training procedure. Thus, the RUL prediction accuracy strongly depends on the quality of the constructed run-to-failure

* Corresponding author.
E-mail addresses: andre.ellefsen@ntnu.no (A. Listou Ellefsen), emil.bjorlykhaug@ntnu.no (E. Bjørlykhaug), vilmar.aesoy@ntnu.no (V. Æsøy), sergey.ushakov@ntnu.no (S. Ushakov), hozh@ntnu.no (H. Zhang).

https://doi.org/10.1016/j.ress.2018.11.027
Received 18 June 2018; Received in revised form 16 November 2018; Accepted 24 November 2018
Available online 26 November 2018

training data labels.

In contrast, unsupervised DL techniques introduce an initial pre-training stage to extract high-level abstract features from raw unlabeled training data automatically. Thus, the combination of unsupervised and supervised (semi-supervised) learning has the potential for even higher RUL prediction accuracy since the weights are initialized in a region near a good local minimum before supervised fine-tuning is conducted to minimize the global training objective [8].

More advanced and recent activation functions [9], learning rate methods [10], regularization techniques [11], and weight initializations [12,13] have indeed reduced the need for unsupervised pre-training in a variety of domains when the training data is completely labeled. Nevertheless, in real-life PHM applications, high-quality run-to-failure labeled training data is not easily obtained, especially from new equipment. However, unsupervised pre-training in semi-supervised setups has the potential to perform with high RUL prediction accuracy even with reduced amounts of labeled training data in the fine-tuning procedure. Additionally, most data collected in real-life PHM applications is subjected to several operating conditions and fault modes. This increases the inherent degradation complexity, which makes it more difficult for the prognosis algorithm to discover clear trends in the input data directly. To cope with this issue, the initial unsupervised pre-training stage can be utilized. Unsupervised pre-training extracts more degradation related features before supervised fine-tuning, and hence, has the potential to support the whole architecture to better understand the underlying degradation phenomena.

The aim of this paper is to show the effect of unsupervised pre-training in RUL predictions utilizing a semi-supervised setup. The results are verified on the four different simulated turbofan engine degradation datasets in the publicly available Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) dataset, produced and provided by NASA [14]. This study's main contributions are as follows:

- The GA approach effectively tunes hyper-parameters in deep architectures.
- Semi-supervised learning improves the RUL prediction accuracy compared to supervised learning in multivariate time series data with several operating conditions and fault modes when the training data is completely labeled.
- Semi-supervised learning performs higher RUL prediction accuracy compared to supervised learning when the labeled training data in the fine-tuning procedure is reduced.

The overall organization of the paper is as follows. Section 2 introduces recent and related work on the C-MAPSS dataset. Section 3 introduces the necessary background on GAs and the proposed semi-supervised setup. The experimental approach, results, and discussions are considered in Section 4. Finally, Section 5 concludes and closes the paper and provides directions for future work.

2. Related work

The C-MAPSS dataset has been extensively used to evaluate several DL approaches to RUL predictions. This section reviews the most recent studies applied on the C-MAPSS dataset. The selected studies either utilize a Convolutional Neural Network (CNN), a Deep Belief Network (DBN) or Long-Short Term Memory (LSTM) in the proposed deep architecture.

In most PHM applications, sequential data is a standard format of the input data, for example pressure and temperature time series data. LSTM is a well-established DL technique to process sequential data. The original LSTM [15] was developed after the early 1990s, when researchers discovered a vanishing and exploding gradient issue in traditional Recurrent Neural Networks (RNNs) [16]. This issue confirmed that traditional RNNs had difficulty learning long-term dependencies. To cope with this issue, the LSTM introduces a memory cell that regulates the information flow in and out of the cell. Consequently, the memory cell is able to preserve its state over long durations, that is learning long-term dependencies that may influence future predictions. Yuan et al. proposed an LSTM approach for several different faults [17]. The proposed approach was compared with traditional RNN, Gated Recurrent Unit LSTM (GRU-LSTM) and AdaBoost-LSTM. It showed improved performance in all cases. Another LSTM approach was provided by Zheng et al. [6]. The proposed approach provides RUL predictions using two LSTM layers, two Feed-forward Neural Network (FNN) layers, and an output layer. The LSTM layers were able to reveal hidden patterns in the C-MAPSS dataset and achieved higher accuracy compared to the Hidden Markov Model or traditional RNN. A similar study was provided by Wu et al. [18]. In this study, an LSTM was combined with a dynamic difference method in order to extract new features from several operating conditions before the training procedure. These features contain important degradation information, which improves the LSTM to better control the underlying physical process. The proposed approach showed enhanced performance compared to traditional RNN and GRU-LSTM.

Although CNNs have performed excellently on 2D and 3D grid-structured topology data, such as object recognition [20] and face recognition [21], respectively, CNNs can also be applied to 1D grid-structured topology sequential data in PHM applications. Babu et al. proposed a novel CNN approach for RUL predictions [5]. This CNN approach includes two layers with convolution and average-pooling steps, and a final FNN layer to perform RUL predictions. The proposed approach indicated improved accuracy compared to the Multilayer Perceptron (MLP), Support Vector Machine (SVM), and Relevance Vector Machine. More recently, [7] takes a CNN approach. In this study, Li et al. achieved even higher accuracy on the C-MAPSS dataset compared to both the LSTM approach in [6] and the CNN approach in [5]. They employed the recently developed, proven regularization technique "dropout" [11] and the adaptive learning rate method "adam" [10].

Hinton et al. introduced the greedy layer-wise unsupervised learning algorithm in 2006, designing it for DBNs [22]. A DBN consists of stacked Restricted Boltzmann Machines (RBMs) where the hidden layer in the previous RBM will serve as the input layer for the current RBM. The algorithm performs an initial unsupervised pre-training stage to learn internal representations from the input data automatically. Next, supervised fine-tuning is performed to minimize the training objective. Zhang et al. have proposed a multiobjective DBN ensemble approach [19]. This approach combines a multiobjective evolutionary ensemble learning framework with the DBN training process. Accordingly, the proposed approach creates multiple DBNs of varying accuracy and diversity before the evolved DBNs are combined to perform RUL predictions. The combined DBNs are optimized through differential evolution where the average training error is the single objective. The proposed approach outperformed several traditional machine learning algorithms, such as SVM and MLP. The recent studies are summarized in Table 1.

These studies all utilize a completely labeled run-to-failure training dataset in the training procedure. However, in real-life PHM scenarios, most data accumulated is unstructured and unlabeled from the start.

Table 1
Recent DL approaches proposed for RUL predictions on the C-MAPSS dataset [14] (the years between 2016 and 2018).

Author & Refs.	Year	Approach
Li et al. [7]	2018	CNN + FNN
Wu et al. [18]	2018	LSTM
Zheng et al. [6]	2017	LSTM + FNN
Yuan et al. [17]	2016	LSTM
Zhang et al. [19]	2016	MODBNE
Babu et al. [5]	2016	CNN + FNN

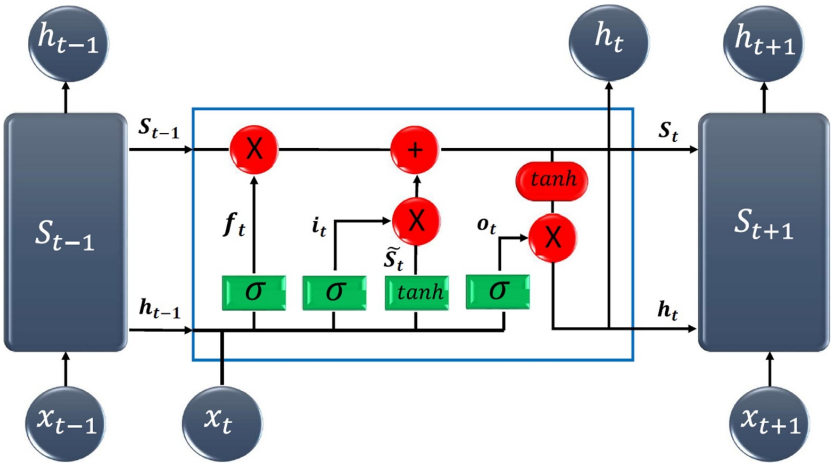


Fig. 1. Vanilla LSTM, adopted from Olah [28]. The blue rectangle represents the memory cell. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Valuable domain knowledge is required to construct run-to-failure data labels. This is both a time-consuming and challenging process. Thus, this study will investigate the effect of unsupervised pre-training in a semi-supervised setup both with reduced and completely labeled training datasets.

3. Proposed semi-supervised setup

This section will introduce the necessary background on the proposed semi-supervised setup. First, the main DL techniques included, RBM and LSTM, are defined. Next, the proposed deep architecture structure as well as the GA approach for hyper-parameter tuning are elaborated.

3.1. Restricted Boltzmann machine

RBMs were originally developed using binary stochastic visible units, v, in the input layer and binary stochastic hidden units, h, in the hidden layer [23]. However, in real-value data applications, like the C-MAPSS dataset, linear Gaussian units replace the binary visible units and rectified linear units replace the binary hidden units [24]. RBMs are symmetrical bipartite graphs since the visible and hidden units are fully connected and units in the same layer have zero connections.

RBMs are energy-based models with the joint probability distribution specified by their energy function [25]:

$$P(v, h) = \frac{1}{Z} e^{-E(v,h)} \tag{1}$$

where Z is the partition function that ensures that the distribution is normalized:

$$Z = \sum_{v} \sum_{h} e^{-E(v,h)} \tag{2}$$

The energy function for RBMs with Gaussian visible units is given by:

$$E(v, h) = \sum_{i=1}^{V} \frac{(v_i - b_i)^2}{2\gamma_i^2} - \sum_{j=1}^{H} c_j h_j - \sum_{i=1}^{V} \sum_{j=1}^{H} \frac{v_i}{\gamma_i} h_j w_{ij} \tag{3}$$

where w_{ij} denotes the weight between the visible unit v_i and hidden unit h_j, b_i and c_j represents the bias terms, V and H expresses the numbers of visible and hidden units, respectively, and γ_i is the standard deviation of v_i. As recommended by Hinton [25], zero mean and unit variance normalization should be applied to the input data. Contrastive divergence is used to train RBMs:

$$\Delta w_{ij} = \epsilon(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{recon}) \tag{4}$$

where ϵ is the learning rate. First, the data distribution samples visible units based on hidden units. Then, the input data is reconstructed, generated by Gibbs sampling, which samples hidden units based on visible units. This process continues until the parameters converge, that is, the hidden layer approximates the input layer. In this way, RBMs are able to model data distributions without any label knowledge. Typically, after the pre-training stage, the reconstruction part of the RBM is omitted and the pre-trained weights facilitate a subsequent supervised fine-tuning procedure.

3.2. Long-Short term memory

Modifications by Gers et al. [26] have been included in the original LSTM, and researchers generally refer to this LSTM setup as the "vanilla LSTM." Although several variants of the vanilla LSTM have been proposed, Greff et al. have shown that none of the variants can improve the vanilla LSTM significantly [27]. Thus, the proposed semi-supervised setup uses the vanilla LSTM.

The memory cell, as illustrated in Fig. 1, consists of three non-linear gating units that protect and regulate the cell state, S_t [28]:

$$f_t = \sigma(W_f x_t + R_f h_{t-1} + b_f) \tag{5}$$

$$i_t = \sigma(W_i x_t + R_i h_{t-1} + b_i) \tag{6}$$

$$o_t = \sigma(W_o x_t + R_o h_{t-1} + b_o) \tag{7}$$

where σ is the sigmoid gate activation function in order to obtain a scaled value between 0 and 1, W is the input weight, R is the recurrent weight, and b is the bias weight.

The new candidate state values, $\tilde{S}_t$, are created by the tanh layer:

$$\tilde{S}_t = tanh(W_s x_t + R_s h_{t-1} + b_s) \tag{8}$$

The previous cell state, S_{t-1}, is updated into the new cell state, S_t, by:

$$S_t = f_t \otimes S_{t-1} + i_t \otimes \tilde{S}_t \tag{9}$$

where $\otimes$ denotes element-wise multiplication of two vectors. First, the forget layer, f_t, determines which historical information the memory cell removes from S_t. Then, the input layer, i_t, decides what new

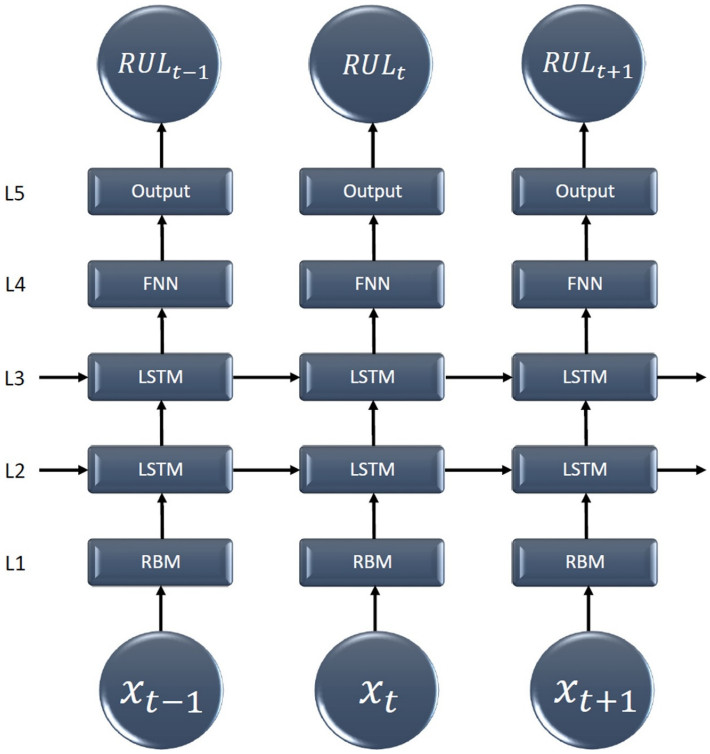


Fig. 2. The proposed semi-supervised deep architecture structure.

information in $\tilde{S}_t$ the memory cell will update and store in S_t.

The output layer, o_t, determines which parts of S_t the memory cell will output. S_t is filtered in order to push the values between -1 and 1:

$$h_t = o_t \otimes \ tanh\,(S_t) \tag{10}$$

Through these steps, the vanilla LSTM has the ability to remove or add information to S_t.

3.3. The proposed deep architecture structure and the genetic algorithm approach

The proposed semi-supervised deep architecture structure is shown in Fig. 2. In the first layer (L1), a RBM will be utilized as an unsupervised pre-training stage in order to learn abstract features from raw unlabeled input data automatically. These features might contain important degradation information, and hence, initialize the weights in a region near a good local minimum before supervised fine-tuning of the whole architecture is conducted. In both the second and the third layer (L2 and L3), an LSTM layer is used to reveal hidden information and learn long-term dependencies in sequential data with multiple operating and fault conditions [6]. Next, an FNN layer is used in the fourth layer (L4) in order to map all extracted features. In the final layer (L5), a time distributed fully connected output layer is attached to handle error calculations and perform RUL predictions.

The GA is a metaheuristic inspired by the natural selection found in nature [29]. It is a powerful tool for finding a near-optimal solution in a big search space. In this work, a GA approach is proposed to tune hyper-parameters. First, the GA approach selects random hyper-parameters for the proposed semi-supervised deep architecture within a given search space. One such set of random hyper-parameters is called an individual and a set of individuals is called a population. Next, the accuracy of each of the individuals in the population are evaluated by training networks with the individuals hyper-parameters. The best results from the training are then kept and used as parents for the next generation of hyper-parameters. Additionally, some random mutation is performed after the crossover for increasing the exploration of the algorithm.

4. Experimental study

In the following experimental study, the proposed semi-supervised deep architecture will be compared to recent studies in the literature as well as purely supervised training. The latter comparison will be performed with and without the initial pre-training stage utilizing the proposed semi-supervised deep architecture when the labeled training data in the fine-tuning procedure is reduced. Experiments are performed on the four subsets provided in the benchmark C-MAPSS dataset: FD001, FD002, FD003, and FD004. All experiments are run on NIVIDIA GeForce GTX 1060 6 GB and the Microsoft Windows 10 operating system. The programming language is Java 8 with deep learning library “deeplearning4j” (DL4J) version 0.9.1 [30].

4.1. The benchmark C-MAPSS dataset and performance evaluation

The C-MAPSS dataset is divided into four subsets, as shown in Table 2, and each subset is further divided into training and test sets of multiple multivariate time series. Each time series is from a different aircraft gas turbine engine and starts with different degrees of initial wear and manufacturing variation, which is unknown to the data analyzer. All engines operate in normal condition at the start, then begin to degrade at some point during the time series. The degradation in the training sets grows in magnitude until failure, while the

Table 2
The C-MAPSS dataset [14].

Dataset	FD001	FD002	FD003	FD004
Time series training set	100	260	100	249
Time series test set	100	259	100	248
Operating conditions	1	6	1	6
Fault conditions	1	1	2	2

degradation in the test sets ends sometime prior to failure, that is the RUL. That is, the last time step for each engine in the test sets provides the true RUL targets. Thus, the main objective is to predict the correct RUL value for each engine in the test sets. The four subsets vary in operating and fault conditions and the data is contaminated with sensor noise. Each subset includes 26 columns: engine number, time step, three operational sensor settings, and 21 sensor measurements. See [14,31] for a deeper understanding of the C-MAPSS dataset.

The scoring function (S) provided in Saxena et al. [31] and the root mean square error (*RMSE*) are used in this study to evaluate the performance of the proposed semi-supervised setup:

$$S = \begin{cases} \sum_{i=1}^{n} e^{(-\frac{d_i}{13})} - 1, & \text{for} \quad d_i < 0 \\ \sum_{i=1}^{n} e^{(-\frac{d_i}{10})} - 1, & \text{for} \quad d_i \geq 0 \end{cases} \tag{11}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} d_i^2} \tag{12}$$

where n is the total number of true RUL targets in the respective test set and $d_i = RUL_{predicted} - RUL_{true}$. As shown in Fig. 3, the *RMSE* gives equal penalty to early and late predictions. In the asymmetric scoring function, however, the penalty for late predictions is larger. Late predictions could cause serious system failures in real-life PHM applications as the maintenance procedure will be scheduled too late. On the other hand, early predictions pose less risk since the maintenance procedure will be scheduled too early, and hence, there is still time to perform maintenance. Nevertheless, the main objective is to achieve the smallest value possible for both S and *RMSE*, that is, when $d_i = 0$.

Only evaluating performance at the last time step for each engine in the test sets has both advantages and disadvantages. High and reliable RUL prediction accuracy at the very end of components and sub-components lifetime have of course great industrial significance, as this period is critical for PHM applications. However, this evaluation approach could hide the true overall prognostics accuracy as the prognostics horizon of the algorithm is not considered. The prognostics horizon is critical in order to achieve trustworthy confidence intervals for the corresponding RUL prediction. These confidence intervals are important due to both inherent uncertainties with the degradation process and potential flaws in the prognosis algorithm [32].

Table 3
Genes in the GA approach.

Gene	Hyper-parameter	Values
1	R_c	115, 120, 125, 130, 135, 140
2	Learning rate RBM layer	10^{-1}, 10^{-2}, 10^{-3}
3	Learning rate remaining layers	10^{-2}, 10^{-3}, 10^{-4}
4	L2 Regularization	10^{-4}, 10^{-5}, 10^{-6}
5	miniBatch	5, 10
6	n L1	32, 64, 128
7	n L2	32, 64, 128
8	n L3	32, 64, 128
9	n L4	8, 16
10	p L2	0.5, 0.6, 0.7, 0.8, 0.9
11	p L3	0.5, 0.6, 0.7, 0.8, 0.9
12	p L4	0.5, 0.6, 0.7, 0.8, 0.9
13	I/O activation function LSTM	sigmoid, tanh
14	Activation function FNN	sigmoid, tanh

Table 4
Parameters of the GA approach.

Parameter	Value
Population size	20
Nr of elite	3
Mutation rate	0.5
Mutation gain	0.3
Evolution iterations	3

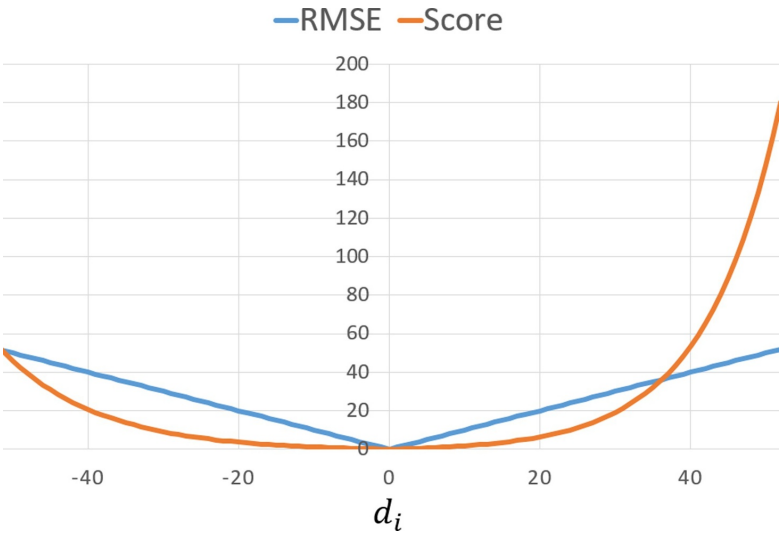


Fig. 3. Simple illustration of the scoring function vs. RMSE, where $d_i = RUL_{predicted} - RUL_{true}$.

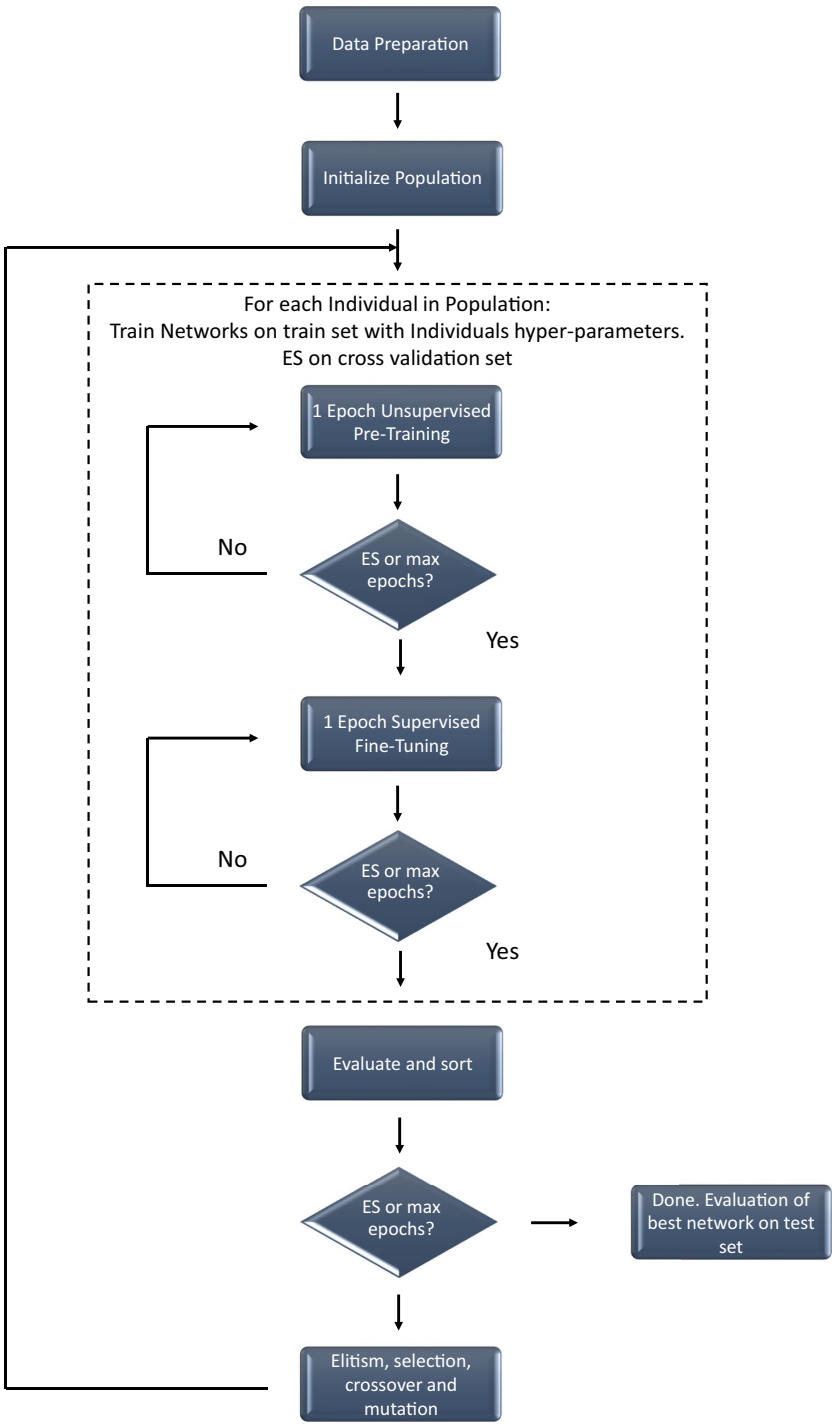


Fig. 4. Flowchart of the GA approach.

Table 5
GA individuals.

FD001	Layer index	DL technique	nIn	nOut	Dropout	Activation function
	1	RBM	14	64	1.0	ReLU
	2	LSTM	64	64	0.9	Sigmoid
	3	LSTM	64	64	0.6	Sigmoid
	4	FNN	64	8	0.6	Sigmoid
	5	Output	8	1	1.0	Identity
	R_c	Learning rate RBM layer	Learning rate remaining layers	L2 regularization	mini batch size	*RMSE* cross-validation set
	115	10^{-2}	10^{-3}	10^{-6}	5	8.49
FD002	Layer index	DL technique	nIn	nOut	Dropout	Activation function
	1	RBM	24	64	1.0	ReLU
	2	LSTM	64	128	0.7	Sigmoid
	3	LSTM	128	32	0.8	Sigmoid
	4	FNN	32	8	0.6	Sigmoid
	5	Output	8	1	1.0	Identity
	R_c	Learning rate RBM layer	Learning rate remaining layers	L2 regularization	mini batch size	*RMSE* cross-validation set
	135	10^{-2}	10^{-3}	10^{-5}	10	9.60
FD003	Layer index	DL technique	nIn	nOut	Dropout	Activation function
	1	RBM	14	32	1.0	ReLU
	2	LSTM	32	128	0.9	Sigmoid
	3	LSTM	128	64	0.9	Sigmoid
	4	FNN	64	8	0.9	Sigmoid
	5	Output	8	1	1.0	Identity
	R_c	Learning rate RBM layer	Learning rate remaining layers	L2 regularization	mini batch size	*RMSE* cross-validation set
	125	10^{-2}	10^{-3}	10^{-6}	5	8.59
FD004	Layer index	DL technique	nIn	nOut	Dropout	Activation function
	1	RBM	24	64	1.0	ReLU
	2	LSTM	64	128	0.8	Sigmoid
	3	LSTM	128	32	0.7	Sigmoid
	4	FNN	32	8	0.6	Sigmoid
	5	Output	8	1	1.0	Identity
	R_c	Learning rate RBM layer	Learning rate remaining layers	L2 regularization	mini batch size	*RMSE* cross-validation set
	135	10^{-2}	10^{-3}	10^{-5}	10	10.45

Table 6
The proposed semi-supervised deep architecture with and without unsupervised pre-training on subset FD004 when the labeled training data is reduced from 100% to 10%. Improvement $= (1 - \frac{\text{Semi}-\text{supervised}}{\text{Supervised}})$.

RMSE	100%	80%	60%	40%	20%	10%
Semi-supervised with 100% training features in the pre-training stage	22.66	23.04	24.07	25.46	30.26	34.19
Supervised	23.62	23.45	24.14	26.40	30.27	34.90
Improvement	4.06%	1.75%	0.29%	3.56%	0.03%	2.03%
S	100%	80%	60%	40%	20%	10%
Semi-supervised with 100% training features in the pre-training stage	2840	3175	3576	5522	9562	22,476
Supervised	3234	3427	3650	6536	15,612	27,138
Improvement	12.18%	7.35%	2.03%	15.51%	38.75%	17.18%
Average training time per epoch (s)	100%	80%	60%	40%	20%	10%
Pre-training stage	7.08	7.08	7.08	7.08	7.08	7.08
Fine-tuning procedure	34.14	28.97	22.39	15.2	9.74	5.93

4.2. Data preparation

4.2.1. Masking and padding

The DL4J library provides a "CSVSequenceRecordReader" to handle time series data. It reads time series data, where each time series is defined in its own file. Each line in the files represents one time step. Consequently, each time series (engine) in the four training sets are split into their own file. The input training data has the following shape: [miniBatchSize, inputSize, timeSeriesLength], where miniBatchSize is the number of engines in the mini batch, input size is the number of columns, and timeSeriesLength is the total number of time steps in the mini batch. The engines have variable time step lengths, and hence, the shorter engines in a mini batch are padded with zeros such that the time step lengths are equal to the longest among them. Accordingly, mask arrays are used during training. These additional arrays record whether a time step is actually present, or whether it is just padding. In all performance evaluations, mask arrays are considered.

4.2.2. Feature selection

Sensor 1, 5, 6, 10, 16, 18, and 19 in subset FD001 and FD003 exhibit constant sensor measurements throughout the engineâ;;s lifetime. Constant sensor measurements does not provide any useful degradation

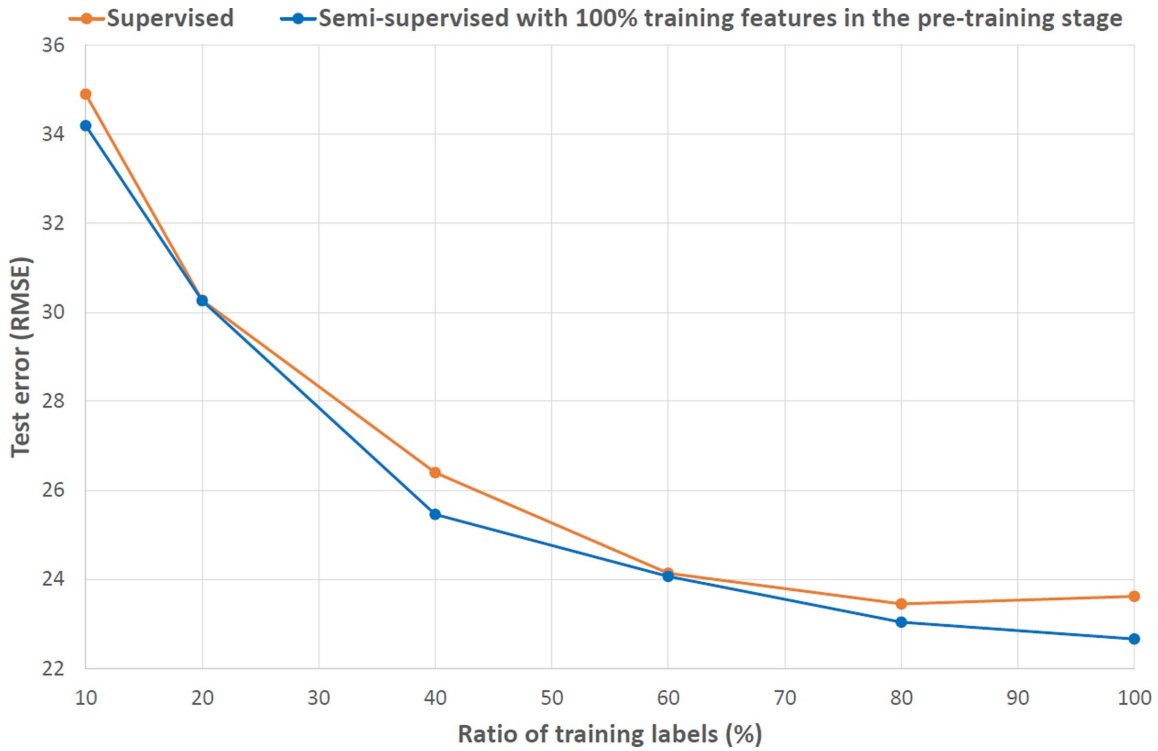


Fig. 5. *RMSE* comparison on subset FD004 when the labeled training data is reduced from 100% to 10%.

information regarding RUL predictions [19,33]. In addition, subset FD001 and FD003 are subjected to a single operating condition [5]. Hence, the three operational settings are excluded. Accordingly, sensor 2, 3, 4, 7, 8, 9, 11, 12, 13, 14, 15, 17, 20, and 21 are used as the input features for subset FD001 and FD003.

Subset FD002 and FD004 are more complex due to six operating conditions [18]. Six operating conditions make it challenging for the prognosis algorithm to detect clear degradation patterns in the input data directly. However, two LSTM layers were able to find hidden patterns in Zheng et al. [6]. Additionally, the initial unsupervised pre-training stage is able to capture hierarchically statistical patterns before the supervised fine-tuning procedure. Consequently, these patterns will enable the whole architecture to cope with the complexity inherent in degradation. Thus, all three operational sensor settings and all sensor measurements are used as the input features for subset FD002 and FD004.

4.2.3. RUL targets

True RUL targets are only provided at the last time step for each engine in the test sets. In order to construct labels for every time step for each engine in the training sets, Heimes et al. [33] used an MLP function estimator to show that it is reasonable to estimate RUL as a constant value when the engines operate in normal condition. Based on their experiments, a degradation model was proposed with a constant RUL value (R_c) of 130 and a minimum value of 0. This piece-wise linear RUL target function is still the most common approach in the literature [5–7,18,19]. However, R_c varies among the different studies. For this study, the GA approach is used to test different R_c since it has a notable impact on the experimental performance for the different subsets in the C-MAPSS dataset.

4.2.4. Data normalization

All input features and labels are normalized with zero mean unit variance (z-score) normalization:

$$z = \frac{x - \mu}{\sigma} \tag{13}$$

where μ is the mean and σ is the corresponding standard deviation.

4.3. Deep architecture configuration and training

In the initial RBM layer, a rectified linear unit (ReLU) is used as the activation function as ReLUs improve the performance of RBMs compared to the tanh activation function [9]. Stochastic gradient descent is the selected optimization algorithm and adaptive moment estimation (Adam) [10] is the learning rate method applied to the deep architecture. Recently, Adam has shown great results on the C-MAPSS dataset [7,18]. To better preserve the information in the pre-trained weights, the learning rate in the initial RBM layer is one order of magnitude higher than the learning rate in the remaining layers. ReLU weight initialization [13] is applied to the RBM layer while Xavier weight initialization [12] is applied to the remaining layers in the proposed semi-supervised deep architecture.

Truncated backpropagation through time (TBPTT) is used in this study due to a large amount of time steps in the training sets. TBPTT performs more frequent parameter updates compared to standard backpropagation through time. This both reduces computational complexity and improves learning of temporal dependencies [34]. The forward and backward passes are set to 100 time steps, as the shortest time series in the C-MAPSS dataset contains 128 time steps.

In the training procedure, each complete training subset is split into a training set and a cross-validation set. In subset FD001 and FD003,

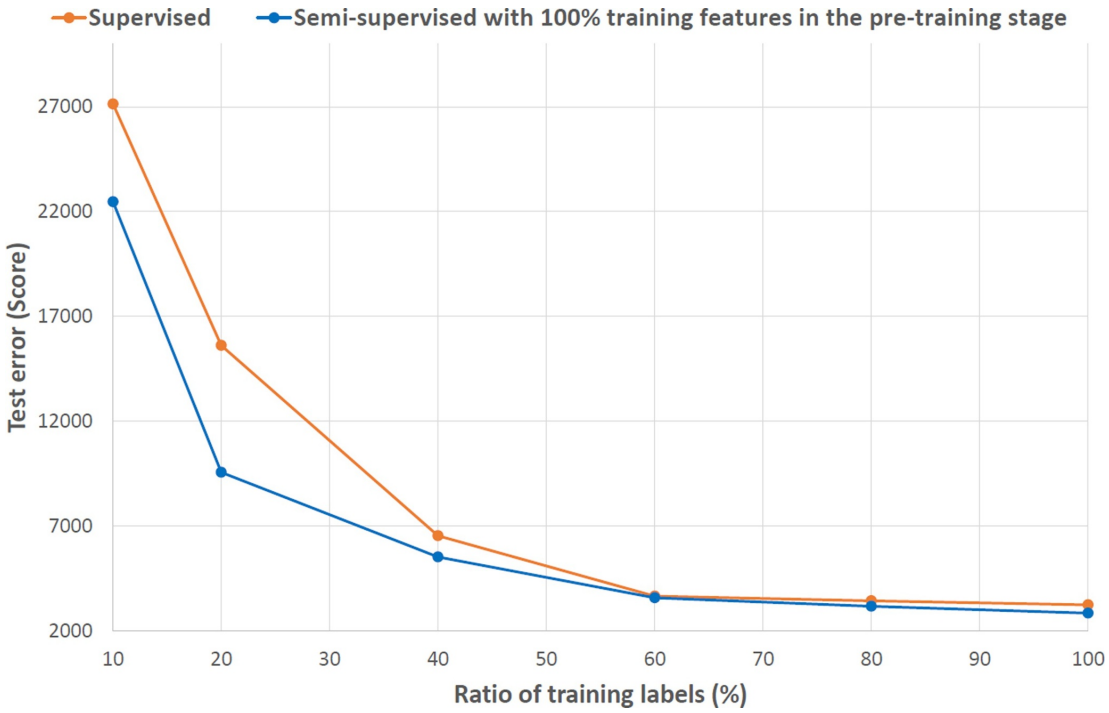


Fig. 6. *S* comparison on subset FD004 when the labeled training data is reduced from 100% to 10%.

20% of the total engines in the complete training subsets are randomly selected for cross-validation. The remaining 80% are designated as the training sets. Due to an increased complete training subset size in subset FD002 and FD004, 10% of the total engines are randomly selected for cross-validation while the remaining 90% are designated as the training sets.

Table 3shows all the hyper-parameters which the GA approach needs to optimize for each subset. The recent and well-proven regularization technique dropout [11] is applied to the deep architecture. Dropout introduces the hyper-parameter, p, which randomly drops units during training. In this way, dropout approximately combines an exponential number of different architectures. Thus, the deep architecture learns to make generalized representations of the input data, which enhances the feature extraction ability. In Table 3, n and p refer to the number of hidden units and the probability of retaining each hidden unit in the coupled hidden layer L, respectively. A p value of 1.0 is functionally equivalent to zero dropout, i.e. 100% probability of retaining each hidden unit. A typical value for p used in the literature is 0.5 [7,18]. However, p depends on n. In this study, the GA approach is able to test different values of n in both L1, L2, L3, and L4, and hence, it is also able to test different values of p in the range from 0.5 to 0.9. As Patterson and Gibson [35] recommend, to preserve important features in the input data, dropout is disabled in the first layer, L1. Additionally, dropout is not used in the output layer, L5. It should be noted that dropout is only applied to the non-recurrent connections in the LSTM layers.

The GA approach is run once for each subset. It trains a diverse number of individuals on the training sets and evaluates the *RMSE*, Eq. 12, on the cross-validation set as its objective function. In this way, the GA approach optimizes the hyper-parameters for each subset. To limit the time consumed during the optimization process, the population size is restricted to 20 individuals and the population is evolved three times with the selected GA parameters as shown in Table 4. This results in an average training time of 60 hours for each subset. However, the training time will reduce significantly along with future developments in GPUs. Additionally, to prevent overfitting, early stopping (ES) is applied to monitor the performance during the training process of each individual. In the unsupervised pre-training stage, ES is used to monitor the reconstruction error on the training set. If the number of epochs with no improvement exceeds nine, the unsupervised pre-training procedure is terminated. In the fine-tuning procedure, ES is used to monitor the RMSE accuracy on the cross-validation set. If the number of epochs with no improvement exceeds four, the fine-tuning procedure is terminated. Finally, the top five GA individuals for each subset are evaluated on the test sets where both *RMSE* and *S* are calculated. A complete flowchart of the GA approach is shown in Fig. 4 and the best GA individuals for each subset are shown in Table 5. In Table 5, nIn and nOut represents the number of input and output (hidden) units for each layer, respectively.

4.4. Experimental results and discussions

The aim of this paper is to show increased RUL prediction accuracy in multivariate time series data subjected to several operating conditions and fault modes utilizing a semi-supervised setup. The experiments conducted in this study shows the effect of unsupervised pre-training both when the training data is completely labeled and when the labeled training data in the fine-tuning procedure is reduced.

4.4.1. The effect of unsupervised pre-training in RUL predictions

Subset FD004 is chosen for this experiment due to the complexity inherent in its six operating conditions and two fault modes. As shown in Table 6, semi-supervised learning provides higher RUL prediction accuracy compared to supervised learning when the training data is 100% labeled. This indicates that the unsupervised pre-training stage initializes the weights using a more suitable local minimum than

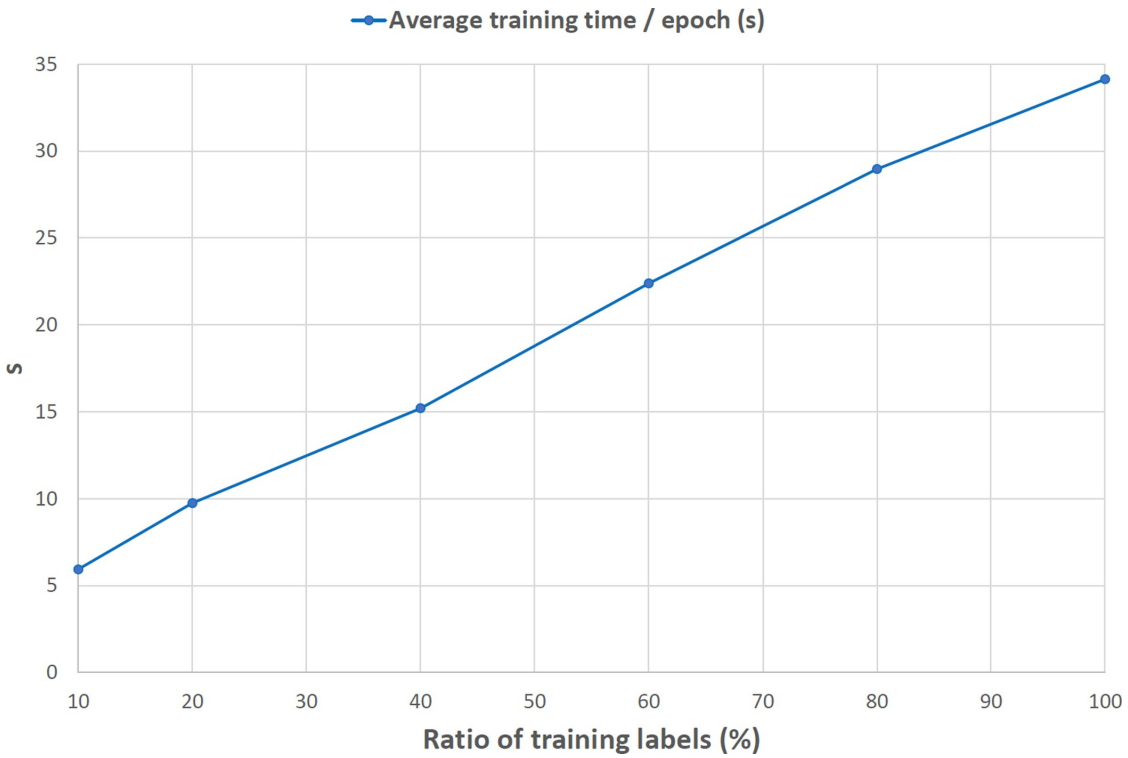


Fig. 7. Average training time in seconds per epoch in the fine-tuning procedure when the labeled training data is reduced from 100% to 10%.

weights that are randomly initialized. Consequently, unsupervised pre-training supports better comprehension of the inherent degradation complexity in the whole architecture.

In real-life PHM scenarios, high-quality labeled training data is hard to acquire. To address this problem, this study has performed an experiment where only reduced parts of the training data in subset FD004 contains labels. The labels in the training set are randomly reduced into fractions of 20%, 40%, 60%, 80%, and 90%, respectively. To minimize any selection bias, the random selection process is repeated five times for each fraction. Each random selection is then trained on the training set and evaluated on the test set where *RMSE* and *S* are calculated. Finally, the top three performance results are averaged as shown in Table 6. It should be noted that a similar experiment, which has made interesting and valuable results using a variational autoencoder (VAE), is conducted on subset FD001 in [36].

To show the effect of unsupervised pre-training, the proposed deep architecture is trained with and without the initial pre-training stage. In the initial pre-training stage, the proposed deep architecture is trained with 100% training features. The ES procedure is used to monitor the performance. As shown in Figs. 5 and 6, the proposed semi-supervised deep architecture provides the overall highest RUL prediction accuracy when trained with the initial unsupervised pre-training stage. It should be noted that the proposed deep architecture, when trained in a purely supervised manner, also provides satisfactory RUL prediction accuracy, especially when more than 60% of the training labels are included. This proves that recent weight initializations and regularization techniques, such as Xavier and dropout, have indeed reduced the need for unsupervised pre-training. Dropout in particular improves the feature extraction ability by approximately combining several different architectures in the fine-tuning procedure. However, the improvement of utilizing semi-supervised learning is noticeable when more than 40% of the training labels are removed, as shown in Table 6.

Additionally, as shown in Fig. 7, the average training time per epoch will almost linearly decrease with decreasing training labels, e.g. 15.2 s training time at 40% labels, which is 15.2 s/34.14 s= 44.5% training time per epoch compared to 100% labels. Also, as seen in Figs. 5 and 6, the RUL prediction accuracy is satisfactory when more than 60% training labels are included. Depending on the reliability and safety requirements of the application, the trade-off of reduced RUL prediction accuracy might be acceptable if the training time is critical.

4.4.2. Comparison with the literature

Studies that have reported results on all four subsets in the C-MAPSS dataset have been selected for comparison. Although the initial *Rc* values are somewhat different, the results are still comparable. As shown in Tables 7 and 8, the proposed semi-supervised deep architecture has achieved promising results compared to the recent studies when the training data is completely labeled. The CNN approach in Li et al. [7] achieved slightly higher *RMSE* prediction accuracy on subset FD002. However, the proposed semi-supervised deep architecture indicates substantially improved *S* prediction accuracy on all subsets. Consequently, the proposed semi-supervised deep architecture reduces the

Table 7
RMSE comparison with the literature on the C-MAPSS dataset.

DL approach & refs.	FD001	FD002	FD003	FD004
CNN + FNN [5]	18.45	30.29	19.82	29.16
LSTM + FNN [6]	16.14	24.49	16.18	28.17
MODBNE [19]	15.04	25.05	12.51	28.66
CNN + FNN [7]	12.61	**22.36**	12.64	23.31
Proposed semi-supervised setup	**12.56**	22.73	**12.10**	**22.66**

Table 8
Score function comparison with the literature on the C-MAPSS dataset.

DL approach & Refs.	FD001	FD002	FD003	FD004
CNN + FNN [5]	1287	13,570	1596	7886
LSTM + FNN [6]	338	4450	852	5550
MODBNE [19]	334	5585	422	6558
CNN + FNN [7]	274	10,412	284	12,466
Proposed semi-supervised setup	**231**	**3366**	**251**	**2840**

average number of late predictions across the test sets considerably. This is because the unsupervised pre-training stage extracts more degradation related features before supervised fine-tuning. Thus, this stage supports the whole architecture to better understand the underlying degradation trends. Late predictions impose a serious threat to reliability and safety in real-life PHM applications as the maintenance procedure will be scheduled too late. Therefore, semi-supervised learning is a promising approach in RUL predictions tasks both subjected to a single and multiple operating conditions and fault modes.

5. Conclusion and future work

This paper has investigated the effect of unsupervised pre-training in RUL predictions utilizing a semi-supervised setup. The experiments are performed on the publicly available C-MAPSS dataset. Additionally, a GA approach was proposed to tune the number of diverse hyper-parameters in deep architectures. Combining all the hyper-parameters in Table 3 results in a total of 8 748 000 combinations. Although, the GA approach only used 20 individuals and three evolutions, it was able to optimize hyper-parameters for each subset in the C-MAPSS dataset effectively. This is a promising approach compared to using a time consuming, exhaustive search. However, the average training time of 60 hours for each subset will be further optimized in future work.

In the experimental study, the proposed semi-supervised setup is compared to purely supervised training as well as recent studies in the literature. The proposed semi-supervised setup achieved promising RUL prediction accuracy with both completely and reduced amounts of labeled training data. Hence, unsupervised pre-training is indeed a promising feature in real-life PHM applications subjected to multiple operating conditions and fault modes, as large amounts of high-quality labeled training data might be both challenging and time-consuming to acquire. Unsupervised pre-training supports the deep architecture to improve our understanding of the inherent complexity by extracting more features that contain important degradation information.

In this study, an RBM was utilized as the initial unsupervised pre-training stage. However, RBM is a rather old, unsupervised DL technique. Today, more powerful unsupervised DL techniques are available. For instance, the VAE [36,37] seems promising. The VAE models the underlying probability distribution of the training data using variational inference. It is possible to extend to a wide range of model architectures, and this is one of its key advantages compared to RBM, which requires careful model design to maintain tractability [38].

In RUL predictions based on data-driven approaches, such as DL, the accuracy strongly depends on the quality of the constructed run-to-failure training data labels. This study confirms that R_c has a notable impact on the RUL prediction accuracy for each subset. Nevertheless, the piece-wise linear degradation model used in this study is considered a major limitation as each engine in each subset has, in fact, an individual degradation pattern. Recently, the VAE has been used for unsupervised reconstruction based anomaly detection by applying a reconstruction error as an anomaly score [39]. Thus, in future work, the VAE will also be used in order to create an unsupervised fault detector to optimize R_c for each engine in each subset in the C-MAPSS dataset.

Normally, tanh is used as the input and output (I/O) activation function in LSTMs. However, in this study it was discovered that sigmoid performed better than tanh as the LSTM I/O activation function in combination with the initial RBM layer with ReLU as the activation function. A novel rectified LSTM I/O activation function would be a positive contribution to be included in future work.

Acknowledgment

This work was supported by the Norwegian University of Science and Technology within the Department of Ocean Operations and Civil Engineering under project no. 90329106 and funded by the Research Council of Norway, grant no. 245613/O30. The authors would like to thank Digital Twins For Vessel Life Cycle Service (DigiTwin) NFR 280703.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.ress.2018.11.027

References

[1] Kalgren PW, Byington CS, Roemer MJ, Watson MJ. Defining phm, a lexical evolution of maintenance and logistics. 2006 IEEE Autotestcon. 2006. p. 353–8. https://doi.org/10.1109/AUTEST.2006.283685.
[2] Peng Y, Wang Y, Zi Y. Switching state-space degradation model with recursive filter/smoother for prognostics of remaining useful life. IEEE Trans Ind Inf 2018. https://doi.org/10.1109/TII.2018.2810284. 1–1
[3] Zhao G, Zhang G, Ge Q, Liu X. Research advances in fault diagnosis and prognostic based on deep learning. Prognostics and System Health Management Conference (PHM-Chengdu). IEEE; 2016. p. 1–6. https://doi.org/10.1109/PHM.2016.7819786.
[4] Chen XW, Lin X. Big data deep learning: challenges and perspectives. IEEE Access 2014;2:514–25. https://doi.org/10.1109/ACCESS.2014.2325029.
[5] Sateesh Babu G, Zhao P, Li X-L. Deep convolutional neural network based regression approach for estimation of remaining useful life. Cham: Springer International Publishing; 2016. p. 214–28. https://doi.org/10.1007/978-3-319-32025-0_14. ISBN 978-3-319-32025-0
[6] Zheng S, Ristovski K, Farahat A, Gupta C. Long short-term memory network for remaining useful life estimation. 2017 IEEE International Conference on Prognostics and Health Management (ICPHM). IEEE; 2017. p. 88–95.
[7] Li X, Ding Q, Sun J-Q. Remaining useful life estimation in prognostics using deep convolution neural networks. Reliab Eng Syst Saf 2018;172:1–11.
[8] Erhan D, Manzagol P-A, Bengio Y, Bengio S, Vincent P. The difficulty of training deep architectures and the effect of unsupervised pre-training. Artificial Intelligence and Statistics. 2009. p. 153–60.
[9] Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks. In: Gordon G, Dunson D, DudÃk M, editors. Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics; vol. 15 of Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA: PMLR; 2011. p. 315–23.
[10] Kingma D.P., Ba J.. Adam: a method for stochastic optimization. arXiv:14126980 2014;.
[11] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 2014;15(1):1929–58.
[12] Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. Proceedings of the 13th International Conference on Artificial Intelligence and Statistics. 2010. p. 249–56.
[13] He K, Zhang X, Ren S, Sun J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. Proceedings of the IEEE international conference on computer vision. 2015. p. 1026–34.
[14] Saxena A., Goebel K.. Turbofan engine degradation simulation data set. NASA Ames Prognostics Data Repository (https://tiarcnasagov/tech/dash/groups/pcoe/prognostic-data-repository/), NASA Ames Research Center, Moffett Field, CA2008;.
[15] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput 1997;9(8):1735–80. https://doi.org/10.1162/neco.1997.9.8.1735.
[16] Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult. IEEE Trans Neural Netw 1994;5(2):157–66. https://doi.org/10.1109/72.279181.
[17] Yuan M, Wu Y, Lin L. Fault diagnosis and remaining useful life estimation of aero engine using lstm neural network. IEEE International Conference on Aircraft Utility Systems (AUS). IEEE; 2016. p. 135–40.
[18] Wu Y, Yuan M, Dong S, Lin L, Liu Y. Remaining useful life estimation of engineered systems using vanilla lstm neural networks. Neurocomputing 2018;275:167–79.
[19] Zhang C, Lim P, Qin AK, Tan KC. Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics. IEEE Trans Neural Netw Learn Syst 2017;28(10):2306–18. https://doi.org/10.1109/TNNLS.2016.2582798.
[20] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: Pereira F, Burges CJC, Bottou L, Weinberger KQ, editors. Advances in Neural Information Processing Systems 25. Curran Associates, Inc.; 2012. p. 1097–105.
[21] Taigman Y, Yang M, Ranzato M, Wolf L. Deepface: closing the gap to human-level performance in face verification. 2014 IEEE Conference on Computer Vision and

Pattern Recognition. 2014. p. 1701–8. https://doi.org/10.1109/CVPR.2014.220.
[22] Hinton GE, Osindero S, Teh Y-W. A fast learning algorithm for deep belief nets. Neural Comput 2006;18(7):1527–54. https://doi.org/10.1162/neco.2006.18.7.1527.
[23] Freund Y, Haussler D. Unsupervised learning of distributions on binary vectors using two layer networks. In: Moody JE, Hanson SJ, Lippmann RP, editors. Advances in Neural Information Processing Systems 4. Morgan-Kaufmann; 1992. p. 912–9.
[24] Nair V, Hinton GE. Rectified linear units improve restricted Boltzmann machines. Proceedings of the 27th international conference on machine learning (ICML-10). 2010. p. 807–14.
[25] Hinton GE. A practical guide to training restricted Boltzmann machines. Neural Networks: Tricks of the Trade. Springer; 2012. p. 599–619.
[26] Gers FA, Schmidhuber JA, Cummins FA. Learning to forget: continual prediction with lstm. Neural Comput 2000;12(10):2451–71. https://doi.org/10.1162/089976600300015015.
[27] Greff K, Srivastava RK, Koutník J, Steunebrink BR, Schmidhuber J. Lstm: a search space odyssey. IEEE Trans Neural Netw Learn Syst 2017;28(10):2222–32. https://doi.org/10.1109/TNNLS.2016.2582924.
[28] Olah C. Understanding lstm networks. 2015. 2015. URL http://colahgithubio/posts/2015-08-Understanding-LSTMs/img/LSTM3-chainpng
[29] Roberge V, Tarbouchi M, Labonté G. Comparison of parallel genetic algorithm and particle swarm optimization for real-time uav path planning. IEEE Trans Ind Inf 2013;9(1):132–41.
[30] Eclipse deeplearning4j development team, deeplearning4j: open-source distributed deep learning for the jvm. Apache Software Foundation License 20 http://deeplearning4jorg 2018;.
[31] Saxena A, Goebel K, Simon D, Eklund N. Damage propagation modeling for aircraft engine run-to-failure simulation. 2008 International Conference on Prognostics and Health Management. IEEE; 2008. p. 1–9.
[32] Sikorska JZ, Hodkiewicz M, Ma L. Prognostic modelling options for remaining useful life estimation by industry. Mech Syst Signal Process 2011;25(5):1803–36. https://doi.org/10.1016/j.ymssp.2010.11.018.
[33] Heimes FO. Recurrent neural networks for remaining useful life estimation. International Conference on Prognostics and Health Management, PHM 2008.. IEEE; 2008. p. 1–6.
[34] Sutskever I. Training recurrent neural networks. Toronto, Ont, Canada: University of Toronto; 2013.
[35] Patterson J, Gibson A. Deep learning: a practitioner's approach. "O'Reilly Media, Inc."; 2017.
[36] Yoon AS, Lee T, Lim Y, Jung D, Kang P, Kim D, et al. Semi-supervised learning with deep generative models for asset failure prediction. CoRR 2017. abs/1709.00845
[37] Kingma D.P., Welling M.. Auto-encoding variational bayes. arXiv:131261142013;.
[38] Goodfellow I, Bengio Y, Courville A. Deep learning. MIT Press; 2016. http://www.deeplearningbook.org
[39] Park D, Hoshi Y, Kemp CC. A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder. IEEE Rob Autom Lett 2018;3(3):1544–51.

C

Paper III

SPECIAL SECTION ON ADVANCES IN PROGNOSTICS AND SYSTEM HEALTH MANAGEMENT

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 10, 2019, accepted January 23, 2019, date of publication January 25, 2019, date of current version February 12, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2895394

An Unsupervised Reconstruction-Based Fault Detection Algorithm for Maritime Components

ANDRÉ LISTOU ELLEFSEN, EMIL BJØRLYKHAUG, VILMAR ÆSØY, AND HOUXIANG ZHANG, (Senior Member, IEEE)

Department of Ocean Operations and Civil Engineering, Norwegian University of Science and Technology, 6009 Ålesund, Norway

Corresponding author: André Listou Ellefsen (andre.ellefsen@ntnu.no)

This work was supported in part by the Department of Ocean Operations and Civil Engineering, Norwegian University of Science and Technology, under Grant 90329106, and in part by the Research Council of Norway, under Grant 280703.

ABSTRACT In recent years, the reliability and safety requirements of ship systems have increased drastically. This has prompted a paradigm shift toward the development of prognostics and health management (PHM) approaches for these systems' critical maritime components. In light of harsh environmental conditions with varying operational loads, and a lack of fault labels in the maritime industry generally, any PHM solution for maritime components should include independent and intelligent fault detection algorithms that can report faults automatically. In this paper, we propose an unsupervised reconstruction-based fault detection algorithm for maritime components. The advantages of the proposed algorithm are verified on five different data sets of real operational run-to-failure data provided by a highly regarded industrial company. Each data set is subject to a fault at an unknown time step. In addition, different magnitudes of random white Gaussian noise are applied to each data set in order to create several real-life situations. The results suggest that the algorithm is highly suitable to be included as part of a pure data-driven diagnostics approach in future end-to-end PHM system solutions.

INDEX TERMS Automatic fault detection, deep learning, maritime industry, prognostics and health management, unsupervised learning.

I. INTRODUCTION

Ship systems are more complex and integrated than ever before. Thus, the degradation of critical maritime components included in these systems poses a serious threat to safe and profitable maritime operations [1]. In general, maintenance in shipping either follows a reactive maintenance (RM) or preventive maintenance (PvM) approach [2]. RM can be described as post-failure repair, and hence, it will create large and unnecessary costs when critical maritime component failures occur during operation. PvM involves predetermined maintenance intervals based on constant intervals or age-based or imperfect maintenance [3]. PvM will, of course, provide high reliability, but it involves unneeded maintenance inspections and procedures involving completely functional systems. Additionally, critical maritime components are, in fact, subject to random failure patterns due to different environmental conditions with varying operational loads [4]. Neither RM nor PvM is sufficient to identify these kinds of failures. The need for prognostics and health management (PHM) approaches which incorporate automatic fault detection and associated remaining useful life (RUL) predictions is urgent. RUL predictions aim to obtain the ideal maintenance policy through predictions of the available time until failure after a fault is detected within the component [5]. In this way, PHM approaches have the potential to prevent critical maritime component failures, and hence, considerably enhance maritime operational performance and safety [6].

Recently, deep learning (DL) has emerged as a potent data-driven area for accurate RUL predictions for component degradation [5], [7]. RUL-based DL techniques utilize raw input sensor data and are less dependent on prior domain knowledge of component mechanics. However, they depend on large, labeled run-to-failure data in the training process. Thus, the RUL predictions strongly depend on the accuracy of the fault detection algorithm, that is, the process of separating normal operating data from faulty degradation data in order to create run-to-failure labels.

The associate editor coordinating the review of this manuscript and approving it for publication was Dong Wang.

In general, traditional fault detection algorithms based on signal processing methods, such as Empirical Mode Decomposition [8] and Wavelet Transform [9], are to some extent application specific and need prior domain knowledge to distinguish normal operating data from faulty degradation data. Due to varying operational conditions, fault detection algorithms for critical maritime components should not be application specific. Additionally, with respect to the maritime industry generally, there is a lack of fault labels of critical maritime components [10]. This creates major issues towards successful implementation of fault detection algorithms that utilize a supervised classifier to separate normal operating data from faulty degradation data [11]. Thus, maritime components require independent and intelligent fault detection algorithms in order to detect and report faults automatically.

This paper investigates the possibilities for automatic fault detection within maritime components. In order to do so, an unsupervised reconstruction-based fault detection algorithm for maritime components is introduced. The algorithm can be applied to several machine learning (ML) algorithms and encoder-decoder (ED)-structured DL techniques. Thus, it will be tested on four techniques: traditional Feed-forward Neural Network with one hidden layer (1FNN), Autoencoder (AE), Variational Autoencoder (VAE), and Long-Short Term Memory (LSTM). Each technique is trained and evaluated on five different data sets of real operational run-to-failure data of the same maritime component collected from a highly regarded industrial company. Each data set is subject to a fault at an unknown time step. Additionally, different magnitudes of random white Gaussian noise are applied to each data set to create several real-life situations in order to test the robustness of the algorithm. First, the algorithm estimates an anomaly score function by calculating a reconstruction error at each time step in faulty degradation data. Then, the algorithm detects a fault automatically by estimating the time step with the highest acceleration in the anomaly score function. This study's main contributions are as follows:

- ED-structured DL techniques prove robustness towards noisy real operational input data.
- The proposed algorithm is not application specific, that is, the algorithm proves consistent high accuracy in real operational input data when subjected to varying operational conditions. Additionally, the algorithm is considered more generic than fault indications based on user-specified threshold values.
- The proposed algorithm reports faults automatically with no prior knowledge of component degradation mechanics.

The overall organization of the paper is as follows. Section II introduces recent and related work on intelligent fault detection algorithms. Section III introduces the necessary background on traditional FNN and ED-structured DL techniques. The experimental approach, results, and discussions are considered in section IV. Finally, Section V concludes and closes the paper and provides directions for future work.

II. RELATED WORK

The development of intelligent fault detection algorithms has exploded in the last two years. The majority is based on reconstruction-based fault detection by applying a reconstruction error as an anomaly score. The core idea is to train a specific machine learning (ML) algorithm, in an unsupervised manner, to reconstruct normal operating data. The ML algorithm will then provide a higher reconstruction error on unforeseen trends in faulty degradation data. Brandsæter *et al.* [12]used Auto Associative Kernel Regression (AAKR) for reconstruction and the Sequential Probability Ratio Test for anomaly detection provided. In order to determine the fault condition, a lower bound and upper bound threshold value was used. Yang *et al.* [13] used Support Vector Regression (SVR) for reconstruction and probability information based on three statistical indexes for anomaly detection. However, both AAKR and SVR are considered shallow ML algorithms which might not reconstruct high-dimensional and noisy operational data accurately. ED-structured DL techniques are well-suited to first compress and then reconstruct such operational data. The compressed version of the input supports the reconstruction process to extract information relevant to the normal operating data. In this way, ED-structured DL techniques cannot reconstruct unforeseen patterns in faulty degradation data, which results in a larger reconstruction error.

Recent studies have employed variations on the traditional AE for fault detection of rolling bearings, verified on the data set provided by Case Western Reserve University Bearing Data Center [14]. Lu *et al.* [15] demonstrated the effectiveness of a Stacked Denoising Autoencoder (SDA). The SDA showed improved accuracy for signals containing ambient noise and different working loads compared to traditional fault detection algorithms. Nevertheless, the accuracy indicated inconsistency between different working loads. Liu *et al.* [16] used a Gated Recurrent Unit-based nonlinear predictive Denoising Autoencoder (GRU-NP-DAE) provided. The proposed method showed improved accuracy compared to several state-of-the-art methods, including the SDA provided in [15]. Both the SDA and the GRU-NP-DAE trained a supervised classifier to separate normal operating data from faulty degradation data. Thus, both approaches require fault labels in the training process. Additionally, the approaches were trained under a de-noising criterion [17], that is, the input was corrupted stochastically while the target for reconstruction was kept as the original input. To make full use of both acoustic and vibratory signals, Li *et al.* [18] used a deep random forest fusion (DRFF) technique. The proposed approach combined deep feature representations and data fusion strategies to show improved performance of gearbox fault diagnostics. Nevertheless, the DRFF technique also trained a supervised classifier.

Although the above approaches have shown superior fault detection accuracy compared to traditional fault detection algorithms, they are less suitable for maritime components. First, maritime components are subjected to varying

environmental and operating conditions. Thus, a suitable fault detection algorithm should not rely on user-specified threshold values. Second, supervised classifiers require fault labels in the training process. This is a barrier given that there is a common lack of fault labels in the maritime industry. Finally, maritime components are subjected to random amounts of noise in real operational input data. Thus the de-noising criterion is not completely realistic, as in real-life situations the target for reconstruction will also contain the noise. Hence, maritime components require more independent and intelligent fault detection algorithms.

In the last two years, independent and intelligent fault detection algorithms have begun to develop. Park *et al.* [19] introduced an LSTM based Variational Autoencoder (LSTM-VAE) anomaly detector for robot-assisted feeding. The LSTM-VAE reports an anomaly when a reconstruction-based anomaly score is higher than a varying state-based threshold. The threshold changes over the estimated state of a task execution. Malhotra *et al.* [20] used an LSTM approach to reconstruct time-series data. The reconstruction error was used to compute a health index (HI) curve. Then, the HI curve was used to create run-to-failure labels in order to predict the RUL. The unsupervised reconstruction-based fault detection algorithm for maritime components which we propose in this work follows the idea of generic fault detection provided in [19]. However, the main difference is the utilization of the time step with the highest acceleration as varying fault indications. Additionally, the proposed algorithm can be further used to create run-to-failure labels in order to predict the RUL, similar to the approach in [20].

III. BACKGROUND ON ED STRUCTURED DL TECHNIQUES

This section will introduce the necessary background on the traditional FNN and the ED-structured DL techniques used in this study. First, FNN, AE, VAE, and LSTM are defined. Next, the configuration and performance evaluation of the unsupervised reconstruction models are elaborated.

A. FEED-FORWARD NEURAL NETWORK

Traditional FNNs form the basis of the ED-structured DL techniques used in this paper. FNNs aim to approximate some function f^* by mapping an input $\boldsymbol{x}$ to a target y, that is, $y = f^*(\boldsymbol{x})$. An FNN defines a mapping $\boldsymbol{y} = f(\boldsymbol{x}; \boldsymbol{\theta})$ and learns the value of the parameters $\boldsymbol{\theta}$, which consists of weights and biases, through the back-propagation algorithm [21]. FNNs are typically called networks because they are represented by combining together several layers [22]. Each unit in layer l computes its own activation value:

$$a_j^l = \sigma(z_j^l) \tag{1}$$

where σ is the activation function and the argument is the weighted sum

$$z_j^l = b_j^l + \sum_k w_{jk}^l a_k^{l-1} \tag{2}$$

of the output a_k^{l-1} from unit k in the previous layer $l-1$. b_j^l denotes the bias, while w_{jk}^l represent the weight factors. In the first hidden layer $l = 1$, the input is $a_j^0 = x_j$, where x_j, $j = 1 \ldots n$, are the inputs to the FNN. As each layer is fully connected, the weighted sum of the outputs of layer $l-1$ is over all units k.

B. AUTOENCODER

An AE is an FNN trained to reconstruct its input through a "bottleneck" representation of latent variables (hidden units) z [23]. As seen in Figure 1, the AE consists of an encoder function $z = f_{\theta_e}(x)$ and a decoder function that produces a reconstruction $r = g_{\theta_d}(z)$. The AE objective function is as follows [23]:

$$J_{AE}(\theta_e, \theta_d) = \sum L(x, r) \tag{3}$$

The optimization of the parameters θ_e and θ_d, which consist of weights and biases, are learned concurrently in the reconstruction process and compared to the original input data in order to obtain the lowest possible reconstruction error $L(x, r)$. In this work, $L(x, r)$ is the mean squared error (MSE), and hence, the AE objective function becomes:

$$J_{AE}(\theta_e, \theta_d) = \frac{1}{m} \sum_{i=1}^{m} ||x_i - g_{\theta_d}(f_{\theta_e}(x_i))||^2 \tag{4}$$

where m is the number of units in the input layer. AEs can be stacked with several hidden layers, depending on the dimensionality of the input data, and it is trained by the back-propagation algorithm. Significantly, unsupervised pre-training might be necessary for AEs with many hidden layers.

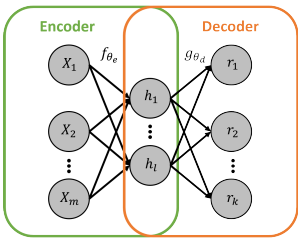


FIGURE 1. A simple illustration of an AE. *m* units in the input layer, *l* units in the hidden layer (bottleneck), and *k* units in the output layer.

C. VARIATIONAL AUTOENCODER

The VAE is a modern variation of the traditional AE, developed by Kingma and Welling [24]. Compared to the traditional AE, the VAE models the underlying probability distribution using Bayesian inference. Thus, the latent variables z are stochastic variables, and this improves generalization. As seen in Figure 2, the VAE consists of an encoder function $z = q_{\theta_e}(z|x)$ and a decoder function $r = p_{\theta_d}(x|z)$. The objective function of the VAE is to maximize the variational lower bound J_{VAE} associated with data point x [22]:

$$J_{VAE}(\theta_e, \theta_d) = -D_{KL}\big(q_{\theta_e}(z|x) \,||\, p_{\theta_d}(z)\big) + E_{q_{\theta_e}(z|x)}[\log p_{\theta_d}(x|z)] \tag{5}$$

where D_{KL} is the Kullback-Leibeler (KL) divergence. The first term provides a regularization since it measures how closely the latent variables match the encoder function (latent loss), while the second term is the reconstruction log-likelihood (generative loss). However, the reconstruction error term in Eq. 5 requires a Monte Carlo estimate of the expectation, and this is not easily differentiable [24]. A reparameterization trick of z is applied to obtain the gradients of the decoder in order to use the back-propagation algorithm. The reparameterization trick introduces a deterministic variable such that $z = \boldsymbol{\mu} + \boldsymbol{\sigma}\varepsilon$, $\varepsilon \sim \mathcal{N}(0, 1)$ [24]. Thus, the encoder now generates a vector of means and a vector of standard deviations instead of a vector of real values. As seen in Figure 2, these vectors are then used as the latent vector in the decoder. For real-valued input data, a Gaussian reconstruction distribution is used in the decoding process. Like AEs, the VAE can be stacked with several hidden layers depending on the dimensionality of the input data. Also, pretraining might be necessary with many hidden layers.

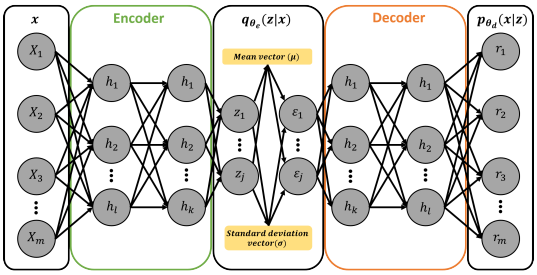


FIGURE 2. A simple illustration of a VAE. *m* units in the input layer, *l* and *k* units in the hidden layers of the encoder and decoder, and *j* units in latent vector.

D. LONG-SHORT TERM MEMORY

Today, modifications by [25]–[27] have been included in the original LSTM [28], and the literature refers to this as the "vanilla LSTM". This study uses "vanilla LSTM" with no peephole connections. As opposed to traditional Recurrent Neural Networks, the LSTM introduces a memory cell that regulates the information flow in and out of the cell. Thus, the memory cell is able to preserve its state over time, such that it learns long-term dependencies. As seen in Figure 3, the memory cell consists of three non-linear gating units that protect and regulate the cell state, $\boldsymbol{S}_t$ [29]:

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_f \boldsymbol{x}_t + \boldsymbol{R}_f \boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \tag{6}$$

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_i \boldsymbol{x}_t + \boldsymbol{R}_i \boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \tag{7}$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_o \boldsymbol{x}_t + \boldsymbol{R}_o \boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \tag{8}$$

where σ is the logistic sigmoid gate activation function, $\sigma(x) = \frac{1}{1+e^{-x}}$, which provides a scaled value between 0 and 1. $\boldsymbol{W}$ is the input weight, $\boldsymbol{R}$ is the recurrent weight, and $\boldsymbol{b}$ is the bias weight. The new candidate state values, $\tilde{\boldsymbol{S}}_t$, are created

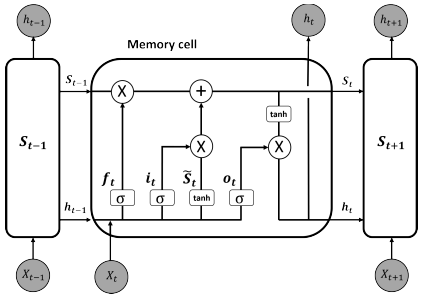


FIGURE 3. A simple illustration of an LSTM. f_t, i_t, and o_t represents the forget, input, and output gate, respectively.

by the tanh layer:

$$\tilde{\boldsymbol{S}}_t = tanh(\boldsymbol{W}_s \boldsymbol{x}_t + \boldsymbol{R}_s \boldsymbol{h}_{t-1} + \boldsymbol{b}_s) \tag{9}$$

The previous cell state, $\boldsymbol{S}_{t-1}$, is updated into the new cell state, $\boldsymbol{S}_t$, by

$$\boldsymbol{S}_t = \boldsymbol{f}_t \otimes \boldsymbol{S}_{t-1} + \boldsymbol{i}_t \otimes \tilde{\boldsymbol{S}}_t \tag{10}$$

where $\otimes$ denotes element-wise multiplication of two vectors. First, $\boldsymbol{f}_t$ determines which historical information the memory cell should forget. Then, $\boldsymbol{i}_t$ decides what new information in $\tilde{\boldsymbol{S}}_t$ the memory cell will input and store in $\boldsymbol{S}_t$. Finally, $\boldsymbol{o}_t$ determines which parts of $\boldsymbol{S}_t$ the memory cell will output:

$$\boldsymbol{h}_t = \boldsymbol{o}_t \otimes tanh(\boldsymbol{S}_t) \tag{11}$$

Through these equations, the LSTM has the ability to remove or add information to $\boldsymbol{S}_t$, which makes it highly suitable to process time-series data. Like AEs and VAEs, the LSTM is trained by the back-propagation algorithm and can be stacked with several hidden layers depending on the dimensionality of the input data.

E. UNSUPERVISED RECONSTRUCTION MODELS

In this study, 1FNN, AE, VAE, and LSTM are structured as an ED in order to create several diverse reconstruction models for comparison. The 1FNN is the simplest model and configured by one hidden layer with 14 units in both the encoder and decoder. In other words, the 1FNN is equal to an AE with one hidden layer. The AE, VAE, and LSTM are structured as deep models and configured by three hidden layers with 17, 8, and 4 units in the encoder and three hidden layers with 4, 8, and 17 units in the decoder, respectively. Let $\boldsymbol{x}_t = [x_1 \ldots x_n]_t$ denote the vector of input sensor measurements at time step t. Each reconstruction model is trained in an unsupervised manner, such that at each time step t the input $\boldsymbol{x}_t$ is also used as the target $\boldsymbol{y}_t$ for the reconstruction, $\boldsymbol{y}_t = \boldsymbol{x}_t$. A fully connected output layer is attached to each reconstruction model to handle error calculations. The selected loss function in the output layer is the MSE:

$$MSE = \frac{1}{n}\sum_{i=1}^{n} ||\hat{y}_i - y_i||^2 \tag{12}$$

where n is the number of sensors, and $\hat{y}_i$ and y_i are the i_{th} predicted and target sensor measurement, respectively.

IV. EXPERIMENTAL STUDY

In the following experimental study, each reconstruction model is trained and evaluated on five different data sets of real operational run-to-failure data of the same maritime component collected from an industrial company. First, each reconstruction model is trained on normal operating data. Next, an anomaly score function is estimated for each model by calculating the MSE, Eq. 12, at each time step in faulty degradation data. Finally, a generic and intelligent fault detection algorithm is employed to detect an unknown fault automatically. All experiments are run on NVIDIA GeForce GTX 1060 6 GB and the Microsoft Windows 10 operating system. The programming language is Java 8 and the deep learning library is "deeplearning4j" (DL4J) version 1.0.0-beta2 [30].

A. DATA SETS

The five data sets used in this study are provided by a highly regarded industrial company and collected from the same maritime component. A confidentiality agreement bars us from stating the actual name of the maritime component, fault types, and sensor measurements. The data sets start with different operational loads and corresponding sensor measurements. As seen in Table 1, each data set differs in total time step length T_{total}. Data sets 1 and 4 are subjected to fault type A, while data set 2, 3, and 5 are subjected to fault type B. In each data set, the maritime component operates in normal conditions at the start, then begin to degrade at an unknown point during the time series. The degradation grows in magnitude until failure. Thus, the main objective is automatically to detect the time step where the degradation starts, that is, the fault time step f_t. In order to train the reconstruction models, the initial 25% of each data set is considered normal operating data (training data) and the remaining 75% is considered faulty degradation data (test data). Thus, the total time step lengths in the training and test data are $T_{nod} = T_{total} \cdot 0.25$ and $T_{fdd} = T_{total} \cdot 0.75$, respectively. Each data set has 14 sensor measurements.

TABLE 1. Real operational run-to-failure data sets of a maritime component.

Data set	Fault type	T_{total}	T_{nod}	T_{fdd}	w
1	A	887	222	665	19
2	B	909	227	682	19.5
3	B	1859	465	1394	39.8
4	A	2554	638	1916	54.7
5	B	3643	911	2732	78.1

B. DATA NORMALIZATION AND PREPARATION

Each sensor measurement x_n in the input and target vector, $\boldsymbol{y}_t = \boldsymbol{x}_t = [x_1 \dots x_n]_t$, is normalized with zero mean and unit variance (z-score) normalization:

$$\hat{x}_n = \frac{x_n - \mu}{\sigma} \tag{13}$$

where μ and σ is the mean and the corresponding standard deviation of the population, respectively. Additionally, maritime components are subjected to random amounts of noise in real operational input data. Thus, to increase the complexity of each training data set and create differentiated real-life maritime situations, different magnitudes of random white Gaussian noise, g, is added to each $\hat{x}_n$ at each time step t. We assume that the real world noise is random white Gaussian noise. P_{signal} and P_{noise} are the average power of the signal and the noise in the training data, respectively, and defined as follows:

$$P_{signal} = \frac{1}{T_{nod}} \sum_{t=1}^{T_{nod}} \left(\sqrt{\frac{1}{n}\left(\hat{x}_1^2 + \dots + \hat{x}_n^2\right)} \right)_t \tag{14}$$

$$P_{noise} = \frac{1}{T_{nod}} \sum_{t=1}^{T_{nod}} \left(\sqrt{\frac{1}{n}\left((\hat{x}_1 + g)^2 + \dots + (\hat{x}_n + g)^2\right)} \right)_t \tag{15}$$

Then, the signal-to-noise-ratio (SNR) can be defined as:

$$SNR(\%) = \frac{P_{signal}}{P_{noise}} \cdot 100 \tag{16}$$

C. CONFIGURATION AND TRAINING

The reconstruction models are configured with joint hyper-parameters in order to make reliable comparisons. Stochastic gradient descent (SGD) is the selected optimization algorithm and adaptive moment estimation (Adam) is the learning rate method. The learning rate is $l_r = 10^{-3}$ and the $l2$ regularization value is 10^{-4}. Xavier weight initialization is applied to all layers. The rectified linear unit (ReLU) activation function is used in 1FNN, AE, and VAE. However, in the LSTM, the tanh activation function is used in order to push the input and output values between -1 and 1. The selected hyper-parameters are summarized in Table 2. During the training process of each reconstruction model, an early stopping (ES) approach is used in order to reconstruct the normal operating data as accurately as possible. The ES approach monitors the total reconstruction error of all time steps $E_{T_{nod}}$ for each epoch in the training data:

$$E_{T_{nod}} = \sum_{t=1}^{T_{nod}} \left(\frac{1}{n} \sum_{i=1}^{n} ||\hat{y}_i - y_i||^2 \right)_t \tag{17}$$

If the number of epochs with no reduction on $E_{T_{nod}}$ exceeds four, the training process is terminated. Then, the reconstruction model, in the epoch with the lowest $E_{T_{nod}}$, is saved and evaluated on the faulty degradation data.

TABLE 2. Joint hyper-parameters.

Hyper-parameter	Method/value
Optimization algorithm	SGD
l_r method	Adam
l_r	10^{-3}
$l2$ regularization	10^{-4}
Weight initialization	Xavier
Activation function	ReLU (tanh in LSTM)

D. PREDICTION OF FAULT TIME STEP IN FAULTY DEGRADATION DATA

The anomaly score function is estimated by calculating the MSE, Eq. 12, at each time step in the faulty degradation data. Then, the calculations and the corresponding time steps are saved in a score list S_l and a time step list T_l, respectively. Next, a generic and intelligent fault detection algorithm is employed in order to predict the fault time step $\hat{f}_t$. First, the algorithm creates three sliding windows of length $w = T_{fdd}/35$. Table 1 shows w for each data set. The value of 35 is used for all data sets in order to keep the same percentage level, that is $(1/35) \cdot 100 = 2.86\%$, on the faulty degradation data. In this work, the value of 35 is based on trial and error. However, w is a critical parameter and should be tuned carefully for other practical applications. The value of w will depend on the amount of noise in S_l. Second, the three windows slide across S_l for each time step in T_l. A distance equal to w is used between each sliding window. In order to remove noise in S_l, the average reconstruction score S_{avg} is calculated in the three windows. Third, the velocity v between windows 1 and 2 and between windows 2 and 3 are calculated. Finally, the acceleration a and the corresponding $\hat{f}_t$ are estimated. The sliding window operation is illustrated in Figure 4 and the proposed algorithm is elaborated in Algorithm 1. Large sensor measurements deviations compared to typical sensor measurements in normal operating data is a valid indication of a fault. The aim of the proposed algorithm is to detect the time step with the highest acceleration a_{max} in faulty degradation data. a_{max} is used as the fault criterion since this point indicates increasing v, and hence, a rapid increase in S_l. This increase in v indicates that one or several sensor measurements have started to deviate from the normal operating data rapidly. Due to latency in physical components, a_{max} is a better indication of a fault than the highest increase in v, since there is a time delay before the fault will result in large sensor measurements deviations. The proposed algorithm is considered more generic than previous fault indications based on threshold values.

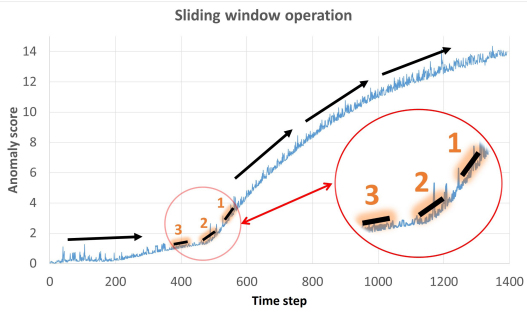


FIGURE 4. Illustration of the sliding window operation. Three windows (highlighted in orange) slide across S_l through time.

Algorithm 1 Algorithm for Calculating the Time Step With the Highest Acceleration in Faulty Degradation Data

Input: w, S_l, T_l, T_{fdd}
Output: $\hat{f}_t$
Initialisation :
$a_{max} \leftarrow 0$
$w \leftarrow T_{fdd}$ / 35
Creating three sliding windows of length w which slide across S_l for each time step in T_l.
A distance equal to w is used between each sliding window.
S_{avg} is calculated in each sliding window.
for $i := 0$ to T_{fdd} **do**
  $v1 \leftarrow S_{avg}1$ - $S_{avg}2$
  $v2 \leftarrow S_{avg}2$ - $S_{avg}3$
  $a \leftarrow v1$ - $v2$
  if $(a > a_{max})$ **then**
    $a_{max} \leftarrow a$
    $\hat{f}_t \leftarrow T_l[i]$ - $(w \cdot 2.5)$
    w is multiplied by 2.5 in order to find the center of the sliding-window operation.
  end if
end for
return $\hat{f}_t$

E. EXPERIMENTAL RESULTS AND DISCUSSION

The predicted fault time step $\hat{f}_t$ for each reconstruction model is shown in Table 3. In order to evaluate the results, valuable domain knowledge, provided by the industrial company, is used to determine the true fault time step f_t for each data set. In Table 3, the predicted fault time step is highlighted when $\hat{f}_t = f_t$. Four different real-life situations are created by applying 100%, 90%, 80%, and 70% SNRs to the training data in order to test the robustness of each reconstruction model. Additionally, to minimize any prediction performance bias, the training and evaluation process for each real-life situation is repeated five times for each reconstruction model. Then, the average $\hat{f}_t$ is calculated, as shown in Table 3. With reduced SNR, the input and target vector for reconstruction are corrupted stochastically, meaning $\tilde{\boldsymbol{x}}_t = \boldsymbol{x}_t = \boldsymbol{y}_t$. An alternative approach is to train the reconstruction models under a de-noising criterion [17], that is, the input vector is stochastically corrupted $\tilde{\boldsymbol{x}}_t = \boldsymbol{x}_t$ while the target vector is kept as the original input $\boldsymbol{y}_t = \boldsymbol{x}_t$. However, when trained in an unsupervised manner, this criterion is considered unrealistic, given the likelihood of noisy input data in real-life situations.

As seen in Table 4, $E_{T_{nod}}$ increases along with reduced SNR for the deep models, AE, VAE, and LSTM. To this extent, reduced SNR is a regularization technique that reduces overfitting. Thus, the deep models achieve robust feature extractions and are forced to generalize on the trends in the training data. Therefore, as seen in Table 3, the deep models actually improve or maintain the same prediction performance on the faulty degradation data even as the SNR on the training

TABLE 3. Predicted fault time step $\hat{f}_t$ compared to true fault time step f_t on the faulty degradation data for each reconstruction model.

Data set	Fault type	T_{fdd}	f_t	SNR(%)	$\hat{f}_t$ 1FNN	AE	VAE	LSTM
1	A	665	157	100	148	151	154	158
				90	367	151	153	158
				80	412	154	152	158
				70	476	154	153	158
2	B	682	148	100	**148**	146	**148**	155
				90	152	150	147	**148**
				80	381	150	146	150
				70	463	149	146	161
3	B	1394	477	100	492	455	**477**	481
				90	480	479	479	481
				80	632	479	479	481
				70	791	481	482	481
4	A	1916	1306	100	1281	1281	1281	1281
				90	1278	1281	1281	1281
				80	1280	1282	1281	1281
				70	1282	1281	1281	1281
5	B	2732	787	100	807	752	807	732
				90	866	655	783	739
				80	932	728	796	740
				70	1043	732	800	742

TABLE 4. Total reconstruction error $E_{T_{nod}}$ on the training data for each reconstruction model.

Data set	Fault type	T_{nod}	SNR(%)	$E_{T_{nod}}$ 1FNN	AE	VAE	LSTM
1	A	222	100	0.74	22.40	26.60	39.90
			90	0.41	52.04	62.52	81.89
			80	0.43	100.23	114.86	143.91
			70	0.45	166.26	187.41	225.86
2	B	227	100	0.93	18.50	47.30	60.50
			90	0.44	62.50	79.34	101.61
			80	0.42	108.37	131.51	165.66
			70	0.45	178.53	204.50	252.08
3	B	465	100	1.85	23.30	41.80	77.60
			90	0.84	103.35	115.65	153.64
			80	0.86	218.79	229.03	293.27
			70	0.93	380.95	385.17	476.72
4	A	638	100	2.55	7.20	5.90	15.40
			90	9.84	128.62	109.80	135.08
			80	13.95	301.69	268.16	325.66
			70	11.31	545.57	489.48	592.33
5	B	911	100	8.63	29.90	30.20	204.20
			90	5.07	199.88	182.16	378.73
			80	1.76	465.41	396.02	644.59
			70	1.93	784.34	714.80	1015.63

data reduces. Table 4 also shows that $E_{T_{nod}}$ decreases along with reduced SNR in data sets 1, 2, 3, and 5 for the 1FNN. Thus, the 1FNN learns the noise rather than the trends in the training data. This noise, obviously, is not part of the faulty degradation data. Therefore, as seen in Table 3, the 1FNN provides worse and less consistent prediction performance on the faulty degradation data as the SNR on the training data reduces. Nevertheless, $E_{T_{nod}}$ increases with reduced SNR in data set 4 for the 1FNN. This results in equal prediction performance on the faulty degradation data as the deep models.

The accuracy evaluations on the faulty degradation data in the four real-life situations for each reconstruction model are shown in Tables 5, 6, 7, and 8, respectively. The accuracy is defined as follows:

$$Accuracy\,(\%) = \left(1 - \frac{||\hat{f}_t - f_t||}{T_{fdd}}\right) \cdot 100 \qquad (18)$$

The 1FNN provides inconsistently average accuracy performance in the four situations. The average accuracy decreases along with reduced SNR, and hence, confirms the influences of noise. As opposed to the 1FNN, the deep models

TABLE 5. Accuracy evaluation on the faulty degradation data with 100% SNR applied to the training data for each reconstruction model.

100% SNR Data set	Accuracy (%) 1FNN	AE	VAE	LSTM
1	99.647	99.098	99.549	99.850
2	100	99.707	100	98.974
3	98.924	98.422	100	99.713
4	98.695	98.695	98.695	98.695
5	99.268	98.719	99.268	97.987
Avg. Accuracy	99.107	98.928	**99.502**	99.044

TABLE 6. Accuracy evaluation on the faulty degradation data with 90% SNR applied to the training data for each reconstruction model.

90% SNR Data set	Accuracy (%) 1FNN	AE	VAE	LSTM
1	68.421	99.098	99.398	99.850
2	99.413	99.707	99.853	100
3	99.785	99.857	99.857	99.713
4	98.434	98.695	98.695	98.695
5	97.108	95.168	99.854	98.243
Avg. Accuracy	92.632	98.505	**99.531**	99.300

TABLE 7. Accuracy evaluation on the faulty degradation data with 80% SNR applied to the training data for each reconstruction model.

80% SNR Data set	Accuracy (%) 1FNN	AE	VAE	LSTM
1	61.654	99.549	99.248	99.850
2	65.839	99.707	99.707	99.707
3	88.881	99.856	99.857	99.713
4	98.643	98.695	98.695	98.695
5	94.693	97.840	99.671	98.280
Avg. Accuracy	81.941	99.130	**99.435**	99.249

TABLE 8. Accuracy evaluation on the faulty degradation data with 70% SNR applied to the training data for each reconstruction model.

70% SNR Data set	Accuracy (%) 1FNN	AE	VAE	LSTM
1	52.030	99.549	99.399	99.850
2	53.812	99.853	99.707	98.094
3	77.475	99.713	99.641	99.713
4	98.747	98.695	98.695	98.695
5	90.629	97.987	99.524	98.353
Avg. Accuracy	74.539	99.159	**99.393**	98.941

TABLE 9. Average training time per epoch TT_{avg} for each reconstruction model.

Data set	TT_{avg} (seconds) 1FNN	AE	VAE	LSTM
1	1.0	2.8	1.6	48.9
2	0.9	2.8	1.7	47.8
3	2.1	5.4	3.2	100.0
4	3.2	8.3	4.5	143.0
5	4.4	11.7	6.9	201.9

provide consistently average accuracy performance in all situations. Thus, the deep models confirm robustness towards noisy real operational input data. The VAE proves to be the most reliable ED-structured reconstruction model since it provides a slightly better overall accuracy performance than the AE and LSTM. In addition to the accuracy, the average training time per epoch TT_{avg} needs to be considered for each reconstruction model. Table 9 shows TT_{avg} in the five data sets. Both AE and VAE provides satisfactory training

time. Compared to the AE and VAE, the LSTM provides extremely slow training time in all data sets. This is due to the internal cell structure of the LSTM, which results in a high amount of trainable parameters when structured as an ED. Thus, an ED structured LSTM is not recommended when it is trained in an unsupervised reconstruction-based manner. The total amount of trainable parameters for each reconstruction model is shown in Table 10.

TABLE 10. Total amount of trainable parameters for each reconstruction model.

Reconstruction model	Parameters
1FNN	420
AE	955
VAE	1010
LSTM	5796

As previously mentioned, each data set starts with different operational conditions and corresponding sensor measurements. The performance of the VAE range between 98.695% and 100% accuracy throughout the five data sets in the four different real-life situations. Thus, the proposed algorithm proves high independence towards varying operational conditions, which are expected in the maritime environment. Overall, the algorithm has proven to be highly suitable to automatically detect faults within maritime components. By combining the algorithm with fault isolation based on valuable human domain knowledge, it establishes performance strong enough to be included as a pure data-driven diagnostics approach in future end-to-end PHM system solutions where the a_{max} value could be used as the fault indicator.

V. CONCLUSION AND FUTURE WORK

This paper has investigated the possibilities for automatic fault detection within maritime components. Due to different environmental conditions with varying operational loads, and the common lack of fault labels in the maritime industry, maritime components require application-independent and intelligent fault detection algorithms in order to detect and report faults automatically. Therefore, an unsupervised reconstruction-based fault detection algorithm has been proposed in this paper. The algorithm has been applied to four different ED structured reconstruction models. The experiments were performed on five different data sets of real operational run-to-failure data of the same maritime component collected from a highly regarded industrial company. Each data set was subjected to a fault at an unknown time step. Different magnitudes of random white Gaussian noise have been applied to each data set in order to create four real-life situations. First, each reconstruction model is trained on normal operating data in an unsupervised manner. Then, the algorithm estimates an anomaly score function by calculating a reconstruction error at each time step in faulty degradation data. Finally, the algorithm detects a fault automatically by estimating the time step with the highest acceleration in the anomaly score function. The acceleration is chosen as the fault indicator due to latency in physical components. Thus, there is an expected time delay before a fault will result in large sensor measurements deviations. By this approach, the algorithm is considered more generic compared to previous user-specified threshold values. Additionally, the algorithm is independent of any prior domain knowledge of component degradation mechanics.

The algorithm achieved an average accuracy between 99.393% and 99.531% when compared to the true fault time step based on valuable human domain knowledge. Overall, the algorithm has both proven to be robust towards noisy real operational input data and independent of varying operational conditions. Thus, the algorithm, in combination with fault isolation based on valuable human domain knowledge, is highly suitable to be included as a pure data-driven diagnostics approach in future end-to-end PHM system solutions for maritime applications. In such a system, the value of the highest acceleration will be used as the fault indicator. Additionally, the corresponding time step to the fault indicator can be further used to create run-to-failure labels for any data-driven prognostics algorithm automatically. Future work will address these issues.

ACKNOWLEDGMENT

The authors would like to thank Digital Twins For Vessel Life Cycle Service (DigiTwin).

REFERENCES

[1] A. L. Ellefsen, V. Æsøy, S. Ushakov, and H. Zhang, "A comprehensive survey of prognostics and health management based on deep learning for autonomous ships," *IEEE Trans. Rel.*, to be published.
[2] K. E. Knutsen, G. Manno, and B. J. Vartdal, "Beyond condition monitoring in the maritime industry," *DNV GL Strategic Res. Inovation Position Paper*, 2014. [Online]. Available: https://www.researchgate.net/publication/263583976_Beyond_condition_monitoring_in_the_maritime_industry
[3] R. Kothamasu, S. H. Huang, and W. H. VerDuin, "System health monitoring and prognostics—A review of current paradigms and practices," *Int. J. Adv. Manuf. Technol.*, vol. 28, nos. 9–10, pp. 1012–1024, 2006.
[4] T. M. Allen, "Us navy analysis of submarine maintenance data and the development of age and reliability profiles," U.S. Navy SUBMEPP, Kittery, ME, USA, Tech. Rep., 2001.
[5] 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," *Rel. Eng. Syst. Saf.*, vol. 183, pp. 240–251, Mar. 2019.
[6] B.-M. Batalden, P. Leikanger, and P. Wide, "Towards autonomous maritime operations," in *Proc. IEEE Int. Conf. Comput. Intell. Virtual Environ. Meas. Syst. Appl. (CIVEMSA)*, Mar. 2017, pp. 1–6.
[7] X. Li, Q. Ding, and J.-Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Rel. Eng. Syst. Saf.*, vol. 172, pp. 1–11, Apr. 2018.
[8] E. Delechelle, J. Lemoine, and O. Niang, "Empirical mode decomposition: An analytical approach for sifting process," *IEEE Signal Process. Lett.*, vol. 12, no. 11, pp. 764–767, Nov. 2005.
[9] J. Gilles, "Empirical wavelet transform," *IEEE Trans. Signal Process.*, vol. 61, no. 16, pp. 3999–4010, Aug. 2013.
[10] A. S. Zymaris, Ø. Å. Alnes, K. E. Knutsen, and N. M. P. Kakalis, "Towards a model-based condition assessment of complex marine machinery systems using systems engineering," in *Proc. 3rd Eur. Conf. Prognostics Health Manage. Soc.*, Bilbao, Spain, 2016, pp. 1–15.
[11] G. Wu, "Fault detection method for ship equipment based on BP neural network," in *Proc. Int. Conf. Robots Intell. Syst. (ICRIS)*, May 2018, pp. 556–559.
[12] A. Brandsæter, G. Manno, E. Vanem, and I. K. Glad, "An application of sensor-based anomaly detection in the maritime industry," in *Proc. IEEE Int. Conf. Prognostics Health Manage. (ICPHM)*, Jun. 2016, pp. 1–8.

[13] C. Yang, J. Liu, Y. Zeng, and G. Xie, "Real-time condition monitoring and fault detection of components based on machine-learning reconstruction model," *Renew. Energy*, vol. 133, pp. 433–441, Apr. 2019.

[14] *Case Western Reserve University Bearing Data Center*. Accessed: Dec. 4, 2018. [Online]. Available: https://csegroups.case.edu/bearingdatacenter/pages/download-data-file

[15] C. Lu, Z.-Y. Wang, W.-L. Qin, and J. Ma, "Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification," *Signal Process.*, vol. 130, pp. 377–388, Jan. 2017.

[16] H. Liu, J. Zhou, Y. Zheng, W. Jiang, and Y. Zhang, "Fault diagnosis of rolling bearings with recurrent neural network-based autoencoders," *ISA Trans.*, vol. 77, pp. 167–178, Jun. 2018.

[17] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, no. 12, pp. 3371–3408, Dec. 2010.

[18] C. Li, R.-V. Sanchez, G. Zurita, M. Cerrada, D. Cabrera, and R. E. Vásquez, "Gearbox fault diagnosis based on deep random forest fusion of acoustic and vibratory signals," *Mech. Syst. Signal Process.*, vols. 76–77, pp. 283–293, Aug. 2016.

[19] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an LSTM-based variational autoencoder," *IEEE Robot. Automat. Lett.*, vol. 3, no. 3, pp. 1544–1551, Jul. 2018.

[20] P. Malhotra *et al.*, "Multi-sensor prognostics using an unsupervised health index based on LSTM encoder-decoder," in *Proc. Workshop Mach. Learn. Prognostics Health Manage.*, 2016, pp. 1–10.

[21] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient BackProp," in *Neural Networks: Tricks of the Trade*. Berlin, Germany: Springer, 2012, pp. 9–48.

[22] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[23] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[24] D. P. Kingma and M. Welling. (2013). "Auto-encoding variational Bayes." [Online]. Available: https://arxiv.org/abs/1312.6114

[25] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," in *Proc. 9th Int. Conf. Artif. Neural Netw. (ICANN)*, vol. 2. Sep. 1999, pp. 850–855.

[26] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Netw.*, vol. 18, no. 5, pp. 602–610, 2005.

[27] F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count," in *Proc. IEEE-INNS-ENNS Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 3. Jul. 2000, pp. 189–194.

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

[29] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

[30] *Eclipse Deeplearning4j Development Team, Deeplearning4j: Open-Source Distributed Deep Learning for the JVM Apache Software Foundation License 2.0*, 2018. [Online]. Available: http://deeplearning4j.org

ANDRÉ LISTOU ELLEFSEN received the master's degree in subsea technology from the Norwegian University of Science and Technology (NTNU), Trondheim, Norway, in 2016. He is currently pursuing the Ph.D. degree with NTNU, Åalesund, Norway, as part of the Mechatronics Laboratory, Department of Ocean Operations and Civil Engineering. His current research interests include artificial intelligence, deep learning, decision support, predictive maintenance, and digital twins.

EMIL BJØRLYKHAUG is currently pursuing the Ph.D. degree with the Department of Ocean Operations and Civil Engineering, Norwegian University of Science and Technology. His current research interests include robotic technologies for automating fish processing, tools that may facilitate industrial robots in performing more complex tasks, deep learning, and computer vision.

VILMAR ÆSØY graduated from the Norwegian University of Science and Technology (NTNU), in 1989, and continued his research on natural gas fueled marine engines at NTNU/MARINTEK, until 1997. He received the Ph.D. degree for his research on natural gas ignition and combustion through experimental investigations and numerical simulations, in 1996. From 1989 to 1997, he was engaged in several large R&D projects developing gas fueled engines and fuel injection systems for the diesel engine manufacturers, Wärtsilä, and Bergen Diesel, Roll-Royce. From 1998 to 2002, he was a R&D Manager for Rolls-Royce Marine Deck Machinery. Since 2002, he has been employed in teaching with the Aalesund University College, where he is also developing and teaching courses in marine product and systems design on bachelor's and master's level. In 2010, he received the green ship machinery professorship. His current research interest includes energy and environmental technology, with focus on combustion engines and the need for more environmental friendly and energy efficient systems.

HOUXIANG ZHANG (M'04–SM'12) received the Ph.D. degree in mechanical and electronic engineering, in 2003, and the Habilitation degree in informatics from the University of Hamburg, in 2011. Since 2004, he has been a Postdoctoral Fellow and a Senior Researcher with the Department of Informatics, Faculty of Mathematics, Informatics and Natural Sciences, Institute of Technical Aspects of Multimodal Systems, University of Hamburg, Germany. He was with the Aalesund University College, in 2016. In 2011, he joined the Norwegian University of Science and Technology, Norway, where he is currently a Professor on mechatronics. His current research interests include biological robots and modular robotics, especially on biological locomotion control, and virtual prototyping in demanding marine operation. He has applied for and coordinated more than 20 projects supported by the Norwegian Research Council, German Research Council, and industry. In these areas, he has published over 160 journal and conference papers as author or co-author. He received four best paper awards and four finalist awards for Best Conference Paper at the International Conference on Robotics and Automation.

D

Paper IV

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 25, 2019, accepted May 28, 2019, date of publication May 31, 2019, date of current version June 12, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2920297

Validation of Data-Driven Labeling Approaches Using a Novel Deep Network Structure for Remaining Useful Life Predictions

ANDRÉ LISTOU ELLEFSEN[1], SERGEY USHAKOV[2], VILMAR ÆSØY[1], AND HOUXIANG ZHANG[1], (Senior Member, IEEE)

[1]Mechatronics Laboratory, Department of Ocean Operations and Civil Engineering, Norwegian University of Science and Technology Ålesund, 6009 Ålesund, Norway
[2]Department of Marine Technology, Norwegian University of Science and Technology, 7491 Trondheim, Norway

Corresponding author: Andre Listou Ellefsen (andre.ellefsen@ntnu.no)

This work was supported in part by the Department of Ocean Operations and Civil Engineering, Norwegian University of Science and Technology, under Project 90329106, and in part by the Research Council of Norway under Grant 280703.

ABSTRACT Today, most research studies that aim to predict the remaining useful life (RUL) of industrial components based on deep learning techniques are using piecewise linear (PwL) run-to-failure targets to model the degradation process. However, this PwL degradation model assumes a constant initial RUL value in which only time is needed to model normal operating conditions. Thus, it ignores the entire diagnostics aspect. To provide high and reliable RUL prediction accuracy, a prognostics algorithm must incorporate diagnostics information. This paper will provide the Prognostics and Health Management Community an empirical study that validates the PwL degradation model against other, more recent data-driven labeling approaches. We compare three different data-driven labeling approaches for RUL predictions. First, an unsupervised reconstruction-based fault detection algorithm is used to provide valuable diagnostics information. Then, optimized initial RUL values are calculated based on this information. Finally, these values are used to construct PwL, descriptive statistics, and anomaly score function run-to-failure targets for subset FD001 in the popular and publicly available C-MAPSS data set. A deep network structure is proposed and trained on the three different run-to-failure targets in order to predict the RUL. During the training process, a genetic algorithm approach is used to tune a selected search space of hyper-parameters. The results suggest that the network trained on PwL run-to-failure targets with the optimized initial RUL values performs the best and provides the most reliable RUL prediction accuracy. This network also outperforms the most robust results in the literature.

INDEX TERMS Data-driven labeling approaches, deep learning, fault detection, prognostics and health management, remaining useful life.

I. INTRODUCTION

Data-driven Prognostics and Health Management (PHM) applications use algorithms built on sensor measurements to perform fault detection, condition assessment, and remaining useful life (RUL) predictions [1]. Prognostics algorithms predict the progression of faults. Thus, the associated RUL predictions tend to achieve the ideal maintenance policy through predictions of the available time until failure after a fault is detected within the component [2]. In this way, PHM applications have the potential to prevent failures before they occur, and hence, considerably increase operational availability, reliability, and life expectancy of industrial systems.

During the last three years, state-of-the-art deep learning (DL) techniques have outperformed traditional data-driven prognostics algorithms in RUL predictions for engine degradation [3]–[5]. Researchers have typically used the publicly available Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) data set, produced and provided by NASA [6], to train and evaluate the proposed DL approaches. The C-MAPSS data set consists of numerous time series of aircraft gas turbine engines where the engines

The associate editor coordinating the review of this manuscript and approving it for publication was Dong Wang.

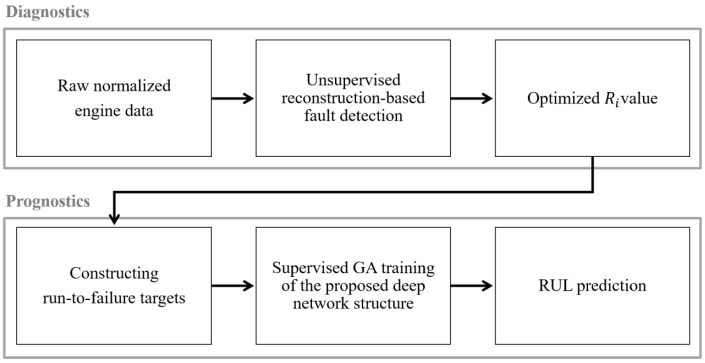


FIGURE 1. An overview of the complete training structure.

are subjected to a varying number of time steps and different degrees of degradation. Within the PHM research field, the C-MAPSS data set is acknowledged as the benchmark data set for data-driven prognostics algorithms.

Today, DL techniques that aim to predict RUL still depend on large amounts of run-to-failure targets in order to model the degradation process in the supervised training procedure. Hence, most studies construct run-to-failure targets based on the piece-wise linear (PwL) degradation model, which Heimes [7] proposed in 2008. This degradation model assumes a constant initial RUL (R_i) value when the engines operate in normal conditions. Then, the model degrades linearly until failure after the engines are subjected to a fault, namely, after the fault time step. A subsequent assumption is that all engines utilize the same constant R_i value. In other words, the constructed run-to-failure targets depend on the total number of time steps in each engine and not on the actual degradation process. By the latter assumption, the entire diagnostics aspect is ignored. In real-life PHM applications, any supervised prognostics algorithm should depend on an accurate fault detection algorithm in order to construct reliable run-to-failure targets. Then, the prognostics algorithm is able to model the true degradation process and potentially achieve higher and more reliable RUL prediction accuracy. Therefore, it would be highly beneficial for the PHM community to possess a study that validates the PwL degradation model against other and more recent data-driven labeling approaches.

The objective of this paper is to make a thorough comparison of three different data-driven labeling approaches, based on accurate fault detection, for RUL predictions. First, raw normalized engine data will act as the input for an unsupervised reconstruction-based fault detection algorithm in order to predict the fault time step for each engine [8]. Next, an optimized R_i value for each engine can be obtained. These values are then used to construct PwL, descriptive statistics (DS) [9], in order to model degradation by finding some consistency in the phenomenon leading to failure, and anomaly score function (ASF), which is obtained from the unsupervised reconstruction-based fault detection algorithm, run-to-failure targets for subset FD001 in the C-MAPSS data set. Additionally, this paper proposes a deep network structure for RUL predictions, which will be trained on the three different data-driven labeling approaches. A Genetic Algorithm (GA) approach [5] will also be used to tune hyper-parameters during the supervised training process since each labeling approach requires different values of hyper-parameters within the deep network structure in order to perform with the highest RUL prediction accuracy possible. A flow chart of the complete training structure, where the final RUL prediction incorporates valuable diagnostics information is shown in Figure 1. Finally, the proposed deep network structure trained on the run-to-failure targets with the highest RUL prediction accuracy will be compared to the most robust results in the literature. This is done to demonstrate that prognostics algorithms achieve higher RUL prediction accuracy when trained on run-to-failure targets based on accurate fault detection. This study's main contributions are as follows:

- A comprehensive comparison between PwL, DS, and ASF run-to-failure targets with optimized R_i values is conducted.
- A deep network structure for RUL predictions is proposed.
- The network trained on PwL run-to-failure targets with optimized R_i values outperforms both the networks trained on DS and ASF run-to-failure targets, as well as, the most robust results in the literature with respect to RUL predictions on subset FD001 in the C-MAPSS data set.

The overall organization of the paper is as follows. Section II introduces recent and related work on subset FD001. Section III introduces the necessary background on Feed-forward Neural Network (FNN), Convolutional Neural Network (CNN), Long-Short Term Memory (LSTM), and the proposed deep network structure. The experimental study is elaborated in Section IV. Section V, considers important experimental results and discussions. Finally, Section VI concludes the paper and provides directions for future work.

II. RELATED WORK

Subset FD001 in the C-MAPSS data set has been frequently used to evaluate most DL approaches proposed for RUL predictions in recent years. In data-driven PHM applications, time series data is the standard input format. The LSTM [10] is a well-established DL technique that essentially was designed to process time series data. Zheng *et al.* [11] stacked two LSTM layers, two FNN layers, and a final output layer in order to provide RUL predictions. The proposed approach achieved higher RUL prediction accuracy compared to the Hidden Markov Model and a traditional Recurrent Neural Network (RNN).

A Deep Belief Network (DBN) [12] consists of stacked Restricted Boltzmann Machines (RBMs). Zhang *et al.* [3] proposed a multiple objective evolutionary ensemble learning frameworks for the DBN training process. Consequently, the proposed approach constructs multiple DBNs of varying accuracy and diversity before the evolved DBNs are combined to perform RUL predictions. The proposed approach outperformed several traditional machine learning algorithms, such as Support Vector Machine and Multilayer Perceptron.

During the past decade, CNNs have outperformed more traditional approaches in several domains, including object recognition [13] and face recognition [14]. However, CNNs have also more recently performed excellently on prognostics problems. Li *et al.* [4] proposed a new CNN approach in order to provide RUL predictions. In this approach, all convolution operations are performed in one dimension. Thus, the CNN extracts and learns low-level to high-level representations of each raw sensor measurement from the very start rather than learning the spatial relationship between the sensor measurements and then extracting prognostics information.

Yoon *et al.* [15] used a semi-supervised learning approach to predict the RUL. Their approach included an embedding network obtained from a Variational Autoencoder (VAE) followed by an RNN which was trained based on the latent space defined by the VAE. However, the main goal of this study was to show high RUL prediction accuracy with limited run-to-failure targets in the training procedure.

Ellefsen *et al.* [5] also used a semi-supervised learning approach to predict the RUL. An initial RBM layer was used as an unsupervised pre-training stage in order to initialize the weights in a region near a good local minimum before supervised fine-tuning of the whole network was conducted. The remaining layers of their network consisted of two LSTM layers, one FNN layer, and a final output layer to perform RUL predictions. Additionally, a GA approach was used to tune a big search space of hyper-parameters.

All above-mentioned studies utilize the PwL degradation model with the same constant R_i value for all engines. Even though the constant R_i value varies among different studies, the diagnostics aspect is ignored in these studies. However, one study uses a different degradation model to predict the RUL. Malhotra *et al.* [16] used an LSTM encoder-decoder (LSTM-ED) approach to reconstruct the engines. A reconstruction error was then used to compute a health index (HI) curve for both the training and test set. Then, the HI curves were subjected to normalization and linear regression. Finally, RUL estimations were performed by matching the HI curves. Similar to [16], this study also utilizes a reconstruction error at each time step for each engine to construct an ASF [8]. The ASF will both be used to predict an optimized R_i value for each engine and to create run-to-failure targets as one of the data-driven labeling approaches compared in this study.

III. BACKGROUND

This section will introduce the necessary background on the proposed deep network. First, FNN and the main DL techniques, 1D CNN and LSTM, are defined. Finally, the proposed deep network structure is elaborated.

A. FEED-FORWARD NEURAL NETWORK

FNNs form the basis of the DL techniques used in this study. The objective of this network is to approximate a function f^* by mapping an input $\boldsymbol{x}$ to a target y, that is, $y = f^*(\boldsymbol{x})$. An FNN defines a mapping $\boldsymbol{y} = f(\boldsymbol{x}; \boldsymbol{\theta})$ and learns the value of the parameters $\boldsymbol{\theta}$ (weights and biases) through the back-propagation algorithm [17]. FNNs are typically called networks since they are represented by stacking several layers [18]. Each unit in layer l computes its own activation value:

$$a_j^l = \sigma(z_j^l) \tag{1}$$

where σ is the activation function and the argument is the weighted sum

$$z_j^l = b_j^l + \sum_k w_{jk}^l a_k^{l-1} \tag{2}$$

of the output a_k^{l-1} from unit k in the previous layer $l-1$. b_j^l is the bias and w_{jk}^l are the weight factors. In the first hidden layer $l = 1$, the input is $a_j^0 = x_j$, where $x_j, j = 1 \ldots n$, are the inputs to the FNN. As each layer is fully connected, the weighted sum of the outputs of layer $l-1$ is over all units k.

B. CONVOLUTIONAL NEURAL NETWORK

CNNs are a specialized kind of FNNs designed for processing multiple arrays of 1D, 2D, or 3D grid-like topology data [18]. Examples of a 1D, 2D, and 3D grid are time series data where each feature is considered as a 1D grid of time steps at regular time intervals, image data is considered as a 2D grid of pixels, and video or volumetric images, respectively. Regardless of the input data, 1D, 2D, and 3D CNNs share the same key advantages, including convolution operations, shared weights, pooling, and the use of many layers [19]. However, the main difference is how the kernel (filter) slides across the data, namely, how the convolution operation is performed.

Today, sensor data is the most common data type format for data-driven PHM applications [2]. Subset FD001 contains

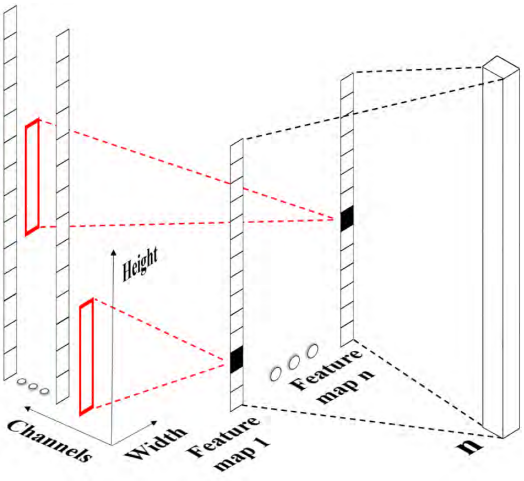


FIGURE 2. An illustration of the 1D convolution operation for multivariate time series data. The red rectangles represent 1D kernels.

several shorter time series of the overall data, where each time series (engine) is subjected to several sensor measurements. The spatial relationship between the sensor measurements is not of great importance [4]. Therefore, 1D CNN is highly suitable and will be used in this study. With respect to mathematical understanding, the convolution operation is typically denoted with an asterisk, and hence, the discrete 1D convolution operation can be defined as [18]:

$$\boldsymbol{s}(t) = (\boldsymbol{x} * \boldsymbol{k})(t) = \sum_{a} \boldsymbol{x}(t-a)\boldsymbol{k}(a) \tag{3}$$

where $\boldsymbol{x} = [x_1 \dots x_t]$ is a 1D input vector of time steps t, and $\boldsymbol{k}$ is a 1D kernel. The kernel is defined by its height k_h and slides through the whole input vector with a stride equal to one in 1D CNNs. The complete output, $\boldsymbol{s}(t)$, is usually referred to as the feature map. Figure 2 illustrates the 1D convolution operation for multivariate time series data. The height equals the number of time steps, the width is equal to one, and the amount of channels (depth) equals the number of input features. Due to the relatively low input dimension in FD001, pooling will not be used in this study. Like FNNs, CNNs are also trained by the back-propagation algorithm, but the reduced number of parameters and shared weights improve the training efficiency. It should also be noted that CNNs are capable of handling raw normalized input data. Hence, data pre-processing is rare.

C. LONG-SHORT TERM MEMORY

In recent times, the original LSTM [10] has been subjected to adjustments by [20]–[22], and the literature refers to this as the "vanilla LSTM." This study utilizes "vanilla LSTM" with no peephole connections. The LSTM introduces a memory cell that controls the information flow in and out of the cell. Hence, the memory cell is able to maintain its state over time, such that it learns long-term dependencies, and this

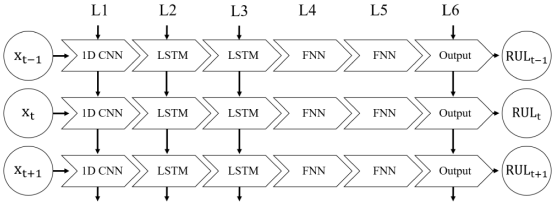


FIGURE 3. The proposed deep network structure.

feature is its superior strength compared to traditional RNNs. The memory cell consists of three non-linear gating units that control and protect the cell state, $\boldsymbol{S}_t$ [23]:

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_f \boldsymbol{x}_t + \boldsymbol{R}_f \boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \tag{4}$$

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_i \boldsymbol{x}_t + \boldsymbol{R}_i \boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \tag{5}$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_o \boldsymbol{x}_t + \boldsymbol{R}_o \boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \tag{6}$$

where σ is the logistic sigmoid gate activation function, $\sigma(x) = \frac{1}{1+e^{-x}}$, which provides a scaled value between 0 and 1. $\boldsymbol{W}$ is the input weight, $\boldsymbol{R}$ is the recurrent weight, and $\boldsymbol{b}$ is the bias weight. The new candidate state values, $\tilde{\boldsymbol{S}}_t$, are created by the tanh layer:

$$\tilde{\boldsymbol{S}}_t = tanh(\boldsymbol{W}_s \boldsymbol{x}_t + \boldsymbol{R}_s \boldsymbol{h}_{t-1} + \boldsymbol{b}_s) \tag{7}$$

The previous cell state, $\boldsymbol{S}_{t-1}$, is updated into the new cell state, $\boldsymbol{S}_t$, by

$$\boldsymbol{S}_t = \boldsymbol{f}_t \otimes \boldsymbol{S}_{t-1} + \boldsymbol{i}_t \otimes \tilde{\boldsymbol{S}}_t \tag{8}$$

where $\otimes$ indicates element-wise multiplication of two vectors. First, $\boldsymbol{f}_t$ decides which historical information the memory cell should forget. Next, $\boldsymbol{i}_t$ determines what new information in $\tilde{\boldsymbol{S}}_t$ the memory cell will input and store in $\boldsymbol{S}_t$. Finally, $\boldsymbol{o}_t$ decides which parts of $\boldsymbol{S}_t$ the memory cell will output:

$$\boldsymbol{h}_t = \boldsymbol{o}_t \otimes tanh(\boldsymbol{S}_t) \tag{9}$$

Through these steps, the LSTM has the power to remove or add information to $\boldsymbol{S}_t$, which makes it extremely fit to process time series data. Like FNNs and CNNs, the LSTM is trained by the back-propagation algorithm.

D. THE PROPOSED DEEP NETWORK STRUCTURE

The proposed deep network structure is shown in Figure 3. In the first layer (L1), a 1D CNN will be utilized to extract and learn low-level temporal features from each sensor measurement individually [4]. These features might contain important degradation information which will then be used to form more complex patterns within the next layers. In both the second and the third layer (L2 and L3), an LSTM layer is used to reveal hidden information and learn long-term dependencies within the features obtained from L1 [5], [11]. Next, an FNN layer is used in both the fourth (L4) and the fifth (L5) layers in order to map all extracted features. In addition, the well-proven regularization technique dropout [24] is applied to L5. Dropout randomly drops units during training. In this way,

dropout approximately connects an exponential number of different structures. Thus, the network learns to make generalized representations of the input data, which will prevent the network from extracting the same degradation features repeatedly. In the final layer (L6), a time distributed, fully connected output layer is attached to handle error calculations and perform RUL predictions.

IV. EXPERIMENTAL STUDY

In the following experimental study, all experiments are run on NVIDIA GeForce GTX 1060 6 GB and the Microsoft Windows 10 operating system. The programming language is Java 8 and the deep learning library is "deeplearning4j" (DL4J) version 1.0.0-SNAPSHOT [25]. It should be noted that the DL techniques included in the proposed deep network structure are optimized by the NVIDIA CUDA Deep Neural Network library (cuDNN) [26]. cuDNN is a GPU-accelerated library of primitives for DL techniques. In DL4J, time series data has the following input shape: [miniBatchSize, inputSize, timeSeriesLength], where miniBatchSize is the number of time series in a mini batch, input size is the number of columns, and timeSeriesLength is the total number of time steps in the mini batch. If time series in a mini-batch have variable time step length, the shorter time series are padded with zeros such that the time step lengths are equal to the longest among them. Consequently, mask arrays are used during training. These additional arrays record whether a time step is really present, or whether it is just padding.

A. SUBSET FD001 IN THE BENCHMARK C-MAPSS DATA SET

Subset FD001 consists of 100 time series from aircraft gas turbine engines in both the training and test set. Each engine starts with different degrees of initial wear and manufacturing variation. These initial degradation mechanics are unknown to the public. All engines operate in normal condition at the start, then begin to degrade at an unknown time step during the time series. The degradation in the training set grows in magnitude, namely with increasing acceleration, until failure. The degradation in the test set, however, ends sometime prior to failure. Accordingly, true RUL targets are provided at the last time step for each engine in the test set. The data is contaminated with sensor noise and subset FD001 includes 24 input features: three operational sensor settings and 21 sensor measurements. Please see [27] for a detailed description of each input feature. Table 1 summarizes subset FD001.

B. PERFORMANCE EVALUATIONS

The scoring function (S) provided in [27] and the root mean square error ($RMSE$) are used in this study as performance evaluations for the test set:

$$S = \begin{cases} \sum_{i=1}^{n} e^{(-\frac{d_i}{13})} - 1, & \text{for} \quad d_i < 0 \\ \sum_{i=1}^{n} e^{(-\frac{d_i}{10})} - 1, & \text{for} \quad d_i \geq 0 \end{cases} \tag{10}$$

TABLE 1. Subset FD001 in the C-MAPSS data set [6].

FD001	
Engines in training set	100
Total number of time steps in training set	20,631
Engines in test set	100
Total number of time steps in test set	13,096
True RUL targets in test set	100
Operating conditions	1
Fault conditions	1

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} d_i^2} \tag{11}$$

where n is the total number of true RUL targets in the test set and $d_i = RUL_{predicted,i} - RUL_{true,i}$. In both performance evaluations, the main objective is to achieve the smallest value possible, that is, when $d_i = 0$. The $RMSE$ gives equal penalty to early and late RUL predictions, namely, when $d_i < 0$ and $d_i > 0$, respectively. In S, however, the penalty for late RUL predictions is larger. This is because late RUL predictions are prone to system failures in real-life PHM applications as maintenance operations will be scheduled too late. On the other hand, early predictions pose less risk to system failures since maintenance operations will be scheduled too early.

Previously, both hold-out and k-fold cross-validation have been used for hyper-parameter tuning on subset FD001 [5], [11]. However, in this study, the total number of time steps in the training set is considered large enough to utilize a hold-out approach, that is, splitting the total training set into 80 engines for training and 20 engines for cross-validation, randomly. In addition to S and $RMSE$, the root mean square error horizon ($RMSE_{hz}$) is used in this study as a performance evaluation for both the training set and the cross-validation set:

$$RMSE_{hz} = \sqrt{\frac{1}{m}\sum_{j=1}^{m} d_j^2} \tag{12}$$

where m is the total number of constructed run-to-failure targets in both the training set and cross-validation set, and $d_j = RUL_{predicted,j} - RUL_{target,j}$. The $RMSE_{hz}$ will be used to compare the true overall prognostics accuracy of the different labeling approaches. The prognostics horizon is a critical measurement designed to evaluate the different labeling approaches with respect to both inherent uncertainties with the degradation process and potential flaws with the constructed run-to-failure targets.

C. DIAGNOSTICS - DETECTING THE FAULT TIME STEP

Ellefsen *et al.* [8] used an unsupervised reconstruction-based fault detection algorithm for maritime components. Their proposed algorithm is also used in this work in order to predict the fault time step for each engine in FD001. First, a VAE, with three hidden layers and corresponding hidden units (28,14,7) in the encoder and three hidden layers with corresponding hidden units (7,14,28) in the decoder, is trained on

normal operating data in an unsupervised manner. It should be noted that the selection process of the hidden units, $h1$, $h2$, and $h3$, is based on the following experience-based formula:

$$h1 = \mathbb{Z}(24 \cdot 1.2) \quad h2 = \mathbb{Z}\left(\frac{h1}{2}\right) \quad h3 = \mathbb{Z}\left(\frac{h2}{2}\right)$$

where 24 is the number of input features in FD001. The initial 25% of each engine is considered normal operating data. Then, the algorithm estimates a raw anomaly score function (ASF) by calculating a reconstruction error, the mean square error (MSE), at each time step for each engine:

$$MSE = \frac{1}{n}\sum_{i=1}^{n} ||\hat{y}_i - y_i||^2 \tag{13}$$

where n is the number of input features, and $\hat{y}_i$ and y_i are the i_{th} predicted and target feature measurement, respectively. Next, the algorithm creates three sliding windows of length w in order to smooth the ASF:

$$w = \frac{T_t}{p} \tag{14}$$

where T_t is the total number of time steps in each engine and p is a tune-able parameter. First, the three windows slide across the raw ASF for each time step. A distance equal to w is used between each sliding window. In order to remove a certain amount of noise in the raw ASF, the average reconstruction error is calculated in the three windows. Since p decides the length of w, it also decides the amount of smoothing performed on the raw ASF. Thus, p should be tuned carefully based on the amount of noise in the raw ASF. In this work, $p = 30$ is used for all engines in order keep the same percentage level, that is $(1/30) \cdot 100 = 3.33\%$, on T_t. This p value will not smooth the raw ASF too much, and hence, keep important degradation trends. Second, the velocity between windows 1 and 2 and between windows 2 and 3 are calculated. Finally, the acceleration between the two velocities is estimated. Please see [8] for a more detailed explanation of the algorithm.

Compared to the data sets used in [8], the nature of degradation is somewhat different in FD001. In this data set, the degradation grows with increasing acceleration until failure. Thus, the highest acceleration, which is used as the fault criterion in [8], is not suitable for FD001. Therefore, an alternative approach for predicting the fault time step $\hat{f}_t$ is used in this study. First, the highest acceleration in normal operating data a_{nod} is calculated for each engine. a_{nod} is equivalent to the maximum increase in deviation between the normal operating sensor measurements. Then, a dynamic acceleration threshold, $a_{Th} = 1.15 \cdot a_{nod}$, is used as the fault criterion in the remaining data for each engine. In this work, the value of 1.15 is based on trial an error. However, this value is a critical parameter and should be tuned carefully for other applications. This value will depend on the nature of degradation. Finally, $\hat{f}_t$ is estimated when the acceleration increases a_{Th}. Thus, the algorithm aims to detect the initial time step where one or several sensor measurements have started to deviate from the normal operating data rapidly. Table 2 shows T_t, $\hat{f}_t$, and the corresponding R_i for each engine in FD001.

TABLE 2. Total time step length T_t, predicted fault time step $\hat{f}_t$, and corresponding initial RUL value R_i for each engine in FD001.

Train set								Cross-validation set			
Engine	T_t	$\hat{f}_t$	R_i	Engine	T_t	$\hat{f}_t$	R_i	Engine	T_t	$\hat{f}_t$	R_i
1	192	63	129	53	195	99	96	2	287	171	116
3	179	47	132	54	257	93	164	5	269	141	128
4	189	62	127	55	193	77	116	7	259	124	135
6	188	86	102	57	137	47	90	13	163	69	94
8	150	53	97	58	147	94	53	21	194	98	96
9	201	86	115	59	231	117	114	31	234	149	85
10	222	70	152	60	172	85	87	33	200	110	90
11	240	120	120	61	185	104	81	42	196	103	93
12	170	94	76	62	180	99	81	46	256	91	165
14	180	76	104	63	174	84	90	47	214	129	85
15	207	86	121	64	283	140	143	50	198	79	119
16	209	72	137	66	201	83	118	52	213	79	134
17	276	123	153	67	312	130	182	56	275	90	185
18	195	71	124	69	362	245	117	65	153	59	94
19	158	39	119	71	208	96	112	68	199	71	128
20	234	104	130	72	213	83	130	70	137	84	53
22	202	103	99	73	213	104	109	74	166	75	91
23	168	94	74	75	229	94	135	81	239	100	139
24	147	60	87	76	210	147	63	93	155	57	98
25	230	129	101	77	154	38	116	94	258	101	157
26	199	101	98	78	231	91	140				
27	155	85	70	79	199	114	85				
28	165	72	93	80	185	58	127				
29	163	100	63	82	214	106	108				
30	194	116	78	83	293	176	117				
32	191	121	70	84	267	135	132				
34	194	66	128	85	188	88	100				
35	181	111	70	86	278	119	159				
36	158	52	106	87	178	105	73				
37	170	77	93	88	213	80	133				
38	194	114	80	89	217	105	112				
39	128	58	70	90	154	73	81				
40	188	79	109	91	135	46	89				
41	216	88	128	92	340	167	173				
43	207	98	109	95	283	114	169				
44	192	131	61	96	336	204	132				
45	158	73	85	97	202	66	136				
48	231	81	150	98	156	68	88				
49	215	75	140	99	185	74	111				
51	213	92	121	100	200	71	129				

D. DATA-DRIVEN LABELING APPROACHES

This study compares three different data-driven labeling approaches for constructing run-to-failure targets. The optimized R_i values in Table 2 are used to construct run-to-failure targets based on the PwL degradation model, DS, and on the raw ASF obtained from the anomaly detector in Section IV-C.

1) PIECE-WISE LINEAR

In the original PwL degradation model by Heimes [7], all engines in the training and cross-validation sets utilize the same R_i value when the engines operate in normal condition. The major limitation of this assumption is that the fault time step for each engine depends on T_t and not on the actual degradation pattern. Actually, each engine has an individual degradation pattern [5]. Therefore, the PwL degradation model used in this study utilizes an optimized R_i value for each engine. These R_i values are dependent on the actual degradation pattern in each engine. Algorithm 1 shows the procedure on how to construct PwL run-to-failure targets for engine i.

2) DESCRIPTIVE STATISTICS

DS [9] aims to find some consistency in the phenomenon leading to failure. In other words, there are typical values of

Algorithm 1 Algorithm for Constructing Piece-Wise Linear Run-to-Failure Targets for Engine i

Input: $T_t, \hat{f}_t, R_i$
Output: PwL_i
  for $t := 0$ to T_t **do**
    if $(t \leq \hat{f}_t)$ **then**
      $PwL_i \leftarrow R_i$
    else
      $PwL_i \leftarrow (T_t - t)$
    end if
  end for
  return PwL_i

the sensor measurements at the failure time step (F) for each engine in both the training set and cross-validation set. Previous research has proven that sensors 2, 3, 4, 7, 11, 12, and 15 are subjected to a clear degradation trend and that they are contaminated with less noise than the remaining sensors [28]. This sensor selection process is of high importance for the degradation precision of the subsequent constructed run-to-failure targets. First, the mean values of F in the selected sensors are calculated:

$$\begin{aligned} E(X(F)) &= \left[E(x^2(F)), \ldots, E(x^{15}(F))\right] \\ &= \left[\frac{1}{m}\sum_{i\in I} x_i^2(F_i), \ldots, \frac{1}{m}\sum_{i\in I} x_i^{15}(F_i)\right] \\ &= \left[E^2, \ldots, E^{15}\right] \end{aligned} \tag{15}$$

where m is the number of failures, I is the set of engines that experienced a failure, F_i is the failure time step of engine i, and $E(X(F))$ is the vector of mean values observed at each failure time step. Second, the mean values are used to construct run-to-failure targets at any time t up until failure for engine i:

$$\begin{aligned} Y_i(t) &= X_i(t) - E(X(F)) \\ &= \left[\left(x_i^2(t) - E^2\right)^2 + \cdots + \left(x_i^{15}(t) - E^{15}\right)^2\right]^{\frac{1}{2}} \end{aligned} \tag{16}$$

where $Y_i(t)$ is the raw run-to-failure targets. Third, the raw run-to-failure targets are scaled according to the R_i value obtained from Table 2 for each engine:

$$DS_i(t) = \frac{R_i \cdot (Y_i(t) - Y_i(T_t))}{Y_i(t_1) - Y_i(T_t)} \tag{17}$$

where $Y_i(t)$ is the current raw run-to-failure target, $Y_i(T_t)$ is the last raw run-to-failure target, and $Y_i(t_1)$ is the first raw run-to-failure target. Finally, polynomial regression is performed on $DS_i(t)$ in order to remove noise. It should be noted that the polynomial regression used in this study performs a QR decomposition of the underlying Vandermonde matrix and the degree of the polynomial is 2. Figure 4 compares the raw DS targets and DS targets with polynomial regression.

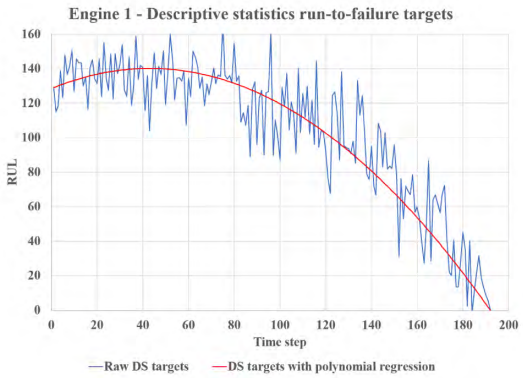


FIGURE 4. Comparison between raw DS targets and DS targets with polynomial regression for engine 1.

Algorithm 2 Algorithm for Constructing a Smooth Version of the Anomaly Score Function for Each Engine i

Input: $ASF_i(t), w_s, T_t$
Output: $ASF_i(t)_s$
  $w_s \leftarrow T_t$ / 1
  Creating one sliding window SW of length w_s which slides across $ASF_i(t)$ for each time step t.
  for $t := 0$ to T_t **do**
    $SW \leftarrow ASF_i(t)$
    $SW_{sum} \leftarrow 0$
    for $s := 0$ to w_s **do**
      $SW_{sum} += SW(s)$
    end for
    $ASF_i(t)_s \leftarrow \frac{SW_{sum}}{w_s}$
  end for
  return $ASF_i(t)_s$

3) ANOMALY SCORE FUNCTION

First, the raw ASF for each engine $ASF_i(t)_r$ is scaled according to the R_i value obtained from Table 2 for each engine:

$$ASF_i(t) = \frac{R_i \cdot (ASF_i(t)_r - ASF_i(T_t)_r)}{ASF_i(t_1)_r - ASF_i(T_t)_r} \tag{18}$$

where $ASF_i(t)_r$ is the current raw run-to-failure target, $ASF_i(T_t)_r$ is the last raw run-to-failure target, and $ASF_i(t_1)_r$ is the first raw run-to-failure target. Finally, in order to remove noise and make a smooth version, an additional sliding window SW of length $w_s = T_t/1$ is created. This sliding window slides across $ASF_i(t)$ for each time step t. Algorithm 2 shows the procedure on how to construct the smooth anomaly score function $ASF_i(t)_s$ for engine i. Figure 5 compares the raw ASF targets and the smooth ASF targets.

4) SELECTED DATA-DRIVEN LABELING APPROACHES

In the following experiments, the PwL, the DS with polynomial regression, and the smooth ASF targets will be

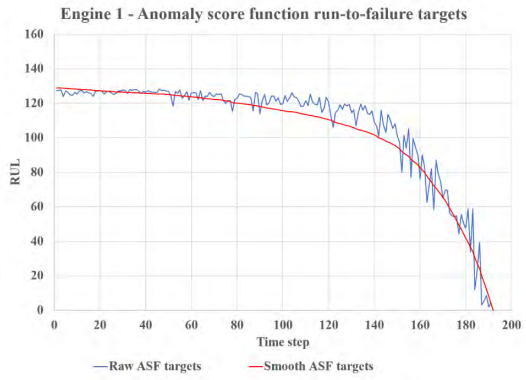


FIGURE 5. Comparison between raw ASF targets and smooth ASF targets for engine 1.

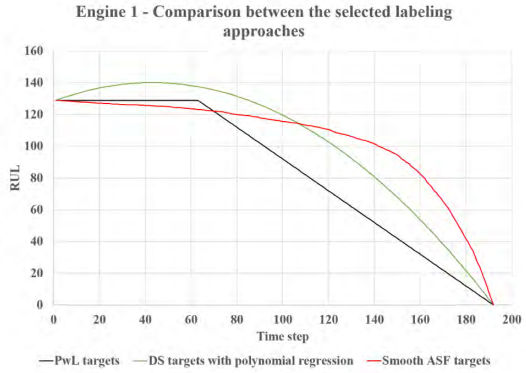


FIGURE 6. Comparison between the selected data-driven labeling approaches for engine 1.

used as supervised run-to-failure training targets for subset FD001. Figure 6 compares the selected data-driven labeling approaches.

E. DATA AUGMENTATION AND NORMALIZATION

Each input measurement x_n in the training set is normalized with zero mean and unit variance (z-score) normalization:

$$\hat{x}_n = \frac{x_n - \mu}{\sigma} \tag{19}$$

where μ and σ is the mean and the corresponding standard deviation of the population, respectively. Then, the normalization statistics obtained from the training set are applied to both the cross-validation set and the test set. Additionally, to reduce overfitting, random white Gaussian noise, g, is added to each $\hat{x}_n$ in each engine in the training set. P_{signal} and P_{noise} are the average power of the signal and the noise, respectively, and defined as follows:

$$P_{signal} = \frac{1}{T_t}\sum_{t=1}^{T_t}\left(\sqrt{\frac{1}{n}\left(\hat{x}_1^2 + \cdots + \hat{x}_n^2\right)}\right)_t \tag{20}$$

$$P_{noise} = \frac{1}{T_t}\sum_{t=1}^{T_t}\left(\sqrt{\frac{1}{n}\left((\hat{x}_1+g)^2 + \ldots + (\hat{x}_n+g)^2\right)}\right)_t \tag{21}$$

where T_t is the total time step length of each engine and n is the number of input features. Then, the signal-to-noise-ratio (SNR) can be defined as:

$$SNR(\%) = \frac{P_{signal}}{P_{noise}} \cdot 100 \tag{22}$$

In all experiments, 95% SNR is applied to the training set.

F. NETWORK CONFIGURATION AND TRAINING

Deep networks introduce several hyper-parameters, which are both challenging and time-consuming to optimize in the training procedure. Additionally, the proposed deep network structure requires different values of hyper-parameters for each labeling approach in order to perform with the highest RUL prediction accuracy possible. Thus, the proposed GA approach in [5] will also be used in this study in order to optimize the hyper-parameters for the networks trained on the three labeling approaches in an efficient manner.

The GA is a metaheuristic inspired by the natural selection process [29]. It is an effective algorithm for finding a near-optimal solution in a big search space, in this case, a big search space of hyper-parameters. However, in order to slightly reduce the search space, the networks will use some joint-hyper parameters which previously have shown great results on subset FD001 [4], [5]. Stochastic gradient descent (SGD) is the selected optimization algorithm and adaptive moment estimation (Adam) is the learning rate method [30]. To better preserve the low-level temporal features obtained from the 1D CNN layer, the learning rate in L1 is $l_r = 5 \cdot 10^{-5}$, while the learning rate in the remaining layers is $l_r = 1 \cdot 10^{-5}$. Xavier weight initialization [31] is applied to all layers. The rectified linear unit activation function [32] is used in both 1D CNN and FNN layers. However, in the LSTM layers, the tanh activation function is used in order to push the input and output values between -1 and 1. The mini-batch size is five engines, as previously optimized in [5]. The selected joint hyper-parameters are summarized in Table 3.

Table 4 shows the hyper-parameters which the GA approach optimized for each of the three networks. n is the number of hidden units in each layer, k_h is the kernel height in L1, and p is the dropout retaining probability of each unit in L5. A p value of 1.0 is functionally equivalent to zero dropout, namely, 100% probability of retaining each hidden unit. First, the GA approach selects random values of each hyper-parameter. One such set of random hyper-parameters is called an individual and a set of individuals is called a population. Each individual in the population is trained on the training set and evaluated on the cross-validation set. The $RMSE_{hz}$, equation 12, is the selected objective function. To prevent overfitting, early stopping is applied to monitor the

TABLE 3. Joint hyper-parameters.

Hyper-parameter	Method/value
Optimization algorithm	SGD
l_r method	Adam
l_r L1	$5 \cdot 10^{-5}$
l_r remaining layers	$1 \cdot 10^{-5}$
Weight initialization	Xavier
Activation function	ReLU (tanh in LSTM)
Mini-batch	5 engines

TABLE 4. Selected hyper-parameters in the GA approach.

Hyper-parameter	Values
n L1	32, 48, 64
n L2	128, 192, 256
n L3	64, 96, 128
n L4	64, 96, 128
n L5	16, 32, 48
k_h L1	4, 6, 8, 10
p L5	0.5, 0.6, 0.7, 0.8
$l2$ regularization	$1 \cdot 10^{-4}$, $1 \cdot 10^{-5}$, $1 \cdot 10^{-6}$

TABLE 5. Parameters of the GA approach.

Parameter	Value
Population size	30
Nr of Elite	1
Mutation Rate	0.5
Mutation Gain	0.3
Evolution iterations	4

performance during the training process of each individual. If the number of epochs with no reduction on $RMSE_{hz}$ on the cross-validation set exceeds four, the training process is terminated. Then, the network, in the epoch with the lowest $RMSE_{hz}$, is saved.

To limit the time consumed during the optimization process, the population size is restricted to 30 individuals. The best individual from the population is then kept and used as the parent for the next generation of hyper-parameters. Additionally, some random mutation is performed after the crossover for increasing the exploration of the algorithm. The population is evolved four times. This results in an average training time of 13.33 hours for each labeling approach, where each individual trained for 80 epochs on average with an average training time per epoch of 5 seconds. The parameters of the GA approach are shown in Table 5. In the end, the top five GA individuals for each labeling approach are evaluated on the test set where both *RMSE* and S are calculated. The GA individuals with the best result on the test set for each labeling approach are shown in Table 6 and the corresponding RUL prediction accuracy are shown in Table 7.

V. EXPERIMENTAL RESULTS AND DISCUSSIONS

The aim of this paper is to make a thorough comparison of three different data-driven labeling approaches for RUL predictions. The degradation significance within each of the constructed run-to-failure targets is extremely important for the RUL prediction performance of the proposed deep network structure. First, the GA optimized networks, as seen in Table 6, for the three labeling approaches are compared with three different performance evaluations on both the training set and the cross-validation set. Finally, the network with the highest RUL prediction accuracy on the test set is compared to the most robust results in the literature.

A. COMPARISON BETWEEN THE DATA-DRIVEN LABELING APPROACHES

The $RMSE_{hz}$ accuracy is considered an important performance indicator since it evaluates how accurately the networks are able to model the true overall degradation process in both the training set and cross-validation set. In addition, high $RMSE_{hz}$ accuracy is critical in order to achieve reliable confidence intervals for the corresponding RUL prediction in real-life PHM applications. As shown in Table 7, the network trained on PwL targets outperforms both the networks trained on DS and ASF targets with respect to the $RMSE_{hz}$ accuracy.

Both the *RMSE* and S accuracy are important performance indicators since high and reliable RUL prediction accuracy at the very end of the engines lifetime have great significance for real-life PHM applications. Thus, *RMSE* and S are only calculated at the last time step for each engine. It should be noted that both *RMSE* and S is the overall accuracy of all engines. In other words, the overall accuracy of 80 engines in the training set, 20 engines in the cross-validation set, and 100 engines in the test set. Additionally, to prevent overfitting, both dropout and random white Gaussian noise will reduce the accuracy on the training set compared to the accuracy on the cross-validation set. As shown in Table 7, the networks trained on PwL and DS targets perform with satisfactory *RMSE* and S accuracy. The network trained on ASF targets, however, performs with unacceptable *RMSE* and S accuracy. This is mainly because the run-to-failure targets decrease with increasing acceleration until failure. Thus, the network struggles to predict the failure ASF target for each engine, that is, when RUL $=$ 0 in both the training set and the cross-validation set. This also indicates that the predicted ASF targets are prone to late RUL predictions, namely, when $RUL_{predicted} - RUL_{true} > 0$. This reflects the extremely low S accuracy. Late RUL predictions could cause serious system failures in real-life PHM applications as maintenance operations will be scheduled too late.

In Figure 7, engines 2, 21, 52, and 70 in the cross-validation set are randomly selected for comparison. As previously mentioned, all three labeling approaches utilize an optimized R_i value for each engine. The high variance in R_i between engines in a mini-batch makes it difficult for the networks to predict the run-to-failure targets when the engines are operating in normal condition. Additionally, each engine in a mini-batch has different T_t. Thus, the shorter engines are padded with zeros such that all T_t are equal. Accordingly,

TABLE 6. GA individuals.

Labeling approach	Layer index	DL technique	nIn	nOut	Params	Total params	k_h L1	p L5	$l2$ regularization
PwL	1	1D CNN	24	32	4640	516,289	6	0.8	$1 \cdot 10^{-4}$
	2	LSTM	32	256	295,936				
	3	LSTM	256	128	197,120				
	4	FNN	128	128	16,512				
	5	FNN	128	16	2064				
	6	Output	16	1	17				
DS	1	1D CNN	24	32	4640	397,313	6	0.8	$1 \cdot 10^{-5}$
	2	LSTM	32	256	295,936				
	3	LSTM	256	64	82,176				
	4	FNN	64	128	8320				
	5	FNN	128	48	6192				
	6	Output	48	1	49				
ASF	1	1D CNN	24	64	6208	487,041	4	0.5	$1 \cdot 10^{-5}$
	2	LSTM	64	256	328,704				
	3	LSTM	256	96	135,552				
	4	FNN	96	128	12,416				
	5	FNN	128	32	4128				
	6	Output	32	1	33				

TABLE 7. The RUL prediction accuracy on subset FD001 for the three data-driven labeling approaches.

Labeling approach	Data set	S	$RMSE$	$RMSE_{hz}$
PwL	Training set	215.32	12.63	20.20
	Cross-validation set	23.69	7.84	25.03
	Test set	185.54	12.08	–
DS	Training set	307.29	13.97	22.34
	Cross-validation set	22.19	7.51	28.76
	Test set	348.85	14.75	–
ASF	Training set	1250.76	23.29	22.35
	Cross-validation set	68.62	14.60	29.77
	Test set	5305.78	29.85	–

TABLE 8. *S* and *RMSE* comparison with the literature on the test set of subset FD001.

Author & Refs.	Year	Approach	S	$RMSE$
Ramasso [33]	2014	RULCLIPPER	216	13.27
Malhotra et al. [16]	2016	LSTM-ED	256	12.81
Zheng et al. [11]	2017	LSTM + FNN	338	16.14
Zhang et al. [3]	2017	MODBNE	334	15.04
Yoon et al. [15]	2017	VAE + RNN	419	14.80
Li et al. [4]	2018	CNN + FNN	274	12.61
Ellefsen et al. [5]	2019	RBM + LSTM + FNN	231	12.56
Ellefsen et al.	2019	1D CNN + LSTM + FNN	**186**	**12.08**

mask arrays are used during the training process in order not to include the padded zeros in the performance evaluations. These masking arrays consist of the same value for each engine. The values are 82.6, 88.2, and 95.5, for the networks trained on PwL, DS, and ASF targets, respectively. Each network starts to predict based on its masking array value so that they do not start predicting on zero for each engine. Thus, this predicting approach is not optimal for the engines that are utilizing a R_i value either lower or higher than the masking array value. This is illustrated in Figure 7.

Nevertheless, the optimized R_i values are based on the degradation process rather than the number of time steps. Hence, the network trained on PwL targets predicts $RMSE_{hz}$, $RMSE$, and S with high accuracy after the predicted fault time step, that is, in the faulty degradation data of the engines lifetime. Thus, the optimized R_i values enable this network to generalize well on data never seen before, namely, the test set. Based on the superior results on the test set, the PwL degradation model is able to construct the most reliable run-to-failure targets for RUL predictions. PwL targets are also highly suitable if the RUL is to be considered as a time-based index, e.g., if the RUL decreases by one and the time step increases by one. This could be highly relevant for real-life PHM applications.

B. COMPARISON WITH THE LITERATURE

The network trained on PwL targets with optimized R_i values was able to generalize well, and hence, performed the highest RUL prediction accuracy on the test set. Thus, this network is compared with the literature. The authors have tried to include the most robust and recent results for comparison. That's why the well-known RULCLIPPER is also included. The RULCLIPPER does not utilize any DL techniques to make RUL predictions. Instead, it predicts the RUL based on imprecise health indicators modeled by planar polygons and similarity-based reasoning [33].

In Table 6, the selected studies are arranged in descending order based on the year they are published. As opposed to [33], the remaining studies utilize prognostics algorithms based on DL techniques to predict the RUL. However, most of these studies do not incorporate diagnostics information since the algorithms are trained on PwL run-to-failure targets with the same R_i value for all engines. On the other hand, the proposed deep network in this study is trained on PwL run-to-failure targets with optimized R_i values for each engine. Thus, the network takes into account the diagnostics aspect before making any RUL predictions. The high generalization towards the test set indicates that the optimized R_i values enable the network to model the true degradation process within subset FD001. To the best of the authors' knowledge, the proposed deep network, when trained on PwL run-to-failure targets with optimized R_i values, provides higher RUL prediction accuracy on subset FD001 than any in the literature.

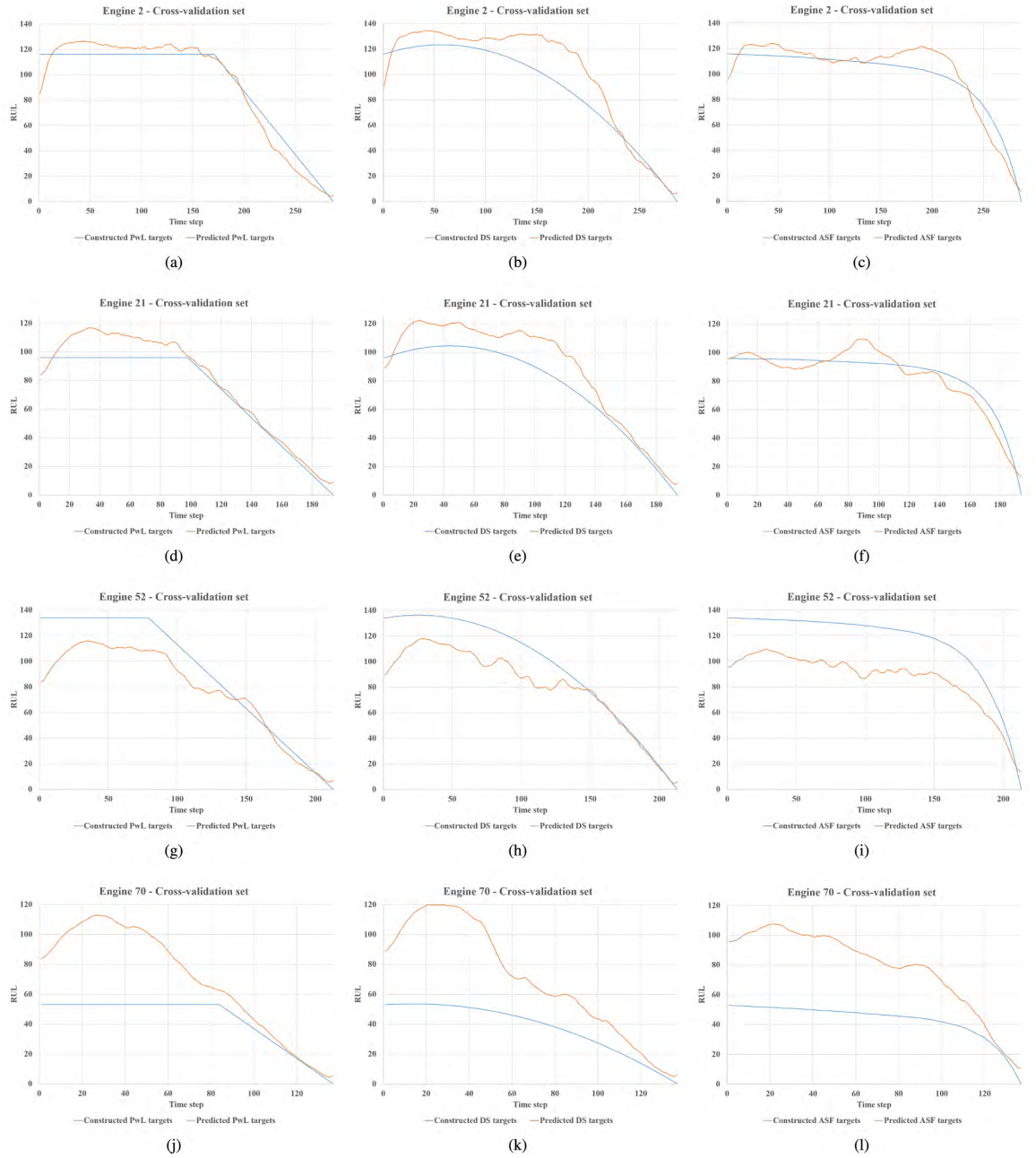


FIGURE 7. Cross-validation set comparison. (a) Engine 2 - PwL targets. (b) Engine 2 - DS targets. (c) Engine 2 - ASF targets. (d) Engine 21 - PwL targets. (e) Engine 21 - DS targets. (f) Engine 21 - ASF targets. (g) Engine 52 - PwL targets. (h) Engine 52 - DS targets. (i) Engine 52 - ASF targets. (j) Engine 70 - PwL targets. (k) Engine 70 - DS targets. (l) Engine 70 - ASF targets.

VI. CONCLUSION AND FUTURE WORK

This paper has compared three different data-driven labeling approaches for constructing run-to-failure targets. Additionally, a deep network structure has been proposed for RUL predictions. The experiments are performed on subset FD001 in the publicly available C-MAPSS data set. Most research studies that aim to predict the RUL based on DL approaches are still using the PwL degradation model to construct run-to-failure targets. This model assumes a constant R_i value that only depends on time to model normal

operating conditions. Hence, it neglects the entire diagnostics aspect. As illustrated in this study, any supervised prognostics algorithm should consider the diagnostics aspect before making any RUL predictions to achieve higher and more reliable accuracy. Thus, an unsupervised reconstruction-based fault detection algorithm has been used in this study to predict the fault time step for each engine. Then, an optimized R_i value for each engine was obtained. These R_i values were then used in the construction process of PwL, DS, and ASF run-to-failure targets. Finally, the proposed deep network structure was trained on the three different constructed run-to-failure targets. Additionally, a GA approach was used to tune the search space of hyper-parameters.

The network trained on PwL run-to-failure targets with optimized R_i values outperformed both the networks trained on DS and ASF run-to-failure targets with respect to RUL predictions. Additionally, this network outperformed the most robust results in the literature. The optimized R_i values are based on the individual degradation process in each engine. Hence, the network predicts $RMSE_{hz}$, $RMSE$, and S with high accuracy in the faulty degradation data of the engine's lifetime. The optimized R_i values enable the network to generalize well on data never seen before. The strong generalization indicates that the network is able to model the true degradation processes within the data set before making any RUL predictions. In other words, the diagnostics aspect is incorporated.

In this work, it was also discovered that the high variance in R_i between engines in a mini-batch made it difficult for the networks to predict the run-to-failure targets when the engines were operating in normal condition. To solve this issue we propose the following. First, a_{Th} can be further optimized in a more generic way for each engine. Second, the utilization of bigger (more parameters) and possibly deeper (more layers) networks. Finally, more training data with more engines with similar degradation processes, namely, with similar R_i values, would be favorable. Future work will address these issues.

Subset FD001 only contains one fault mode and one operating condition. If, however, several operating conditions were introduced in the data set, the unsupervised reconstruction-based fault detection algorithm could face some problems since the sensor measurements might differ strongly between different time steps with different operating conditions. This issue will also be explored in future work.

ACKNOWLEDGMENT

The authors would like to thank Digital Twins For Vessel Life Cycle Service (DigiTwin).

REFERENCES

[1] P. W. Kalgren, C. S. Byington, M. J. Roemer, and M. J. Watson, "Defining PHM, A lexical evolution of maintenance and logistics," in *Proc. IEEE Autotestcon*, Sep. 2006, pp. 353–358.

[2] A. L. Ellefsen, V. Æsøy, S. Ushakov, and H. Zhang, "A comprehensive survey of prognostics and health management based on deep learning for autonomous ships," *IEEE Trans. Rel.*, vol. 68, no. 2, pp. 720–740, 2019. doi: 10.1109/TR.2019.2907402.

[3] C. Zhang, P. Lim, A. K. Qin, and K. C. Tan, "Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2306–2318, Oct. 2017.

[4] X. Li, Q. Ding, and J.-Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Rel. Eng. Syst. Saf.*, vol. 172, pp. 1–11, Apr. 2018.

[5] 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," *Rel. Eng. Syst. Saf.*, vol. 183, pp. 240–251, Mar. 2019.

[6] A. Saxena and K. Goebel. Turbofan engine degradation simulation data set. NASA Ames Prognostics Data Repository, NASA Ames Research Center, Moffett Field, CA, USA. [Online]. Available: https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/

[7] F. O. Heimes, "Recurrent neural networks for remaining useful life estimation," in *Proc. Int. Conf. Prognostics Health Manage. (PHM)*, Oct. 2008, pp. 1–6.

[8] A. L. Ellefsen, E. Bjørlykhaug, V. Æsøy, and H. Zhang, "An unsupervised reconstruction-based fault detection algorithm for maritime components," *IEEE Access*, vol. 7, pp. 16101–16109, 2019.

[9] K. Le Son, A. Barros, M. Fouladirad, E. Levrat, and B. Iung, "Remaining useful life estimation based on probabilistic model," in *Proc. 17th ISSAT Int. Conf. Rel. Qual. Design*, 2011, pp. 10–25.

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

[11] S. Zheng, K. Ristovski, A. Farahat, and C. Gupta, "Long short-term memory network for remaining useful life estimation," in *Proc. IEEE Int. Conf. Prognostics Health Manage. (ICPHM)*, Jun. 2017, pp. 88–95.

[12] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates, 2012, pp. 1097–1105.

[14] Y. Taigman, M. Yang, M. A. Ranzato, and L. Wolf, "DeepFace: Closing the gap to human-level performance in face verification," in *Proc. IEEE Conf. CVPR*, Jun. 2014, pp. 1701–1708.

[15] 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," *CoRR*, vol. abs/1709.00845, pp. 1–9, Sep. 2017.

[16] 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," in *Proc. Workshop Mach. Learn. Prognostic Health Manage.*, 2016, pp. 1–10.

[17] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural Networks: Tricks of the Trade*. Berlin, Germany: Springer, 2012, pp. 9–48.

[18] I. Goodfellow, Y. Bengio, A. Courville, and F. Bach, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[19] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[20] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," in *Proc. 9th Int. Conf. Artif. Neural Netw. (ICANN)*, vol. 2, Sep. 1999, pp. 850–855.

[21] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Netw.*, vol. 18, no. 5, pp. 602–610, 2005.

[22] F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count," in *Proc. IEEE-INNS-ENNS Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 3, Jul. 2000, pp. 189–194.

[23] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[25] Apache Software Foundation License 2.0. (2018). *Eclipse Deeplearning4j Development Team, Deeplearning4j: Open-Source Distributed Deep Learning for the JVM*. [Online]. Available: http://deeplearning4j.org

[26] S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, and E. Shelhamer, "cuDNN: Efficient primitives for deep learning," 2014, *arXiv:1410.0759*. [Online]. Available: https://arxiv.org/abs/1410.0759

[27] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *Proc. Int. Conf. Prognostics Health Manage.*, Oct. 2008, pp. 1–9.
[28] T. Wang, J. Yu, D. Siegel, and J. Lee, "A similarity-based prognostics approach for remaining useful life estimation of engineered systems," in *Proc. Int. Conf. Prognostics Health Manage. (PHM)*, Oct. 2008, pp. 1–6.
[29] V. Roberge, M. Tarbouchi, and G. Labontè, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 132–141, Feb. 2013.
[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980
[31] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.
[32] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," *J. Mach. Learn. Res.*, vol. 15, no. 4, pp. 315–323, 2011.
[33] E. Ramasso, "Investigating computational geometry for failure prognostics," *Int. J. Prognostics Health Manage.*, vol. 5, no. 1, p. 005, 2014.

ANDRÉ LISTOU ELLEFSEN received the master's degree in subsea technology from the Norwegian University of Science and Technology (NTNU), Trondheim, Norway, in 2016. He is currently pursuing the Ph.D. degree with NTNU, Ålesund, Norway, as part of the Mechatronics Laboratory, Department of Ocean Operations and Civil Engineering. His current research interests include artificial intelligence, deep learning, decision support, predictive maintenance, prognostics and health management, and digital twins.

SERGEY USHAKOV received the Ph.D. degree from the Department of Marine Technology, Norwegian University of Science and Technology, in 2012 with a focus on the measurement and characterization of particulate matter emissions from marine diesel engines, where he rejoined, in 2016, as a Professor in marine machinery. For several years, he was with MARINTEK (currently SINTEF Ocean) within the fields of marine diesel engine emission characterization and emission reduction technologies covering both volatile and non-volatile exhaust emissions. During this work, he was involved in a number of bigger and smaller research projects, where accumulated substantial experience with experimental work both in laboratory and on board of different vessels. The current research focus is environmentally friendly shipping as well as the improvement of marine diesel engines' efficiency, especially emphasizing the experimental part of this work.

VILMAR ÆSØY graduated from NTNU, in 1989, and continued his research on natural gas fueled marine engines at NTNU/MARINTEK, in 1997. In 1996, he received the Ph.D. degree for his research on natural gas ignition and combustion through experimental investigations and numerical simulations. From 1989 to 1997, he was involved in several large R&D projects developing gas fueled engines and fuel injection systems for the diesel engine manufacturers, Wärtsilä and Bergen Diesel (Roll-Royce). From 1998 to 2002, he was an R&D Manager for Rolls-Royce Marine Deck Machinery. Since 2002, he has been employed in teaching with the Aalesund University College, developing and teaching courses in marine product and systems design on bachelor's and master's level. In 2010, he received the Green Ship Machinery Professorship. His special research interest is within the field of energy and environmental technology, with a focus on combustion engines and the need for more environmental friendly and energy-efficient systems.

HOUXIANG ZHANG (M'04–SM'12) received the Ph.D. degree in mechanical and electronic engineering from the Robotics Institute, Beihang University, in 2003, and the Habilitation degree in informatics from the University of Hamburg, Germany, in 2011. Since 2004, he has been a Postdoctoral Fellow and a Senior Researcher with the Department of Informatics, Faculty of Mathematics, Informatics and Natural Sciences, Institute of Technical Aspects of Multimodal Systems (TAMS), University of Hamburg. He joined Norwegian University of Science and Technology, Norway, in 2011, where he is currently a Professor in mechatronics. He has involved in two main research areas: 1) biological robots and modular robotics, especially on biological locomotion control, and 2) virtual prototyping in demanding marine operation. He has applied for and coordinated more than 20 projects supported by the Norwegian Research Council (NFR), the German Research Council (DFG), and the industry. In these areas, he has published over 160 journal and conference papers as an author or a coauthor. He has received four best paper awards and four finalist awards for the best conference paper at the International conference on Robotics and Automation.

E

Paper V

Automatic Fault Detection for Marine Diesel Engine Degradation in Autonomous Ferry Crossing Operation

André Listou Ellefsen, Xu Cheng, Finn Tore Holmeset
Vilmar Æsøy and Houxiang Zhang
Department of Ocean Operations and Civil Engineering
as part of the Mechatronics Laboratory
Norwegian University of Science and Technology
Aalesund, 6009, Norway
{*andre.ellefsen, xu.cheng, fiho, vilmar.aesoy, hozh*}*@ntnu.no*

Sergey Ushakov
Department of Marine Technology
Norwegian University of Science and Technology
Trondheim, 7491, Norway
sergey.ushakov@ntnu.no

***Abstract*—The maritime industry generally anticipates having semi-autonomous ferries in commercial use on the west coast of Norway by the end of this decade. In order to schedule maintenance operations of critical components in a secure and cost-effective manner, a reliable prognostics and health management system is essential during autonomous operations. Any remaining useful life prediction obtained from such system should depend on an automatic fault detection algorithm. In this study, an unsupervised reconstruction-based fault detection algorithm is used to predict faults automatically in a simulated autonomous ferry crossing operation. The benefits of the algorithm are confirmed on data sets of real-operational data from a marine diesel engine collected from a hybrid power lab. During the ferry crossing operation, the engine is subjected to drastic changes in operational loads. This increases the difficulty of the algorithm to detect faults with high accuracy. Thus, to support the algorithm, three different feature selection processes on the input data is compared. The results suggest that the algorithm achieves the highest prediction accuracy when the input data is subjected to feature selection based on sensitivity analysis.**

***Index Terms*—Automatic fault detection, feature selection, marine diesel engine, prognostics and health management, variational autoencoder**

I. Introduction

Only five years ago, most people considered autonomous and semi-autonomous ships as a futuristic fantasy [1]. Today, however, this assumption has changed dramatically since inland semi-autonomous ferries will most definitely be in commercial use on the west coast of Norway by the end of this decade [2]. These ferries are intended to navigate entirely by themselves a short distance across a river or a fjord. Thus, the crew members will ideally carry out duties other than maintaining, operating, and navigating the vessels. Additionally, securing regulatory permission, support from the industry, and public approval for semi-autonomous ferries requires evidence they are at least as safe as traditional ferries [3].

Ideally, semi-autonomous ferries will transfer real-time diagnostics and prognostics information to a control center onshore to conduct analysis and schedule maintenance operations of critical systems, components, and sub-components. One of the most critical components is the marine diesel engine as it has a leading position in both propulsion and power generation [4]. Even though the navigation mission for the ferry is rather simple in theory, the marine diesel engine will be subjected to changing environmental conditions and various operational loads. Consequently, faults and failures could occur in a totally random pattern [5]. Hence, in a maintenance perspective, a prognostics and health management (PHM) system, which both include automatic fault detection and associated remaining useful life (RUL) predictions, is crucial in autonomous operations. When the RUL is predicted, the maintenance operation can be scheduled to the next appropriate port of call for the ferry [6]. Nevertheless, the RUL prediction is the available time prior to operational failure after a fault is detected within the engine. Thus, any RUL prediction should depend on an intelligent and reliable fault detection algorithm.

During the last two years, the growth of intelligent fault detection algorithms has increased drastically. Usually, the algorithms have dependent on a supervised classifier [7], [8]. In other words, the algorithms demand fault labels in the training procedure. However, due to a general lack of fault labels for critical components in the maritime industry [9], an appropriate fault detection algorithm should not depend on a supervised classifier. An alternative approach is the utilization of unsupervised reconstruction-based fault detection algorithms [10], [11]. Usually, these algorithms train a Variational Autoencoder (VAE), in an unsupervised practice, to reconstruct normal operation data. In this way, the VAE will provide a greater reconstruction error on unexpected patterns in faulty degradation data. Finally, the reconstruction error is used as an anomaly score function (ASF) before an algorithm is applied to detect faults automatically. However, in semi-autonomous ferries, the sensor measurements might differ strongly between different engine operational loads. This increases the difficulty for the VAE to construct an accurate

ASF. Thus, the input data should be subjected to a feature selection process in order to support the VAE in the demanding reconstruction process.

This paper investigates automatic fault detection for marine diesel engine degradation in a simulated autonomous ferry crossing operation. The unsupervised reconstruction-based fault detection algorithm proposed in [11], is also used in this study to predict faults automatically. The VAE is the selected reconstruction model. Two data sets of real-operational data from a marine diesel engine are used. The first data set is a simulated ferry crossing during normal operation, while the second data set is the exact same ferry crossing operation except a fault is introduced at an unknown time step. First, the VAE is trained on the normal operation data. Then, the VAE estimates an ASF by computing a reconstruction error at each time step in the second data set, namely, the faulty degradation data. In the end, the algorithm detects a fault automatically by predicting the time step with the highest acceleration in the ASF. In order to examine the need for a feature selection process to support the VAE reconstruction process, both the normal operation data and the faulty degradation data are used to create three different input dimension scenarios: all input features, feature selection based on human domain knowledge (HDK), and feature selection based on sensitivity analysis (SA). In all three scenarios, an individual reconstruction model is used due to different input dimensions.

Our on-going project intends to develop an intelligent PHM system to provide real-time decision automation for autonomous maritime operations. Currently, the project mainly consists of two parts. The first part is the development of a step-wise feature selection approach to support both diagnostics and prognostics algorithms. The second part, on the other hand, is devoted to the development of both automatic fault detection algorithms and RUL prediction algorithms. Nevertheless, in this paper, we are only focusing on automatic fault detection. This study's principal contributions are as follows:

- Three input dimension scenarios on real-operational marine diesel engine degradation data are compared.
- Feature selection processes drastically improve the accuracy of unsupervised reconstruction-based fault detection algorithms.

The overall organization of the paper is as follows. Section II introduces the essential background on the VAE and unsupervised reconstruction models. The experimental procedure, results, and discussions are elaborated in section III. Section IV concludes and finishes the paper and presents objectives for future work.

II. Background

This section introduces the essential background on the VAE and the unsupervised reconstruction models.

A. Variational autoencoder

The VAE was developed by Kingma and Welling in 2013 and models the underlying probability distribution utilizing Bayesian inference [12]. The VAE includes an encoder function $z = q_{\theta_e}(z|x)$ and a decoder function $r = p_{\theta_d}(x|z)$. Thus, compared to the traditional autoencoder [13], the VAE improves generalization since the latent variables z are stochastic in nature. The VAE objective function is to maximize the variational lower bound J_{VAE} [14]:

$$J_{VAE}(\theta_e, \theta_d) = -D_{KL}\big(q_{\theta_e}(z|x)\,||\,p_{\theta_d}(z)\big) + E_{q_{\theta_e}(z|x)}[\log\, p_{\theta_d}(x|z)] \tag{1}$$

where D_{KL} is the Kullback-Leibeler divergence. The first expression is referred to the latent loss and measures how close z match the encoder function. The second expression is the reconstruction log-likelihood and referred to the generative loss. Nevertheless, the reconstruction error needs a Monte Carlo estimate of the expectation [12]. Since this estimate is not easily differentiable, a reparameterization scheme of z is used to collect the gradients of the decoder in order to use the back-propagation algorithm [15]. First, the reparameterization scheme applies a deterministic variable such that $z = \boldsymbol{\mu} + \boldsymbol{\sigma}\varepsilon, \varepsilon \sim \mathcal{N}(0,1)$ [12]. In this way, the encoder produces vectors of both means $\boldsymbol{\mu}$ and standard deviations $\boldsymbol{\sigma}$ rather than vectors of real values. Finally, these vectors are applied as the latent vector in the decoder. A Gaussian reconstruction distribution is normally utilized in the decoder for real-valued input data. The VAE can be stacked with many hidden layers in both the encoder and decoder depending on the dimensionality of the input data. It should be noted that unsupervised pre-training should be considered for very deep VAE structures.

B. Unsupervised reconstruction models

As similar to [11], the reconstruction models in this study are also configured with three hidden layers and corresponding hidden units ($h1$,$h2$,$h3$) in the encoder and three hidden layers with corresponding hidden units ($h3$,$h2$,$h1$) in the decoder. However, due to different input dimensions, the selection process of the hidden units is based on the following experience-based formula:

$$h1 = \mathbb{Z}\big(n \cdot 1.2\big) \qquad h2 = \mathbb{Z}\left(\frac{h1}{2}\right) \qquad h3 = \mathbb{Z}\left(\frac{h2}{2}\right)$$

where n is the number of input features in the specific scenario. Consider $\boldsymbol{x}_t = [x_1 \dots x_n]_t$ as the input vector of measurements at time step t. In order to train the reconstruction models in an unsupervised practice, $\boldsymbol{x}_t$ is also utilized as the target $\boldsymbol{y}_t$ for reconstruction at each t. To measure error calculations, each reconstruction model uses a fully connected output layer where the mean squared error (MSE) is the chosen loss function:

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

where n is the number of input features, $\hat{y}_i$ is the i_{th} predicted measurement and y_i is the i_{th} target measurement.

Fig. 1. The small marine diesel engine included in the hybrid power lab at the Department of Ocean Operations and Civil Engineering at the Norwegian University of Science and Technology in Aalesund.

III. Experimental study

In the ensuing experimental study, Microsoft Windows 10 is the operating system, Java 8 is the programming language, "deeplearning4j" (DL4J) version 1.0.0-beta3 [16] is the deep learning library and NVIDIA GeForce GTX 1060 6 GB is the graphics processing unit used. The reconstruction models are trained and evaluated on real-operational data from a marine diesel engine.

A. Data sets

A hybrid power lab, founded by the Department of Ocean Operations and Civil Engineering at the Norwegian University of Science and Technology in Aalesund, is used to collect the data sets. The lab consists of a small marine diesel engine with a generator, a marine battery system, a marine DC switchboard with necessary power converters, and a marine automation system to control the entire process. The power produced is fed back to the power grid in order to simulate load changes in the system. The marine diesel engine is shown in Figure 1.

During the data collection process, the engine is run by an operating profile that aims to simulate a real-life autonomous ferry crossing on the west coast of Norway. First, the ferry leaves shore in a safe and constant velocity. Then, the ferry increases its velocity until a suitable velocity is reached. This velocity is kept constant before the velocity decreases safely. Finally, the ferry breaks just before it docks. The total duration of the ferry crossing is 22 minutes and 40 seconds and the complete engine operating profile is shown in Figure 2.

The engine operating profile is run both when the normal operation data and the faulty degradation data are collected. Thus, the difference between the two data sets is that a fault is introduced at an unknown time step in the faulty degradation data. Hence, the main goal is to predict the time step where the fault occurs, namely, the fault time step f_t.

The engine has both a primary and a secondary water cooling system, where the secondary cools the primary. The primary cooling is controlled internally in the engine by a bi-metal thermostatic valve, which opens at 78 °C and fully open at 90 °C. The secondary cooling is controlled by a frequency operated fan circulating air through a heat exchanger. The fault introduced is a malfunction of the fan. This results in loss of cooling efficiency in the secondary cooling system. An alarm is triggered in the marine automation system when the cooling water temperature increases 85 °C.

TABLE I
Real-operational data sets collected from the marine diesel engine.

Data set	Time (seconds)	Frequency	Time steps
Normal operation data	1360	2 Hz	2720
Faulty degradation data	1360	2 Hz	2720

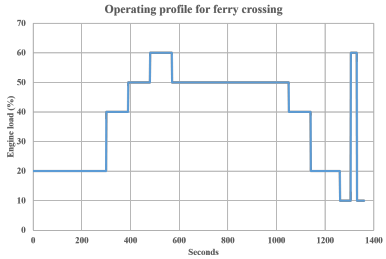


Fig. 2. Operating profile for a simulated autonomous ferry crossing.

Table I summarizes the two data sets collected. As seen in Figure 2, the engine load changes drastically throughout the ferry crossing operating profile. Thus, the sensor measurements differ strongly between the different engine loads. This affects the ability of the VAE to reconstruct an ASF with high degradation relevance. Therefore, in this study, both the normal operation data and the faulty degradation data are further used to create three different input dimension scenarios: all input features, feature selection based on HDK, and feature selection based on SA.

1) All input features: The raw data sets collected in this study includes 47 input features in total, e.g., operational loads, temperature, pressure, flow, and engine speed measurements. This scenario utilizes all 47 input features, and hence, neglects the degradation relevance for each input feature regarding the specific fault used in this study. Thus, in this scenario, the difficulty for the VAE to reconstruct an accurate ASF increases.

2) Feature selection based on human domain knowledge: In this scenario, valuable HDK is used to select degradation relevant input features concerning the specific fault. The goal of this selection process is to reduce the amount of noise in the reconstructed ASF, and hence, support the algorithm to predict f_t with higher accuracy. This selection process results in 22 input features.

3) Feature selection based on sensitivity analysis: This scenario is based on the first part of our on-going project. The step-wise feature selection approach is based on variance-based sensitivity analysis. In order to remove redundant information among the input features and reduce the computational complexity of variance-based sensitivity analysis, a Pearson

TABLE II
JOINT HYPER-PARAMETERS.

Hyper-parameter	Method/value
Optimization algorithm	Stochastic gradient descent
l_r method	Adaptive moment estimation [19]
l_r	$1 \cdot 10^{-4}$
$l2$ regularization	$1 \cdot 10^{-4}$
Weight initialization	Xavier [20]
Activation function	Rectified linear unit [21]

correlation analysis is conducted. Additionally, a surrogate model is adopted since conventional variance-based sensitivity analysis cannot be applied to the data sets directly [17], [18]. This selection process results in 12 input features.

B. Data normalization

Each input measurement x_n in the normal operation data is normalized with zero mean and unit variance normalization:

$$\hat{x}_n = \frac{x_n - \mu}{\sigma} \tag{3}$$

where μ and σ is the mean and the corresponding standard deviation of the normal operation data, respectively. Then, the normalization statistics obtained from the normal operation data are applied to the faulty degradation data.

C. Hyper-parameter configuration and training

The three reconstruction models are configured with joint hyper-parameters, as similar to [11]. Joint hyper-parameters are used in order to create reliable comparisons between the three input dimension scenarios. The selected hyper-parameters are summarized in Table II. An early stopping (ES) approach is used during the training process of each reconstruction model in order to reconstruct the normal operation data as accurately as possible. The total reconstruction error of all time steps in the normal operation data E_{nod} is monitored by the ES approach for each epoch:

$$E_{nod} = \sum_{t=1}^{T_{nod}} \left(\frac{1}{n} \sum_{i=1}^{n} ||\hat{y}_i - y_i||^2 \right)_t \tag{4}$$

where T_{nod} is the total number of time steps in the normal operation data and the second term is the MSE in Eq. 2. The training process is terminated if the number of epochs with no reduction on E_{nod} is greater than four. Finally, the reconstruction model, obtained from the epoch with the lowest E_{nod}, is used for validation on the faulty degradation data.

D. Fault prediction

Ellefsen et al. [11] used an unsupervised reconstruction-based fault detection algorithm for maritime components. Their proposed algorithm is also used in this work in order to predict f_t. First, the raw ASF is estimated by computing the MSE, Eq. 2, at each time step in the faulty degradation data. Normally, the raw ASF includes high amounts of noise. Thus, the algorithm generates three sliding windows of length w in order to smooth the ASF:

$$w = \frac{T_{fdd}}{p} \tag{5}$$

TABLE III
THE TRUE FAULT TIME STEP f_t COMPARED TO THE PREDICTED FAULT TIME STEP $\hat{f}_t$ ON THE FAULTY DEGRADATION DATA FOR EACH SCENARIO.

Scenario	n	f_t	p	w	$\hat{f}_t$
All input features	47	1979	60	45	2529
			70	39	2540
			80	34	2544
			90	30	2548
			100	27	2549
HDK	22	1979	60	45	2012
			70	39	1852
			80	34	1861
			90	30	1863
			100	27	1867
SA	12	1979	60	45	1994
			70	39	2004
			80	34	2000
			90	30	1863
			100	27	1867

where T_{fdd} is the total number of time steps in the faulty degradation data and p is a tune-able and application-dependent parameter. Next, the three sliding windows slide across the raw ASF for each time step, where a distance equivalent to w is applied between each sliding window. Then, in order to remove a certain amount of noise in the raw ASF, the average reconstruction error is calculated in the three windows. Thus, since p decides the length of w, it also decides the amount of smoothing performed on the ASF. Next, the velocity between windows 1 and 2 and between 2 and 3 are calculated. Finally, the acceleration between the two velocities is estimated. A comparison between the raw ASF and the smooth ASF for each scenario is shown in Figure 3.

According to [11], the maximum increase in sensor measurements deviations compared to typical sensor measurements in normal operation data is a clear symptom of a fault. Therefore, the maximum acceleration is used as the fault indicator since this point indicates increasing velocity, and hence, an accelerated increase in the ASF. The increasing velocity indicates that one or several feature measurements have begun to diverge from the normal operation data quickly. Thus, the algorithm detects the maximum acceleration and the corresponding fault time step $\hat{f}_t$. Please see [11], for a more comprehensive explanation of the algorithm.

E. Experimental results and discussions

Table III shows the predicted fault time step $\hat{f}_t$ for each scenario. Five different p values are used to examine the robustness of each reconstruction model. The lowest p value, however, is determined to 60 since any lower value might smooth the ASF too much, and hence, ignoring important degradation patterns. The true fault time step f_t in the faulty degradation data is determined based on the first time the cooling water temperature increases 85 °C. It should be noted that both f_t and $\hat{f}_t$ can be divided by two in order to be consistent with Figure 2.

As seen in Table III, the scenario utilizing all input features performs late $\hat{f}_t$ predictions for all p values. This scenario neglects the relevance of degradation for each input feature concerning the specific fault used in this study. Thus, as seen

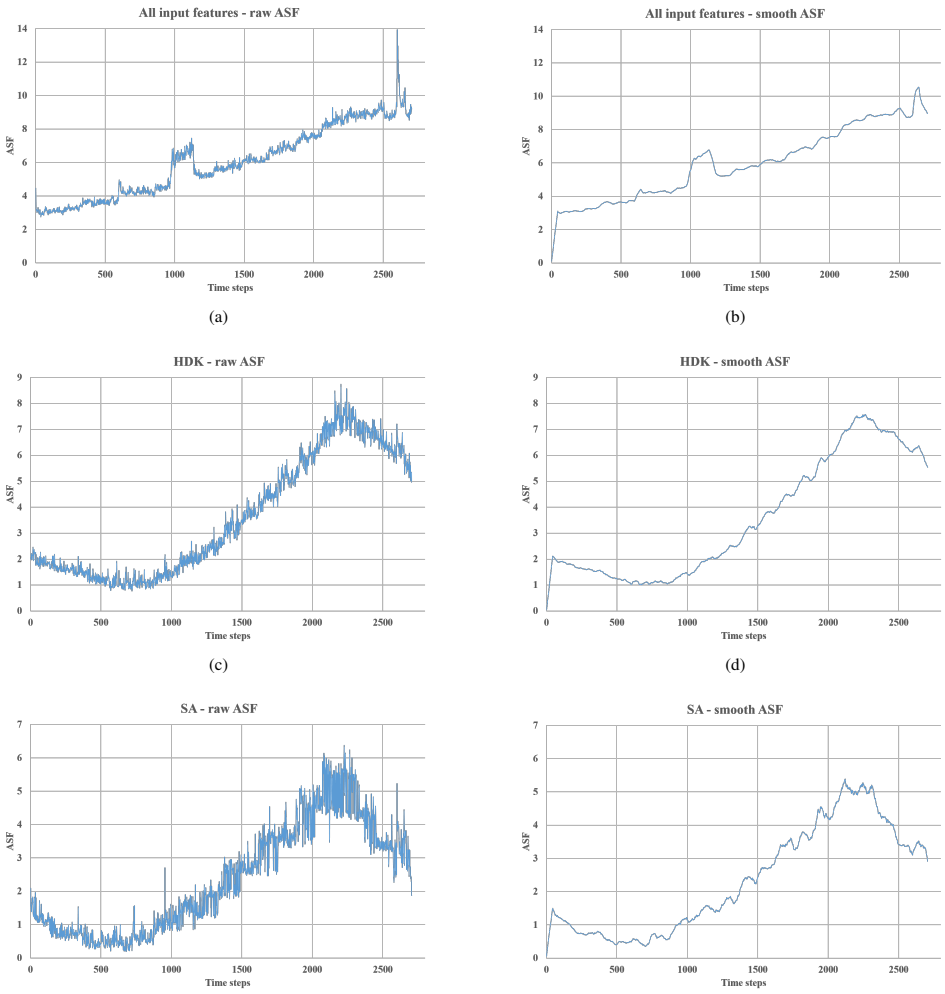


Fig. 3. ASF comparison between the three scenarios. p = 60 in the smooth ASF. (a) All input features - raw ASF. (b) All input features - smooth ASF. (c) HDK - raw ASF. (d) HDK - smooth ASF. (e) SA - raw ASF. (f) SA - smooth ASF.

in Figure 3, a spike occurs in the smooth ASF when the engine load increases rapidly in order for the ferry to break just before it docks. Hence, the algorithm detects the maximum acceleration in the breakpoint in front of the spike. The spike occurs since engine speed, redundant measurements on engine loads, and several battery measurements are included in the data sets. These features increase the difficulty for the VAE to construct an accurate ASF, especially when the engine load is above 50%. However, these features have no degradation relevance for the fault, and hence, they should be disregarded in order to remove noise for the fault detection algorithm. As opposed to the scenario utilizing all input features, the scenarios based on HDK and SA remove both irrelevant and redundant input features concerning the fault. Thus, these scenarios perform accurate $\hat{f}_t$ predictions, especially when p = 60.

The accuracy evaluations on the faulty degradation data in the three scenarios are shown in Table IV. The accuracy is defined as follows:

$$Accuracy\,(\%) = \left(1 - \frac{||\hat{f}_t - f_t||}{2720}\right) \cdot 100 \tag{6}$$

where 2720 is the total number of time steps in the faulty

TABLE IV
ACCURACY EVALUATION ON THE FAULTY DEGRADATION DATA FOR EACH SCENARIO.

p	Accuracy (%)		
	All input features	HDK	SA
60	79.78	98.79	99.45
70	79.38	95.33	99.08
80	79.23	95.66	99.23
90	79.08	95.74	95.74
100	79.04	95.88	95.88
Avg. Accuracy	79.30	96.28	**97.88**

degradation data. As seen in Table IV, both the HDK and SA scenario perform consistent accuracy above 95% for all p values. Nevertheless, the scenario based on SA performs the highest average accuracy.

IV. CONCLUSION AND FUTURE WORK

This paper has examined automatic fault detection for marine diesel engine degradation in a simulated autonomous ferry crossing operation. An unsupervised reconstruction-based fault detection algorithm has been used to predict faults automatically. The VAE is used as the reconstruction model. Two data sets of real-operational data have been collected from a hybrid power lab including a marine diesel engine. The first data set is a simulated ferry crossing during normal operation, while the second data set is the exact same ferry crossing except a fault is introduced at an unknown time step. First, the VAE is trained on the normal operation data. Then, the VAE estimates an ASF by computing a reconstruction error at each time step in the faulty degradation data. In the end, the algorithm detects a fault automatically by predicting the time step with the highest acceleration in the ASF. Although the navigation mission for the ferry is simple, the engine is subjected to drastic changes in operational loads during the simulated ferry crossing operation. This increases the difficulty of the algorithm to detect faults with high accuracy. Thus, to support the algorithm, three different feature selection processes on the input data have been compared.

The algorithm achieved an average accuracy of 97.88% when the input data were subjected to feature selection based on SA. SA removes both irrelevant and redundant input features concerning the specific fault used in this study. Thus, drastically improving the prediction accuracy of the algorithm. However, any feature selection process might remove input features which could be of relevance for other faults with different degradation nature. Hence, introducing several other faults in the hybrid power lab will be part of future work.

ACKNOWLEDGMENT

This work was supported by the Norwegian University of Science and Technology within the Department of Ocean Operations and Civil Engineering under project no. 90329106. The authors would like to thank Digital Twins For Vessel Life Cycle Service (DigiTwin) and the Research Council of Norway, grant no. 280703.

REFERENCES

[1] E. Jokioinen, "Remote and autonomous ships - the next steps: Introduction," *Rolls-Royce, Buckingham Gate, London: The Advanced Autonomous Waterborne Applications (AAWA)*, pp. 4–14, 2016.

[2] O. Levander, "Autonomous ships on the high seas," *IEEE Spectrum*, vol. 54, no. 2, pp. 26–31, February 2017.

[3] R. Jalonen, R. Tuominen, and M. Wahlström, "Remote and autonomous ships - the next steps: Safety and security," *Rolls-Royce, Buckingham Gate, London: The Advanced Autonomous Waterborne Applications (AAWA)*, pp. 56–73, 2016.

[4] J. A. P. Rubio, F. Vera-García, J. H. Grau, J. M. Cámara, and D. A. Hernandez, "Marine diesel engine failure simulator based on thermodynamic model," *Applied Thermal Engineering*, vol. 144, pp. 982–995, 2018.

[5] T. M. Allen, "Us navy analysis of submarine maintenance data and the development of age and reliability profiles," *U.S. Navy SUBMEPP, Kittery, ME, USA, Tech. Rep.*, 2001.

[6] A. L. Ellefsen, V. Æsøy, S. Ushakov, and H. Zhang, "A comprehensive survey of prognostics and health management based on deep learning for autonomous ships," *IEEE Transactions on Reliability*, pp. 1–21, 2019.

[7] C. Lu, Z.-Y. Wang, W.-L. Qin, and J. Ma, "Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification," *Signal Processing*, vol. 130, pp. 377–388, 2017.

[8] H. Liu, J. Zhou, Y. Zheng, W. Jiang, and Y. Zhang, "Fault diagnosis of rolling bearings with recurrent neural network-based autoencoders," *ISA transactions*, vol. 77, pp. 167–178, 2018.

[9] A. S. Zymaris, Ø. Å. Alnes, K. E. Knutsen, and N. M. Kakalis, "Towards a model-based condition assessment of complex marine machinery systems using systems engineering," in *PHM Europe, Bilbao, Spain*, 2016, Conference Proceedings.

[10] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1544–1551, 2018.

[11] A. L. Ellefsen, E. Bjørlykhaug, V. Æsøy, and H. Zhang, "An unsupervised reconstruction-based fault detection algorithm for maritime components," *IEEE Access*, pp. 16 101–16 109, 2019.

[12] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[13] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

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

[15] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 9–48.

[16] "Eclipse deeplearning4j development team, deeplearning4j: Open-source distributed deep learning for the jvm," *Apache Software Foundation License 2.0, http://deeplearning4j.org*, 2019.

[17] X. Cheng, S. Chen, C. Diao, M. Liu, G. Li, and H. Zhang, "Simplifying neural network based model for ship motion prediction: a comparative study of sensitivity analysis," in *ASME 2017 36th International Conference on Ocean, Offshore and Arctic Engineering*. American Society of Mechanical Engineers, 2017.

[18] X. Cheng, G. Li, R. Skulstad, S. Chen, H. P. Hildre, and H. Zhang, "A neural-network-based sensitivity analysis approach for data-driven modeling of ship motion," *IEEE Journal of Oceanic Engineering*, 2019.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[20] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[21] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, vol. 15, 2011, pp. 315–323.

F

Paper VI

Online Fault Detection in Autonomous Ferries: Using Fault-type Independent Spectral Anomaly Detection

André Listou Ellefsen, Peihua Han, Xu Cheng, *Student Member, IEEE,* Finn Tore Holmeset, Vilmar Æsøy, and Houxiang Zhang*, *Senior Member, IEEE*

***Abstract*—Enthusiasm for ship autonomy is flourishing in the maritime industry. In this context, data-driven Prognostics and Health Management (PHM) systems have emerged as the optimal way to improve operational reliability and system safety. However, further research is needed to enhance the essential actions relating to such a system. Fault detection is the first and most crucial action of any data-driven PHM system. In this study, we propose a fault-type independent spectral anomaly detection algorithm for marine diesel engine degradation in autonomous ferries. The benefits of the algorithm are verified on three fault-types where the nature of degradation differs. Both normal operation data and faulty degradation data have been collected from a marine diesel engine, using two different engine load profiles. These profiles aim to replicate real autonomous ferry crossing operations, environmental conditions the ferry may encounter. First, the data is subjected to a feature selection process to remove irrelevant and redundant features. Then, a multi-regime normalization method is performed on the data to merge the engine loads into one context. Finally, a variational autoencoder is trained to estimate velocity and acceleration calculations of the anomaly score. Generic and dynamic threshold limits are simultaneously established to detect the fault time step online. The algorithm achieved an accuracy of 97.66% in the final test when the acceleration was used as the fault detector. The results suggest that the algorithm is independent of fault-types with different nature of degradation related to the marine diesel engine.**

***Index Terms*—Autonomous ferry, marine diesel engine, multi-regime normalization, online fault detection, prognostics and health management**

I. Introduction

TODAY, ship autonomy is the most-sought research objective at the Norwegian University of Science and Technology in Aalesund [1], [2]. However, autonomous ships were considered to be a futuristic fantasy only six years ago [3]. Yet inland autonomous ferries carrying tiny crews primarily to make passengers feel safe will be in commercial use on the west coast of Norway in the very near future [4]. The industry, as well as academics, anticipate that these ferries will improve both safety and profitability [5]. Maintaining, operating, and navigating the vessels without crew involvement will necessitate the use of highly automated systems and belonging sensor equipment, and degradation of such systems during operation poses a serious threat to operations [6].

Prognostics and Health Management (PHM) is the area of research with the greatest promise to manage maintenance operations for zero-downtime performance of autonomous ferries [6]. A data-driven PHM system goes far beyond traditional maintenance approaches, such as reactive maintenance and preventive maintenance, currently in use onboard ships [7]. Such a system use algorithms built on sensor measurements to perform automatic fault detection, fault isolation, fault classification, and associated remaining useful life (RUL) predictions to devise an ideal maintenance schedule that eliminates failures [8]. Autonomous ferries will transfer real-time operational sensor data to a remote control center to conduct the essential actions of a data-driven PHM system (see Figure 1). Thus, it will be possible to schedule maintenance operations to the next appropriate port of call. The ideal maintenance schedule will considerably enhance operational availability and reliability and system safety.

Anomaly detection techniques aim to discover deviations from normal operation data. In a data-driven PHM viewpoint, such deviations are symptoms of incipient faults [9]. Fault detection is the first and most crucial action of any data-driven PHM system. It should be performed automatically to detect the fault time step in degradation data. Then, this time step can be used to construct both labels for fault classification and run-to-failure targets for RUL predictions. Interest in spectral anomaly detection techniques has increased recently. These techniques try to produce the lower dimensional embedding of the input data where anomalies and normal operation data are generally distinct [10]. The reconstruction error at each time step between the input data and its low dimensional reconstruction is then used as an anomaly score to detect anomalies [10]. The principal components analysis method is one of the best-known traditional spectral anomaly detection techniques [11]. However, deep neural networks (DNNs) have recently shown superior performance for this purpose [9]. DNNs allow dimension reduction through several hidden layers with non-linear transformations, and hence, obtain more

*Corresponding author.

André Listou Ellefsen, Peihua Han, Xu Cheng, Finn Tore Holmeset, Vilmar Æsøy, and Houxiang Zhang are with the Department of Ocean Operations and Civil Engineering, as part of the Mechatronics Laboratory, Norwegian University of Science and Technology, Aalesund, 6009, Norway, (e-mail: andre.ellefsen@ntnu.no; peihua.han@ntnu.no; xu.cheng@ntnu.no; fiho@ntnu.no; vilmar.aesoy@ntnu.no; hozh@ntnu.no).

Manuscript received January 19, 2020; revised March 16, 2020; accepted May 1, 2020

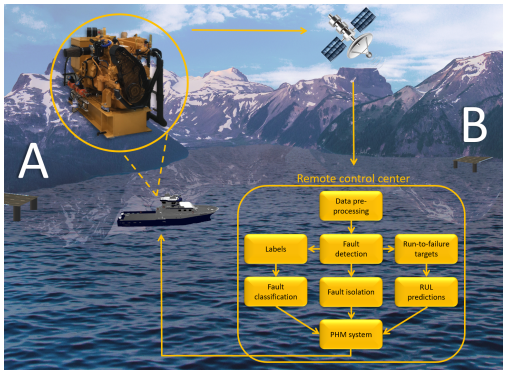


Fig. 1. Illustration of an autonomous ferry, crossing a fjord from dock A to B. Since there are limited amounts of crew members onboard, such ferries need to transfer real-time operational sensor data to a remote control center to conduct the essential actions of a data-driven PHM system. Then, maintenance operations can be scheduled to the next appropriate port of call.

abstract features to produce a better reconstruction of the input data.

The marine diesel engine is one of the most critical components onboard ferries since it has an important role in both propulsion and power generation [12]. It is subjected to rapid variations in operational loads, depending on both the task of operation and environmental conditions. In such complexity, the degradation phenomena cannot be presented directly for cutting-edge spectral anomaly detection algorithms since the sensor measurements are highly connected to the operational loads. Hence, a multi-regime normalization method has to be performed on the raw input data to present the degradation phenomena [13]. Additionally, the nature of degradation of typical fault-types associated with the marine diesel engine might be different from one another and significantly similar to normal operation data.

This paper proposes a fault-type independent spectral anomaly detection algorithm for marine diesel engine degradation in autonomous ferries. The variational autoencoder (VAE) is the selected DNN as it outperforms a feed-forward neural network (FNN) with one hidden layer, the traditional autoencoder (AE), and the long-short term memory (LSTM), in terms of reconstruction-based fault detection for maritime components in [9]. As similar to [14], a replicated autonomous ferry crossing operation is used to produce two engine load profiles. These profiles reflect different environmental conditions affecting the ferry. Both normal operation and faulty degradation data sets are collected from the two profiles, and a fault is introduced at an unknown time step in the degradation data sets. During the experiments, three fault-types with different nature of degradation are used for both validation and final test of the proposed algorithm. The complete algorithm is summarized as follows: First, the VAE is trained on pre-processed normal operation data. Second, the trained VAE is used to calculate the velocity and the acceleration of the anomaly score at each time step in faulty degradation data. Simultaneously, generic and dynamic threshold limits are established. Both the calculations and the threshold limits change dynamically with time. This enables online fault detection as a fault is detected automatically once the velocity and acceleration calculations exceed the threshold limits.

The proposed algorithm is based on our already published fault detection algorithm in [9]. Our previous algorithm makes only offline fault detection possible. However, as opposed to our previous algorithm, the proposed algorithm in this study includes two principal improvements, that is, online and fault-type independent anomaly detection by utilizing generic and dynamic threshold limits. This study's main contributions are as follows:

- A fault-type independent spectral anomaly detection algorithm for marine diesel engine degradation in autonomous ferries is proposed.
- Generic and dynamic threshold limits are proposed to predict the fault time step online.
- The algorithm is independent of fault-types with different nature of degradation related to the marine diesel engine.

The overall organization of the paper is as follows. Section II introduces relevant and related work on spectral anomaly detection. Section III introduces the essential background on the VAE and the semi-supervised reconstruction framework. The experimental approach is explained in detail in section IV. Results and discussions are elaborated in section V. Finally, section VI concludes the paper and presents objectives for future work.

II. Related work

Three different learning procedures exist for spectral anomaly detection algorithms: supervised, semi-supervised, and unsupervised. The availability and quality of the input data largely determine which learning procedure to choose for fault detection. Supervised learning involves training a supervised binary or multi-class classifier to differentiate normal operation data from faulty degradation data. This procedure is extremely powerful if predefined labels for both normal and faulty data points are available during the training stage.

G. Wu proposed a supervised FNN for fault detection of ship equipment in [15]. In [16], Xu et al. proposed an online fault diagnostics method based on convolutional neural networks (CNNs) and transfer learning. The proposed approach was trained in a supervised manner where a softmax output layer was used to classify faults related to both bearings and pumps. A supervised classifier was also used for fault detection in [17]. In this study, however, Sun et al. utilized an initial unsupervised learning procedure, before supervised fine-tuning, to do automatic feature extractions of rolling element bearings. Siegel et al. examined methods for detecting and disrupting arc faults in [18]. Both a binary and multi-class classifier were used during real-time classification experiments.

Even though the above studies have shown superior accuracy in terms of fault detection, there is a lack of labeled faults in the maritime industry [19]. This necessitates the use of semi-supervised or unsupervised learning, which does not require predefined fault labels. In the application of fault

detection, semi-supervised learning only uses normal operation data for training, while unsupervised learning has no previous knowledge of the input data where only intrinsic properties are used [20].

The sensors installed on autonomous ferries can be utilized to accumulate and collect normal operation data to use a semi-supervised learning framework. A VAE was used for anomaly detection in [9], [14]. In both studies, the maximum acceleration in faulty degradation data was estimated and used as the fault detector. However, utilizing the maximum acceleration makes only offline fault detection possible. This is because one would need the faulty degradation data in advance to determine the maximum acceleration.

The utilization of dynamic threshold limits can enable online fault detection. Park et al. [21] proposed an LSTM-based VAE anomaly detector for robot-assisted feeding. A varying state-based threshold value was used to detect anomalies. Thus, online anomaly detection is possible where the threshold value changes over the estimated state of task execution. Additionally, Hundman et al. [22] used non-parametric dynamic thresholds for spacecraft anomaly detection. Nevertheless, these studies apply the dynamic thresholds based on the raw anomaly score or a smooth version of it. For the marine diesel engine, such dynamic thresholds will reflect the nature of degradation of the specific fault-type used for fault detection. Therefore, different dynamic thresholds have to be created for different faults. This contradicts the fact that the goal of the improved fault detection algorithm in this study is to be fault-type independent.

III. Background

This section introduces the background theory on the VAE and the semi-supervised reconstruction framework.

A. Variational autoencoder

The VAE is a variant of the traditional autoencoder (AE) rooted in Bayesian inference [23]. It is composed of an encoder function $z = q_{\theta_e}(z|x)$ and a decoder function $r = p_{\theta_d}(x|z)$. The encoder approximates the underlying probability distribution $p_{\theta_d}(z)$. Then, new data can be generated utilizing the decoder by sampling a set of latent variables z obtained from $p_{\theta_d}(z)$. By modeling the distribution of the latent variables instead of deterministic values, as conducted in the traditional AE, the VAE improves generalization since z are stochastic in nature [24]. Note that θ_e and θ_d are the biases and weights of the encoder and decoder, respectively. The VAE optimizes θ_e and θ_d by maximizing the variational lower bound J_{VAE} [23]:

$$\begin{aligned} J_{VAE}(\theta_e, \theta_d) = &-D_{KL}\big(q_{\theta_e}(z|x)\,||\,p_{\theta_d}(z)\big) \\ &+E_{q_{\theta_e}(z|x)}[\log\, p_{\theta_d}(x|z)] \end{aligned} \tag{1}$$

where D_{KL} is the Kullback-Leibeler (KL) divergence. The KL divergence measures the similarity between the prior distribution of z, $p_{\theta_d}(z)$, and the variational approximation $q_{\theta_e}(z|x)$. Maximizing J_{VAE} minimizes the KL divergence, hence pushing the approximated posterior $q_{\theta_e}(z|x)$ towards the prior $p_{\theta_d}(z)$. The common choice of the prior distribution is a Gaussian distribution, $\mathcal{N}(\mu_z, \Sigma_z)$, where a standard normal distribution $\mathcal{N}(0, 1)$ is utilized. The second expression is the reconstruction log-likelihood of x with sampling from $q_{\theta_e}(z|x)$ and referred to as the generative loss. The distribution of the second expression depends on the data type [10]. For real-valued input data, a Multivariate Gaussian is normally used.

The reconstruction log-likelihood needs to be calculated through Monte Carlo methods [23]. However, since these methods suffer from high variance and high computation resources, a reparameterization trick of z is used to obtain the gradients of the decoder in order to use the back-propagation algorithm. The random variable $z \sim q_{\theta_e}(z|x)$ is replaced by a deterministic transformation, such that, $z = \boldsymbol{\mu} + \boldsymbol{\sigma}\varepsilon$, $\varepsilon \sim \mathcal{N}(0, 1)$ [10]. Thus, given a fixed input x and a variable ε, the total function is deterministic and continuous, meaning back-propagation can compute a gradient that will work for stochastic gradient descent [23]. Then, the encoder only needs to produce vectors of means $\boldsymbol{\mu}$ and standard deviations $\boldsymbol{\sigma}$ instead of vectors of real values.

B. Semi-supervised reconstruction framework

As in [9], the fault detection is conducted through a semi-supervised reconstruction framework, meaning only normal operation data is used for training the VAE. Consider $\boldsymbol{x}_t = [x_1, ..., x_n]_t$ as the input vector at time step t. To enable the VAE to reconstruct the normal operation data, $\boldsymbol{x}_t$ is also used as the target $\boldsymbol{y}_t$ for reconstruction at each t. In this way, the trained VAE is expected to produce relatively large reconstruction errors on unseen degradation data. Since the data gathered from the marine diesel engine is continuous sensor data, a fully connected output layer is attached to the VAE, where the mean squared error (MSE) is utilized to measure the reconstruction capability. Thus, the VAE minimizes the following loss function:

$$L_{VAE} = \frac{1}{n}\sum_{i=1}^{n} ||\hat{y}_i - y_i||^2 \tag{2}$$

where n is the number of input features, and $\hat{y}_i$ and y_i is the i_{th} reconstructed and target measurement, respectively.

As in [9] and [14], the VAE is structured with two hidden layers (h_1, h_2) in the encoder, z units in the latent layer and two hidden layers (h_2, h_1) in the decoder. However, the number of hidden units in each layer differs from the previous studies as they are determined related to the number of input features n:

$$h_1 = \lfloor 1.2n \rfloor, \quad h_2 = \lfloor h_1/2 \rfloor, \quad z = \lfloor h_2/2 \rfloor \tag{3}$$

where $\lfloor \quad \rfloor$ is round down symbol.

IV. Experimental study

The following experimental study, uses Microsoft Windows 10, Java 8, "deeplearning4j" version 1.0.0-beta4 [25] as the deep learning library, and NVIDIA GeForce GTX 1060 6 GB as the graphics processing unit.

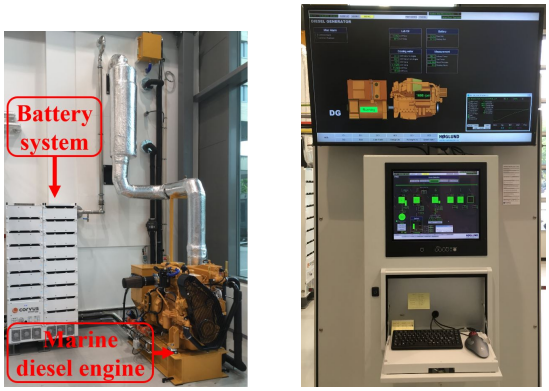


Fig. 2. The battery system, the marine diesel engine, and the automation system used for collecting the data sets.

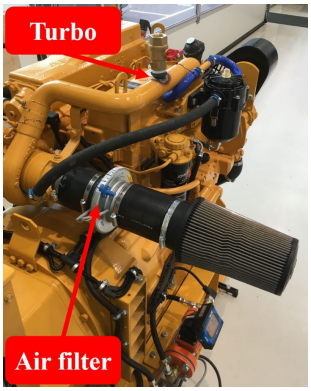


Fig. 4. The restriction and bleed device used to provoke the air filter and turbo fault, respectively.

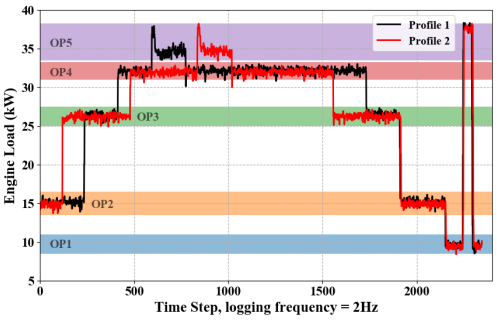


Fig. 3. Engine load profile 1 and 2.

TABLE I
THE SEVEN DATA SETS COLLECTED FROM THE HYBRID POWER LAB

Data set	Profile	Usage	Seconds	Hz	Time steps
Normal operation	1	Training	1173	2	2346
Normal operation	2	Training	1173	2	2346
Turbo degradation	1	Validation	1173	2	2346
Turbo degradation	2	Validation	1173	2	2346
Air filter degradation	1	Validation	1173	2	2346
Air filter degradation	2	Validation	1173	2	2346
Cooling degradation	1	Final test	1173	2	2346

A. Data sets

A hybrid power lab, established by the Department of Ocean Operations and Civil Engineering at the Norwegian University of Science and Technology in Aalesund, is used to collect the data sets. The lab intends to research ship autonomy. As seen in Figure 2, the lab includes a small marine diesel engine, a marine battery system, and a marine automation system to control the facilities. The power produced is supplied back to the power grid to simulate load variations in the system.

During the data collection process, the engine is driven by two different engine load profiles. As similar to [14], the two engine load profiles aim to replicate real-life autonomous ferry crossings on the west coast of Norway. First, the ferry is off-loading and on-loading vehicles before it leaves shore at a safe and constant velocity. Then, the ferry speeds to a suitable velocity. This velocity remains constant until it decreases safely. Finally, the ferry breaks just before it docks. In common, the two profiles are exposed to the same magnitudes of engine loads, but the length of each engine load differs to reflect different environmental conditions. Figure 3 shows the two engine load profiles, profile 1 and profile 2.

In this study, two fault-types are used for validation of the proposed algorithm. These are the air filter fault, the clogging of the air filter, and the turbo fault, malfunction of the turbocharger. The air filter fault demonstrates the effect of a clogged air filter with the use of a restriction device, as seen in Figure 4. During the data collection process, this device is gradually adjusted from fully open to 90% closed to reduce the inlet flow of air to the turbocharger. The purpose of the turbo fault is to replicate efficiency reduction in the turbocharger. This is done by installing a bleed device on the charge air pipe between the turbocharger and the engine inlet manifold, as seen in Figure 4. Gradually bleeding of air during the data collection process results in reduced air pressure to the engine combustion process. A third fault-type is used for the final test of the proposed algorithm: a malfunction of the frequency-operated fan controlling the secondary cooling system in the engine. This fault, which appears in our previous work [14], is hereinafter referred to as the cooling fault. One normal operation data set, one turbo degradation data set, and one air filter degradation data set is collected from each profile. Additionally, one cooling degradation data set is collected from profile 1. Table I summarizes the seven data sets collected from the hybrid power lab.

B. Feature selection

All collected data sets include 47 input features from the hybrid power lab. As discovered in [14], features belonging to the battery system and the automation system are irrelevant for detecting faults in the marine diesel engine. When such features are removed, the VAE will provide a reconstruction process with higher degradation relevance. Additionally, fea-

TABLE II
FEATURE SELECTION FOR THE MARINE DIESEL ENGINE

Index	Description	Unit
1	Boost pressure	bar
2	Engine load	kW
3	Engine cooling water temperature	°C
4	Engine exhaust gas temperature	°C
5	Cooling water temperature out of the engine	°C
6	Engine speed	rpm
7	Diesel generator cooling water flow	liter/min
8	Simulated propulsion load	kW
9	Cooling fan speed controller	rpm

tures with constant measurements are removed since these features provide no degradation information. The Pearson correlation analysis is also used to detect the linear relationship between the input features. If two input features have a high linear relationship, they likely contain redundant information. Then, expert human domain knowledge (HDK) is used to determine which of the redundant input features has less degradation relevance. Actually, in this study, the HDK is acquired from an engine chief engineer with 13 years of sailing experience and three years of experience with the development of a health monitoring system for rotating machinery. The redundant features are removed accordingly.

HDK is also used to remove inaccurate and unknown feature measurements concerning the marine diesel engine. For instance, the cooling water temperature to the engine is removed since it is considered an unknown parameter. This feature is affected by the outdoor temperature, and hence, it varies when data sets are collected at different dates and seasons. Fuel consumption is also removed from the data sets. While it is an important feature for detecting faults in the combustion process in the engine, the measurements obtained from the automation system were too inaccurate to be used in this study. Ultimately, nine input features, which are intended to reflect all degradation patterns in the marine diesel engine, remain in all data sets. Table II lists the final input features.

C. Multi-regime operating conditions and normalization

As seen in Figure 3, the engine load changes drastically during the ferry crossing operation in both profiles. As a result, feature measurements are highly connected to the engine loads. This causes the feature measurements in the normal operation data to differ strongly between different engine loads. Thus, proper data pre-processing, in terms of multi-regime normalization, is necessary to present the actual normal operation phenomena for the VAE during the training phase [13].

Obviously, both profiles fall into five distinct operating conditions based on the engine load. First, the normal operation data sets in Table I are split into five data sets each based on the five operating conditions. Each feature in these data sets is then scaled with zero mean and unit variance (z-score) normalization:

$$\overline{x}_n^o = \frac{x_n^o - \mu^o}{\sigma^o} \tag{4}$$

where x_n is the input feature, $n = 1, 2, ..., 9$, in operating condition o, and μ and σ is the population mean and population standard deviation of that feature. This yields five different normalization statistics, one for each operating condition. Finally, these normalization statistics are applied both to the raw normal operation data in the training phase and to the raw faulty degradation data in the anomaly detection. To apply different normalization statistics, the engine load is monitored at each time step.

TABLE III
HYPER-PARAMETERS

Hyper-parameter	Method/Value
Activation function	Rectified Linear Unit
Learning rate	$1 \cdot 10^{-3}$
$l2$ regularization	$1 \cdot 10^{-4}$
Optimization algorithm	Stochastic Gradient Descent
Optimizer	Adam
Weight initialization	Xavier

D. Training phase and anomaly detection

In the training phase, a VAE is established and trained on both normal operation data sets subjected to multi-regime normalization. An early stopping policy is utilized to reconstruct the normal operation data as precisely as possible by monitoring the average reconstruction error of all mini-batches. If the number of epochs with no decrease in the average reconstruction error is greater than four, the training phase is ended. Then, the VAE, in the epoch with the lowest average reconstruction error, is stored and used for anomaly detection. The mini-batch size is set to 128. The VAE is configured with hyper-parameters that provided great success for maritime components in [9]. These are shown in Table III.

In terms of time series data, it is practical to consider three categories of anomalies: point, collective, and contextual [11]. Point anomalies are single values that differ from previous values, collective anomalies are entire sequences of values that are anomalous, and contextual anomalies are single values

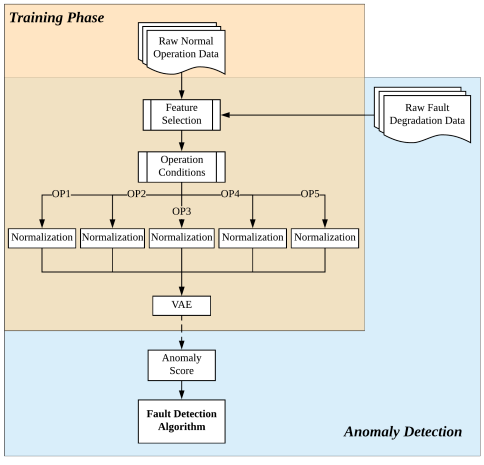


Fig. 5. A complete flowchart of the training phase and anomaly detection.

that are not different from previous values yet are anomalous concerning local values [22]. The nature of degradation in both the turbo and air filter fault is highly connected to the operating conditions and therefore they should be regarded as contextual anomalies. However, the VAE is only able to detect point anomalies. To detect contextual anomalies, the VAE has to be applied within a context [11]. We consider the five operating conditions as different contexts. Thus, the multi-regime normalization statistics also need to be applied to the faulty degradation data to merge the five different contexts into one context. In this way, the VAE can be used for anomaly detection and to estimate an anomaly score at each time step. Figure 5 shows a complete flowchart of the training phase and anomaly detection.

E. Fault detection algorithm

1) Online fault detection: The anomaly score in faulty degradation data AS_d is estimated by using the trained VAE to calculate the MSE, Eq. 2, at each time step t. Then, the algorithm generates three sliding windows of length w to smooth AS_d:

$$w = \frac{T_d}{p} \tag{5}$$

where T_d is the total number of time steps in the faulty degradation data and p is an adjustable parameter. p determines the magnitude of smoothing conducted on AS_d. Hence, careful tuning of p is necessary since excessive smoothing might obscure important degradation trends. The three windows slide across AS_d for each t. A distance equivalent to w is used between each window. Simultaneously, the average anomaly score $AS_{d,avg}$ is computed in each window. Additionally, the velocity v_d between windows 1 and 2 and between windows 2 and 3, and the acceleration a_d between the two velocities are calculated. Finally, the velocity fault time step $\hat{f}_{t,v}$ and the acceleration fault time step $\hat{f}_{t,a}$ are detected when v_d and a_d exceeds their dynamic threshold limits, respectively. The proposed algorithm is shown in Algorithm 1.

Large sensor measurement deviations compared to sensor measurements in normal operation data is a strong indication of an incipient fault [9]. These deviations can, of course, be detected by utilizing AS_d or $AS_{d,avg}$ as the fault detectors. However, both AS_d and $AS_{d,avg}$ will vary between different fault-types since they reflect the nature of degradation. Consequently, the corresponding threshold limits will be highly fault-dependent. The main goal of the proposed algorithm is to be fault-type independent. v_d will measure the rapidity in $AS_{d,avg}$ and indicate if one or several sensor measurements have begun to diverge swiftly from normal operation data. However, a_d will measure increases and decreases in v_d. Due to latency in the marine diesel engine, a_d might be a better indication than v_d since there is an expected time delay before the faults will result in large sensor measurement deviations. Therefore, v_d and a_d are considered as more suitable fault detectors for the algorithm since the calculations are assumed to be similar between different fault-types. Consequently, generic and fault-independent threshold limits can be acquired. These limits are further elaborated in the following paragraph.

Algorithm 1 Algorithm for detecting the fault time step in faulty degradation data.

Input: T_d, AS_d, p, v_n, v_{lower}, v_{upper}, a_n, a_{lower}, a_{upper}
Output: $\hat{f}_{t,v}$, $\hat{f}_{t,a}$

```
Initialization :
w ← T_d / p
v_{d,first} = true
a_{d,first} = true
Generate three sliding windows of length w to slide across
AS_d for each t. AS_{d,avg} is computed in each window.
A distance equivalent to w is used between each window.
for t := 1 to T_d do
  v_d1 ← AS_{d,avg}1 - AS_{d,avg}2
  v_d2 ← AS_{d,avg}2 - AS_{d,avg}3
  a_d ← v_d1 - v_d2
  if (v_{d,first} = true) then
    if (v_d1 > v_n[t] + v_upper or v_d1 < v_n[t] + v_lower)
    then
      f̂_{t,v} ← t - (w · 1.5)
      v_{d,first} = false
    end if
  end if
  if (a_{d,first} = true) then
    if (a_d > a_n[t] + a_upper or a_d < a_n[t] + a_lower)
    then
      f̂_{t,a} ← t - (w · 2.5)
      a_{d,first} = false
    end if
  end if
end for
return f̂_{t,v}, f̂_{t,a}
```

2) Generic and dynamic threshold limits: In this study, the threshold limits are based on the velocity v_n and the acceleration a_n in the average anomaly score of normal operation data for both profiles. The procedure to measure both v_n and a_n is exactly the same as in Algorithm 1. Seven different p values, in the 30 to 90 range, are used during the experiments. In order to obtain the associated dynamic threshold limits, the minimum and maximum velocities of v_n, v_{min}, and v_{max}, and the minimum and maximum accelerations of a_n, a_{min} and a_{max}, are calculated for each p value in each profile. Then, a common set of upper and lower thresholds for both v_n and a_n are calculated based on the following formulas:

$$v_{upper} = \frac{|(v_{max,1} + v_{max,2}) - (v_{min,1} + v_{min,2})|}{2} \tag{6}$$

$$v_{lower} = -v_{upper} \tag{7}$$

$$a_{upper} = \frac{|(a_{max,1} + a_{max,2}) - (a_{min,1} + a_{min,2})|}{2} \tag{8}$$

$$a_{lower} = -a_{upper} \tag{9}$$

The common set of upper and lower thresholds for each p value are shown in Table IV. The limits will change dynamically through time when the upper and lower thresholds are added to v_n and a_n, as performed in Algorithm 1.

TABLE IV
A COMMON SET OF UPPER AND LOWER THRESHOLDS FOR BOTH THE VELOCITY AND THE ACCELERATION

p	v_{lower}	v_{upper}	a_{lower}	a_{upper}
30	-2.63	2.63	-4.10	4.10
40	-3.40	3.40	-4.97	4.97
50	-3.81	3.81	-6.35	6.35
60	-4.40	4.40	-7.26	7.26
70	-5.10	5.10	-8.58	8.58
80	-5.74	5.74	-9.67	9.67
90	-6.32	6.32	-10.54	10.54

The generic and dynamic threshold limits are computed before they are applied in the fault detection algorithm. However, new engine load profiles are likely to be encountered in real-life data-driven PHM systems in autonomous ferries. Then, the computation complexity will increase since v_{min}, v_{max}, a_{min}, and a_{max} of the new profile have to be calculated and incorporated in Eqs. 6, 7, 8, and 9 before new fault detections can start.

V. EXPERIMENTAL RESULTS AND DISCUSSIONS

In this study, both velocity and acceleration calculations will be used as the fault detectors. The air filter and turbo degradation in both profiles will be used as the validation data sets for the proposed algorithm. The validation aims to discover the best performing fault detector and the most suitable p value. Seven different p values, in the 30-90 range, will be compared. A low p value might smooth the anomaly score too much and ignore significant degradation patterns. In contrast, a high p value might provide irrelevant spikes that would also affect the velocity and acceleration calculations. In the end, the final experiment will use the cooling degradation as the final test data set of the algorithm. This experiment aims to further test the fault-type independence of the algorithm.

A. Validation

To validate both $\hat{f}_{t,v}$ and $\hat{f}_{t,a}$, the true fault time step f_t has to be determined. Since both the air filter fault and the turbo fault in both profiles are provoked gradually during the data collection process, f_t can not be decided based a recorded time step. Thus, f_t is determined based on expert HDK. The boost pressure is the key feature to monitor for fault detection for both fault-types. As already mentioned, both faults-types are highly connected to the engine loads and subjected to different nature of degradation. Therefore, f_t is determined where the deviation in boost pressure between normal operation data and faulty degradation data in percentage is largest. The determined f_t for both fault-types in both profiles is shown in Table V.

Table VI shows $\hat{f}_{t,v}$ and $\hat{f}_{t,a}$ for each p value in both profiles for both fault-types. The accuracy evaluations, Acc_v and Acc_a, are based on the following formula:

$$Acc\,(\%) = \left(1 - \frac{||\hat{f}_t - f_t||}{T_d}\right) \cdot 100 \tag{10}$$

where $Acc\,(\%)$ can be considered as the distance between the detection and f_t. In the following discussions, a satisfactory accuracy is considered to be above 95%. For the air filter fault in profile 1, $\hat{f}_{t,v}$ provides satisfactory accuracy by p values between 70 and 90, while $\hat{f}_{t,a}$ provides satisfactory accuracy by p values between 40 and 90. On the other hand, for the air filter fault in profile 2, $\hat{f}_{t,v}$ provides satisfactory accuracy by p values between 30 and 70, while $\hat{f}_{t,a}$ provides satisfactory accuracy by p values between 30 and 60. As Table V reflects, the air filter fault in profile 2 is subjected to a lower deviation in boost pressure than the air filter fault in profile 1. As a consequence, the air filter fault in profile 2 is subjected to lower magnitudes of both velocity and acceleration calculations, and hence, requires smaller upper and lower thresholds. As Table IV shows, low p values result in smaller upper and lower thresholds. This issue reflects the difficulty of creating generic and dynamic threshold limits even for the same fault-type that is subjected to different environmental conditions in the form of different engine load

TABLE V
THE TRUE FAULT TIME STEP f_t

Fault-type	Profile	Largest deviation in boost pressure (%)	f_t
Air filter	1	15.79	1670
	2	10.53	1433
Turbo	1	21.05	1431
	2	21.05	1427

TABLE VI
VALIDATION: THE TRUE FAULT TIME STEP f_t COMPARED TO THE DETECTED FAULT TIME STEP $\hat{f}_t$

Fault-type	Profile	f_t	p	w	$\hat{f}_{t,v}$	$Acc_v(\%)$	$\hat{f}_{t,a}$	$Acc_a(\%)$
Air filter	1	1670	30	78	1255	82.31	1502	92.84
			40	58	1278	83.29	1609	97.40
			50	46	1289	83.76	1648	99.06
			60	39	1549	94.84	1660	99.57
			70	33	1566	95.57	1674	99.83
			80	29	1706	98.47	1680	99.57
			90	26	1709	98.34	1682	99.49
	2	1433	30	78	1362	96.97	1428	99.79
			40	58	1392	98.25	1445	99.49
			50	46	1404	98.76	1458	98.93
			60	39	1532	95.78	1483	97.87
			70	33	1540	95.44	0	38.92
			80	29	0	38.92	0	38.92
			90	26	0	38.92	0	38.92
Turbo	1	1431	30	78	731	70.16	693	68.54
			40	58	771	71.87	745	70.76
			50	46	786	72.51	752	71.06
			60	39	794	72.85	1347	96.42
			70	33	368	54.69	1362	97.06
			80	29	1395	98.47	1374	97.57
			90	26	1399	98.64	1381	97.87
	2	1427	30	78	951	79.71	892	77.20
			40	58	979	80.90	929	78.77
			50	46	991	81.42	1329	95.82
			60	39	1005	82.01	1347	96.59
			70	33	1387	98.29	1361	97.19
			80	29	1393	98.55	1371	97.61
			90	26	1397	98.72	1378	97.91

TABLE VII
VALIDATION: THE AVERAGE ACCURACY FOR EACH p VALUE

p	w	Avg. $Acc_v(\%)$	Avg. $Acc_a(\%)$
30	78	82.29	84.59
40	58	83.58	86.60
50	46	84.11	91.22
60	39	86.37	**97.61**
70	33	86.00	83.25
80	29	83.60	83.42
90	26	83.65	83.55

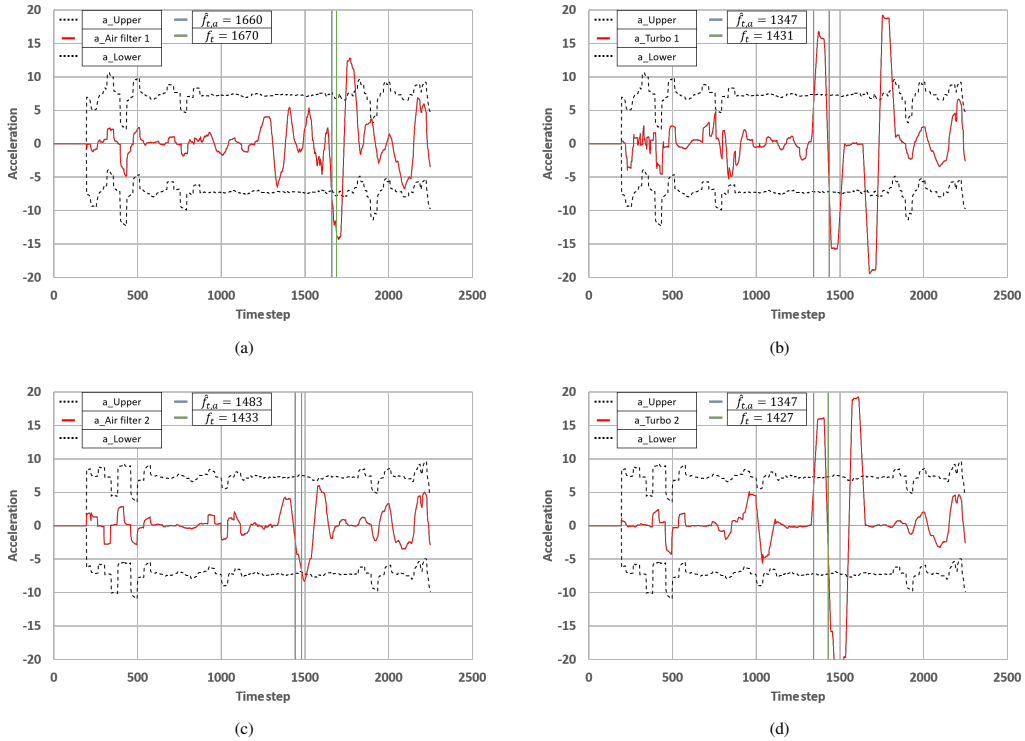


Fig. 6. Automatic fault detection where $p = 60$ and the acceleration is used as the fault detector. (a) Air filter fault in engine load profile 1. (b) Turbo fault in engine load profile 1. (c) Air filter fault in engine load profile 2. (d) Turbo fault in engine load profile 2.

profiles.

For the turbo fault in profile 1, $\hat{f}_{t,v}$ provides satisfactory accuracy by the p values 80 and 90, while $\hat{f}_{t,a}$ provides satisfactory accuracy by p values between 60 and 90. Similarly, for the turbo fault in profile 2, $\hat{f}_{t,v}$ provides satisfactory accuracy by p values between 70 and 90, while $\hat{f}_{t,a}$ provides satisfactory accuracy by p values between 50 and 90. Also as seen in Table V, the turbo fault in both profiles are subjected to a deviation of 21.05%, almost twice the deviation compared to the air filter fault in profile 2. This results in larger magnitudes of both velocity and acceleration calculations. Thus, the turbo fault in both profiles provides the highest accuracies by high p values and corresponding large upper and lower dynamic threshold limits.

To determine the best performing fault detector and the most suitable p value for both fault-types, the average velocity and acceleration accuracy for each p value is calculated, as shown in Table VII. When $p = 60$, the acceleration provides the highest average accuracy of 97.61%. Therefore, the acceleration is considered the most fault-independent fault detector. Figure 6 shows the acceleration calculations and the corresponding dynamic threshold limits when $p = 60$ for both fault-types in both profiles. It is worth mentioning that the acceleration calculations and the dynamic threshold limits are not plotted before the entire sliding window operation is active. In other words, the initial 195 time steps are plotted as zeros $(w(60) \cdot 5 = 195)$.

B. Final test

The main intention of the final test of the proposed algorithm is to further test its independence towards different fault-types. The cooling degradation data in profile 1 is used for this purpose as this fault exhibits a totally different nature of degradation compared to both the air filter fault and the turbo fault. Thus, it can be considered to be new field data that the algorithm has never seen before. To evaluate the fault detection, the true fault time step f_t for the cooling fault is also determined based on expert HDK. When the cooling water temperature increases 85 °C, f_t is determined to be 1713.

As discovered in the validation, the acceleration is the best performing fault detector when $p = 60$. These settings are therefore used in the final test. As Table VIII shows, the algorithm detects the cooling fault with an accuracy of 97.66%. Also noted, both in the validation and the final test the trend is that the acceleration provides early detections, i.e. $\hat{f}_{t,acc} < f_t$, when $p = 60$. However, early detections with a corresponding high accuracy are considered as valid detections since there is an expected time delay in the marine

TABLE VIII
FINAL TEST: THE TRUE FAULT TIME STEP f_t COMPARED TO THE DETECTED FAULT TIME STEP $\hat{f}_t$ FOR COOLING DEGRADATION DATA

Fault-type	Profile	f_t	p	w	$\hat{f}_{t,a}$	$Acc_a(\%)$
Cooling	1	1713	60	39	1658	97.66

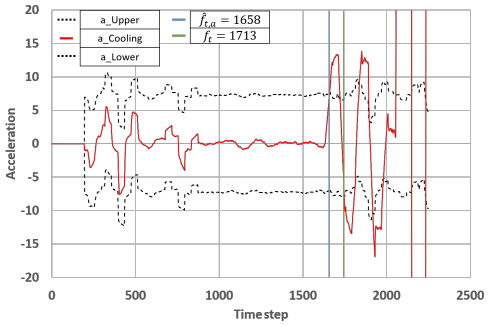


Fig. 7. Automatic fault detection where $p = 60$ and the acceleration is used as the fault detector for cooling degradation data.

diesel engine before the faults will result in large sensor measurements deviations. Figure 7 shows the acceleration calculations and the corresponding dynamic threshold limits for the fault detection of the cooling degradation data. The final test proves that the algorithm is fault-type independent.

VI. CONCLUSION AND FUTURE WORK

This paper has analyzed and proposed a fault-type independent spectral anomaly detection algorithm for marine diesel engine degradation in autonomous ferries where a VAE is used as the DNN. To do so, three fault-types with different nature of degradation have been used during the experiments. Both normal operation data and faulty degradation data have been collected from two different engine load profiles. These profiles aim to replicate real autonomous ferry crossing operations that might affect the ferry.

In the validation of the proposed algorithm, the acceleration has proven to be the most fault-independent fault detector, providing an average accuracy of 97.61%. Additionally, the acceleration achieved an accuracy of 97.66% in the final test of the algorithm. Thus, the algorithm has proved its independence of fault-types with different nature of degradation related to the marine diesel engine.

In this study, the engine loads were divided into five distinct operating conditions manually to do multi-regime normalization. However, if new operating conditions are encountered in real-life systems, this process has to be automated. For instance, through unsupervised clustering algorithms, such as the K-Means algorithm. One has to remember that fault detection is only the first action to be performed in a real-life data-driven PHM system. However, the detected fault time steps obtained from the faulty degradation data can be used to automatically label the data to account for both fault classification and RUL predictions. Also, due to the VAE's generative characteristics, it is possible to derive the reconstruction of the data to analyze the underlying cause of the fault to do fault isolation. Our future work will include these crucial actions.

ACKNOWLEDGMENT

This work was supported by the Norwegian University of Science and Technology within the Department of Ocean Operations and Civil Engineering under project no. 90329106. The authors would like to thank Digital Twins For Vessel Life Cycle Service and the Research Council of Norway, grant no. 280703.

REFERENCES

[1] X. Cheng, G. Li, A. L. Ellefsen, S. Chen, H. P. Hildre, and H. Zhang, "A novel densely connected convolutional neural network for sea state estimation using ship motion data," *IEEE Transactions on Instrumentation and Measurement*, pp. 1–1, 2020.
[2] R. Skulstad, G. Li, T. I. Fossen, B. Vik, and H. Zhang, "Dead reckoning of dynamically positioned ships: Using an efficient recurrent neural network," *IEEE Robotics Automation Magazine*, vol. 26, no. 3, pp. 39–51, Sep. 2019.
[3] E. Jokioinen, "Remote and autonomous ships - the next steps: Introduction," *Rolls-Royce, Buckingham Gate, London: The Advanced Autonomous Waterborne Applications (AAWA)*, pp. 4–14, 2016.
[4] O. Levander, "Autonomous ships on the high seas," *IEEE Spectrum*, vol. 54, no. 2, pp. 26–31, 2017.
[5] L. Kretschmann, H.-C. Burmeister, and C. Jahn, "Analyzing the economic benefit of unmanned autonomous ships: An exploratory cost-comparison between an autonomous and a conventional bulk carrier," *Research in transportation business & management*, vol. 25, pp. 76–86, 2017.
[6] A. L. Ellefsen, V. Æsøy, S. Ushakov, and H. Zhang, "A comprehensive survey of prognostics and health management based on deep learning for autonomous ships," *IEEE Transactions on Reliability*, vol. 68, no. 2, pp. 720–740, 2019.
[7] K. E. Knutsen, G. Manno, and B. J. Vartdal, "Beyond condition monitoring in the maritime industry," *DNV GL Strategic Research & Inovation Position Paper*, 2014.
[8] P. W. Kalgren, C. S. Byington, M. J. Roemer, and M. J. Watson, "Defining phm, a lexical evolution of maintenance and logistics," in *2006 IEEE Autotestcon*, Sept 2006, pp. 353–358.
[9] A. L. Ellefsen, E. Bjørlykhaug, V. Æsøy, and H. Zhang, "An unsupervised reconstruction-based fault detection algorithm for maritime components," *IEEE Access*, vol. 7, pp. 16 101–16 109, 2019.
[10] J. An and S. Cho, "Variational autoencoder based anomaly detection using reconstruction probability," *SNU Data Mining Center - Special Lecture on IE*, vol. 2, no. 1, 2015.
[11] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009.
[12] J. A. P. Rubio, F. Vera-García, J. H. Grau, J. M. Cámara, and D. A. Hernandez, "Marine diesel engine failure simulator based on thermodynamic model," *Applied Thermal Engineering*, vol. 144, pp. 982–995, 2018.
[13] O. Bektas, J. A. Jones, S. Sankararaman, I. Roychoudhury, and K. Goebel, "A neural network filtering approach for similarity-based remaining useful life estimation," *The International Journal of Advanced Manufacturing Technology*, vol. 101, no. 1-4, pp. 87–103, 2019.
[14] A. L. Ellefsen, X. Cheng, F. T. Holmeset, S. Ushakov, V. Æsøy, and H. Zhang, "Automatic fault detection for marine diesel engine degradation in autonomous ferry crossing operation," in *2019 IEEE International Conference on Mechatronics and Automation (ICMA)*, Aug 2019, pp. 2195–2200.
[15] G. Wu, "Fault detection method for ship equipment based on bp neural network," in *2018 International Conference on Robots & Intelligent System (ICRIS)*. IEEE, 2018, pp. 556–559.
[16] G. Xu, M. Liu, Z. Jiang, W. Shen, and C. Huang, "Online fault diagnosis method based on transfer convolutional neural networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 2, pp. 509–520, Feb 2020.

[17] J. Sun, C. Yan, and J. Wen, "Intelligent bearing fault diagnosis method combining compressed data acquisition and deep learning," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 1, pp. 185–195, Jan 2018.
[18] J. E. Siegel, S. Pratt, Y. Sun, and S. E. Sarma, "Real-time deep neural networks for internet-enabled arc-fault detection," *Engineering Applications of Artificial Intelligence*, vol. 74, pp. 35 – 42, 2018.
[19] A. S. Zymaris, Ø. Å. Alnes, K. E. Knutsen, and N. M. Kakalis, "Towards a model-based condition assessment of complex marine machinery systems using systems engineering," in *Proc. 3rd Eur. Conf. Prognostics Health Manage. Soc.*, 2016, pp. 1–15.
[20] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.
[21] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1544–1551, 2018.
[22] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 387–395.
[23] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.
[24] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.
[25] "Eclipse deeplearning4j development team, deeplearning4j: Open-source distributed deep learning for the jvm," *Apache Software Foundation License 2.0, http://deeplearning4j.org*, 2020.

André Listou Ellefsen received his Master degree in Subsea Technology from the Norwegian University of Science and Technology (NTNU), Trondheim, Norway, in 2016. He is currently pursuing the Ph.D. degree with NTNU, Aalesund, Norway, as part of the Mechatronics Laboratory, Department of Ocean Operations and Civil Engineering. His current research interests include artificial intelligence, deep learning, decision support, predictive maintenance, prognostics and health management, and digital twins.

Peihua Han received his Bachelor and Master degree in Department of Architecture and Civil Engineering from Zhejiang University, China, in 2019. He is currently pursuing the Ph.D. degree with Norwegian University of Science and Technology (NTNU), Aalesund, Norway, as part of the Mechatronics Laboratory, Department of Ocean Operations and Civil Engineering. His current research interests include fault diagnosis and prognostics, predictive maintenance, machine learning, and uncertainty qualification.

Xu Cheng received his Master degree in Computer Science and Technology from Zhejiang University of Technology, Hangzhou, China, in 2015. He is currently working at NTNU, Aalesund, Norway, as part of the Mechatronics Laboratory within the Department of Ocean Operations and Civil Engineering, as a Ph.D. candidate. His current research interests include sensitivity analysis, neural network, ship motion modeling.

Finn Tore Holmeset has a background as a marine chief engineer with more than 13 years seagoing experience onboard different ship types and holding a professional letter as an automation mechanic as well. He received a Master degree in Management of Demanding Maritime Operations from the Norwegian University of Science and Technology (NTNU), Aalesund, Norway, in 2018. His current work is to developed the machinery lab at NTNU Aalesund and support various research work projects ongoing at the institute.

Prof. Dr. Vilmar Æsøy graduated from NTNU in 1989, and continued his research on natural gas fueled marine engines at NTNU/MARINTEK to 1997. In 1996 he received his PhD degree for his research on natural gas ignition and combustion through experimental investigations and numerical simulations. During the research period 1989-1997 he was engaged in several large R&D projects developing gas fueled engines and fuel injection systems for the diesel engine manufacturers, Wärtsilä and Bergen Diesel (Roll-Royce). From 1998 to 2002, he worked as R&D manager for Rolls-Royce Marine Deck Machinery. Since 2002 he has been employed in teaching at Aalesund University College, developing and teaching courses in marine product and systems design on bachelor and master level. From January 2010 he received the "green ship machinery" professorship. His special research interest is within the field of energy and environmental technology, with focus on combustion engines and the need for more environmental friendly and energy efficient systems.

Prof. Dr. Houxiang Zhang (IEEE Member 2004-IEEE Senior Member 2012) received his Ph.D. degree on Mechanical and Electronic Engineering from Robotics Institute, Beihang University in 2003. From 2004, he worked as Postdoctoral fellow, senior researcher at the Institute of Technical Aspects of Multimodal Systems (TAMS), Department of Informatics, Faculty of Mathematics, Informatics and Natural Sciences, University of Hamburg, Germany. In Feb. 2011, he finished the Habilitation on Informatics at University of Hamburg. Dr. Zhang joined the NTNU, Norway in April 2011 where he is a Professor on Mechatronics. Dr. Zhang has engaged into two main research areas: 1) Biological robots and modular robotics, especially on biological locomotion control, 2) Virtual prototyping in demanding marine operation. He has applied for and coordinated more than 20 projects supported by Norwegian Research Council (NFR), German Research Council (DFG), and industry. In these areas, he has published over 160 journal and conference papers as author or co-author. Dr. Zhang has received four best paper awards, and four finalist awards for best conference paper at International conference on Robotics and Automation.

G

Paper VII

Real-time Fault Prognostics in Autonomous Ferries: The Advantage of Data Augmentation and Skip Connections

André Listou Ellefsen, Vilmar Æsøy, and Houxiang Zhang, *Senior Member, IEEE*

***Abstract*—Autonomous ferries will likely be in commercial use on the west coast of Norway in the near future. Since such ferries have no maintenance personnel onboard to perform sudden maintenance actions when needed, it is vital to have accurate and reliable fault prognostics in order to schedule maintenance operations. In this context, data-driven Prognostics and Health Management has gained significant attention as a source of solutions. In this paper, we propose a novel data augmentation technique and the SkipRnet for fault prognostics of marine diesel engines in autonomous ferries. The advantages are verified on run-to-failure data of two independent fault-types in two different engine load profiles the ferries may face in real life. The first profile is used for training and validation, while the second is used for real-time testing. The proposed data augmentation technique is used to construct six different augmented data set scenarios based on the first profile. The SkipRnet requires high generalization power toward the second profile since harsh and variable environmental conditions will subject the marine diesel engine to unforeseeable operating conditions. Due to the presence of skip connections, the SkipRnet functions as an accumulation of four independent deep neural networks (DNNs). Therefore, it has the ability to tackle a wider range of complexities in new field data than DNNs without skip connections. The advantage of both data augmentation and skip connections is clearly proven throughout this paper.**

***Index Terms*—Autonomous ferry, data augmentation, prognostics and health management, remaining useful life, skip connections**

Abbreviations and Acronyms

Adam	Adaptive Moment Estimation
C-MAPSS	Commercial Modular Aero-Propulsion System Simulation
DL	Deep Learning
DL4J	Deeplearning4j
DNNs	Deep Neural Networks
FNN	Feed-forward Neural Network
GPUs	Graphics Processing Units
LSTM	Long-short Term Memory
PHM	Prognostics And Health Management
ReLU	Rectified Linear Unit
ResNets	Residual Networks
RMSE	Root Mean Square Error
RTF	Run-to-failure
RUL	Remaining Useful Life
SAE	Sparse Autoencoder
SGD	Stochastic Gradient Descent
SNR	Signal-to-noise-ratio
TL	Transfer Learning

André Listou Ellefsen, Vilmar Æsøy, and Houxiang Zhang are with the Department of Ocean Operations and Civil Engineering, as part of the Mechatronics Laboratory, Norwegian University of Science and Technology, Aalesund, 6009, Norway, (e-mail: andre.ellefsen@ntnu.no; vilmar.aesoy@ntnu.no; hozh@ntnu.no).

Manuscript submitted April 21, 2020

Nomenclature

b	Bias
d	RUL prediction minus target value
d_{ft}	Detected fault time step
$\boldsymbol{f}_t$	Forget gate in LSTM
g	Random white Gaussian noise
$\boldsymbol{h}_t$	Output of $\boldsymbol{S}_t$ in LSTM
$\boldsymbol{i}_t$	Input gate in LSTM
$\boldsymbol{o}_t$	Output gate in LSTM
P_{signal}	Average power of the signal
P_{noise}	Average power of the noise
$\boldsymbol{R}$	Recurrent weight in LSTM
$\boldsymbol{S}_t$	State of memory cell in LSTM
$\tilde{\boldsymbol{S}}_t$	New candidate state values in LSTM
T_t	Total time step length
w	Weight factors
$\boldsymbol{W}$	Input weight in LSTM
$\boldsymbol{x}$	Input vector of measurements
y	Target value
$\boldsymbol{\theta}$	Biases and weights
μ	Mean
σ	Standard deviation
ϕ	Non-linear activation function

I. Introduction

In an ideal future, autonomous ferries will be maintained, navigated, and operated without any crew involvement [1]. However, this would require reliance on fully automated systems and belonging sensor devices. Unanticipated faults and associated failures of such systems during operation pose a profound threat to both profitability and safety since there is no one to perform sudden maintenance actions when needed [2]. Autonomous ferries need to transfer real-time operational sensor data to a human-staffed control center where data-driven algorithms can be utilized to analyze previous, current, and future health conditions of components and sub-components.

The resulting analysis can then be used to schedule maintenance operations in the ferry's next appropriate port of call [3].

Prognostics and Health Management (PHM) is the area of research with the greatest promise to manage such analysis with high accuracy. A data-driven PHM system use algorithms built on historical sensor measurements to detect anomalies, isolate anomalous components, classify different fault-types, and predict the progression of faults [4]. Fault prognostics is the most significant action of a data-driven PHM system as prognostics algorithms aim to estimate the available time before an anomalous component will suffer from operational failure. Such estimations are normally referred to as the remaining useful life (RUL) and used to devise an ideal maintenance schedule. Thus, a data-driven PHM system for autonomous ferries must provide fault prognostics.

Today, fault prognostics of real-world systems remains an area of active research and development [5], [6]. Prognostics algorithms are usually divided into data-driven [7] and model-based [8] approaches. Both are based on mathematics. However, the approaches differ in that model-based approaches use algorithms that describe the physics of the component, while data-driven approaches use algorithms built on historical sensor measurements. Regarding accurate and reliable RUL predictions, data-driven approaches that address deep neural networks (DNNs) [9]–[11], have emerged as extremely powerful – if sufficient historical run-to-failure (RTF) sensor data is provided. However, problems accessing RTF data is common in the maritime industry [12]. Fortunately, data augmentation techniques enable the construction of significant RTF data based on small amounts of already collected RTF data. Nevertheless, data augmentation techniques are mostly used on image data for computer vision purposes in the deep learning (DL) domain [13]. Operational sensor data collected from autonomous ferries will primarily involve time-series data, and data augmentation techniques for this data-type are rarely seen. Thus, this paper proposes a novel data augmentation technique for RTF time series data.

DNNs are difficult to train. In the field of computer vision, residual networks (ResNets) have been introduced to simplify the training procedure of image data [14]. ResNets utilize residual connections, also known as skip connections, between convolutional layers. Skip connections allow for easier optimization since the network can skip training for layers that are not useful and do not improve accuracy. Thus, skip connections make dynamic networks that might optimally tune the number of hidden layers during training. Regarding time-series data, DNNs are especially difficult to train. This is because the degree of complexity in time-series data differ between applications. Different applications are subjected to diverse amounts of sensor noise and a varying number of operating conditions. DNNs consist of vast amounts of hyper-parameters, and hence, require precise tuning for a specific time-series application. Consequently, the most difficult hyper-parameter to tune in DNNs for time-series data is the number of hidden layers that reflects the total number of parameters, namely, weights and biases. DNNs with few parameters are only able to model time-series data with low complexity and vice versa.

In this paper, we adopt the idea of skip connections and apply them to a DNN suitable for time-series data. The DNN consists of two long-short term memory (LSTM) layers, two feed-forward neural network (FNN) layers, one dropout layer, and a fully connected output layer. The combination of LSTM layers and FNN layers in DNN structures has shown outstanding performance in recent RUL prediction research studies [9], [10]. However, skip connections are applied such that the network has the possibility to skip both the second LSTM layer and the second FNN layer. We name the network SkipRnet and validate it on real operational RTF data collected from a marine diesel engine. As in [3], two replicated autonomous ferry crossing operations are used as two different engine load profiles. RTF data of two independent fault-types are collected from both profiles. RTF data from the first profile is used for training and validation. Hence, the proposed data augmentation technique is used to construct more RTF data from the first profile. The goal of the data augmentation is to increase the generalization power of the SkipRnet towards the RTF data in the second profile, which is used as a real-time test. High generalization power towards engine load profiles that the SkipRnet has never seen before is extremely important if the SkipRnet is to be employed in future PHM systems for autonomous ferries to provide real-time RUL predictions. This study's main contributions are as follows:

- Data augmentation increases the generalization power of DNNs constructed for RUL predictions.
- Due to the presence of skip connections, the SkipRnet is able to tackle a wider range of complexities in new field data compared to traditional DNNs.
- In the real-time test, the SkipRnet predicts the RUL of two independent fault-types in an engine load profile that it has never seen before with high accuracy.

The rest of this paper is organized as follows. Section II introduces the latest studies on data-driven fault prognostics. Section III introduces the essential background theory on FNN, LSTM, and the SkipRnet. The case study is described in section IV, while the experimental results and discussions are presented in section V. Finally, section VI concludes the paper and indicates objectives for further work.

II. Literature review

In the latest RUL prediction research studies, DNNs are proposed with a fixed number of hidden layers. The hidden units belonging to each hidden layer are usually tuned on cross-validation data, which is a portion of the training data.

In [9], Ellefsen et al. proposed a semi-supervised DNN for RUL prediction purposes. The DNN consisted of one restricted Boltzmann machine layer, two LSTM layers, one FNN layer, and an output layer. A genetic algorithm was also proposed to tune a chosen search space of hyper-parameters, including learning rate, the number of hidden units, activation functions, etc. Thus, the number of hidden layers was predetermined before any tuning was performed. In [10], Miao et al. proposed a dual-task DNN for joint learning of degradation assessment

and RUL prediction. The proposed DNN included two LSTM layers, one FNN layer as the classification sub-network, and three FNN layers as the regression sub-network. A 5-fold cross-validation procedure was performed to optimize the number of hidden units in each layer. Hence, as in [9], the number of hidden layers was predetermined in this study. Xia et al. proposed a two-stage DNN approach to predict the RUL of bearings in [11]. First, a denoising autoencoder was used to cluster bearing signals into diverse degradation levels. Then, shallow DNNs were constructed for each health level to perform regression. Finally, the regression results were smoothed to achieve the overall RUL prediction. In this study, both the number of hidden layers and the hidden units were predetermined based on experience.

Consequently, in [9]–[11], the trained DNNs assume that new field data will have similar distributions and complexity as the training data. However, autonomous ferries operate under unpredictable environmental conditions, and hence, the marine diesel engine has to operate under various operating conditions. Thus, the assumption that the training data and the future field data follow similar distributions and complexity is improbable to hold. One approach to address this issue is transfer learning (TL). TL involves techniques for transferring knowledge learned in one domain (training data) to a new domain (new field data). In [15], Sun et al. proposed a deep TL network that included three transfer strategies. Weights, weights updates, and hidden features were used to transfer a sparse autoencoder (SAE) to a new domain. The SAE showed improved RUL prediction performance. In [16], de Oliveira da Costa et al. used a domain adversarial neural network approach to transfer knowledge of RUL predictions to a new domain that only contained sensor information. As in [15], the proposed approach showed improved RUL prediction performance compared to DNNs only trained on the training data.

Another interesting approach to tackle the problem of divergent distribution and complexity of the training data and the future field data is the utilization of skip connections. When the SkipRnet is trained and employed to predict the RUL on new field data, it has the potential to make use of different paths with different numbers of parameters. These path alternatives might handle the various operating conditions the marine diesel engine confronts.

III. Theoretical foundation

This section introduces the necessary theoretical foundation. First, it briefly presents both FNN and LSTM. Then it describes the SkipRnet.

A. Feed-forward neural network

An FNN was the first and most basic type of artificial neural network developed. It is also referred to as a Multilayer Perceptron if it's structured with at least an input layer, a hidden layer, and an output layer. An FNN is fully connected such that each unit in one layer has a direct weight connection to all units of the subsequent layer. Normally, FNNs learn in a supervised manner by mapping an input $\boldsymbol{x}$ to a target y. Thus, an FNN describes a mapping $\boldsymbol{y} = f(\boldsymbol{x};\boldsymbol{\theta})$ and uses the back-propagation algorithm [17] to learn the parameters $\boldsymbol{\theta}$ which consist of biases and weights. The output of unit k of layer l is:

$$a_k^l = \phi(z_k^l)\,, \tag{1}$$

where ϕ is a non-linear activation function and the function argument is:

$$z_k^l = b_k^l + \sum_j w_{jk}^l a_j^{l-1} \tag{2}$$

where a_j^{l-1} is the output from unit j in the previous layer $l-1$, w_{jk}^l are weight factors, and b_k^l is the bias. In the first hidden layer $l = 1$, however, the input is $a_i^0 = x_i$, where x_i, $i = 1 \ldots n$ are the inputs to the FNN.

B. Long-short term memory

Hochreiter and Schmidhuber developed the original LSTM in the 1990s [18]. Since then, improvements upon the original LSTM have been implemented [19]–[21] to form what scholars generally call the vanilla LSTM. Most DL programming libraries use the vanilla LSTM with no peephole connections because it offers fast training time on graphics processing units (GPUs). This LSTM variant introduces a memory cell that enables the LSTM to maintain its state over time. In other words, it is able to learn both short-term and long-term dependencies. This is its main advantage compared to traditional recurrent neural networks.

Three non-linear gating units control and protect the state of the memory cell $\boldsymbol{S}_t$ [22]:

$$\boldsymbol{f}_t = \phi(\boldsymbol{W}_f\,\boldsymbol{x}_t + \boldsymbol{R}_f\,\boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \tag{3}$$

$$\boldsymbol{i}_t = \phi(\boldsymbol{W}_i\,\boldsymbol{x}_t + \boldsymbol{R}_i\,\boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \tag{4}$$

$$\boldsymbol{o}_t = \phi(\boldsymbol{W}_o\,\boldsymbol{x}_t + \boldsymbol{R}_o\,\boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \tag{5}$$

where ϕ is the gate activation function, which applies a sigmoid function to scale the values between 0 and 1, $\boldsymbol{W}$ is the input weight, $\boldsymbol{R}$ is the recurrent weight, and $\boldsymbol{b}$ is the bias weight. A tanh activation function creates the new candidate state values $\tilde{\boldsymbol{S}}_t$:

$$\tilde{\boldsymbol{S}}_t = tanh(\boldsymbol{W}_s\,\boldsymbol{x}_t + \boldsymbol{R}_s\,\boldsymbol{h}_{t-1} + \boldsymbol{b}_s) \tag{6}$$

The previous cell state, $\boldsymbol{S}_{t-1}$, and $\tilde{\boldsymbol{S}}_t$ create the new cell state, $\boldsymbol{S}_t$:

$$\boldsymbol{S}_t = \boldsymbol{f}_t \otimes \boldsymbol{S}_{t-1} + \boldsymbol{i}_t \otimes \tilde{\boldsymbol{S}}_t \tag{7}$$

where $\otimes$ specifies element-wise multiplication of two vectors. In this way, $\boldsymbol{f}_t$ determines which historical information should be deleted and $\boldsymbol{i}_t$ decides what new information the memory cell will consider relevant and store. At last, $\boldsymbol{o}_t$ decides which information of $\boldsymbol{S}_t$ the memory cell will output:

$$\boldsymbol{h}_t = \boldsymbol{o}_t \otimes tanh\,(\boldsymbol{S}_t) \tag{8}$$

By means of Eq. 3 - 8, the LSTM has the capability to delete and insert relevant information to $\boldsymbol{S}_t$. This feature makes it well-suited to learn temporal dependencies in time-series data. The LSTM is also trained by the back-propagation algorithm.

C. SkipRnet

As they do in the latest RUL prediction research studies [9], [10], LSTMs and FNNs will act as the main building blocks of the SkipRnet. The LSTM layers are used to learn temporal and long-term dependencies within the features of degradation data. The FNN layers are then used to map all extracted features before a dropout layer is used to reduce overfitting. Dropout [23] randomly drops units of the dropout layer during training. This forces the SkipRnet to learn to make generalized representations of the input data. Generalized representations amplifies the feature extraction capability of the SkipRnet since it prevents it from extracting the same degradation features over and over. The last layer consists of a time distributed and fully connected output layer with one unit. This layer handles error calculations and perform RUL predictions. The mean squared error is utilized as the loss function.

As seen in Figure 1, the SkipRnet has the ability to skip both the second LSTM layer and the second FNN layer during the training procedure. This results in four different paths with differing numbers of parameters. In other words, the SkipRnet can be considered as an accumulation of four different DNNs, and hence, for different time steps in the training data, the SkipRnet will be trained at different rates based on how the error flows backward in the four DNNs. Therefore, the SkipRnet should be able to handle time-series data in a wide range of complexities. Similar to ResNets, the skip connections in the SkipRnet use element-wise addition to combine the activations of two layers. It's worth noting that skip connections can as well be applied to other combinations of LSTM layers and FNN layers than what the SkipRnet uses.

In the following case study, the four different paths will be used as the baseline DNNs for comparison. That is, L1F1, L1F2, L2F1, and L2F2, where L and F represent the number of LSTM layers and FNN layers, respectively. Since the SkipRnet is trained as an ensemble of the four different DNNs, it should provide as good as accuracy as any of the different DNNs in all scenarios.

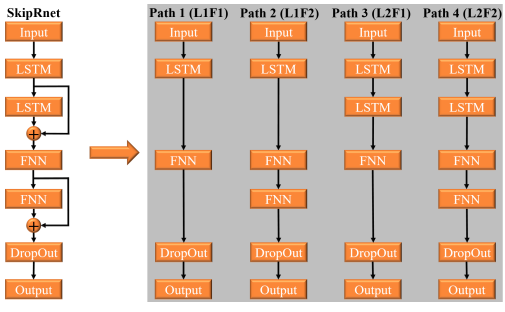


Fig. 1. The SkipRnet and its four different paths, L1F1, L1F2, L2F1, and L2F2, where L and F represent the number of LSTM layers and FNN layers, respectively.

Fig. 2. The marine diesel engine included in the hybrid power lab.

IV. Case Study

The following case study, uses Microsoft Windows 10, Java 8, deeplearning4j (DL4J) version 1.0.0-beta4 [24] as the DL library, and NVIDIA GeForce GTX 1060 6 GB as the GPU.

A. Data sets

A hybrid power lab, which is designed to research ship autonomy, is used to collect the data sets. The lab was established by the Department of Ocean Operations and Civil Engineering at the Norwegian University of Science and Technology in Aalesund. The lab includes a marine diesel engine, a marine battery system, and a marine automation system to control the process. To simulate load alterations in the system, the produced power is supplied back to the power grid. The diesel engine is shown in Figure 2.

Two engine load profiles have been used during the data collection process. As seen in Figure 3, the profiles aim to replicate two different environmental conditions the autonomous ferry may encounter during a ferry crossing on the west coast of Norway. To obtain degradation data, two typical and independent fault-types associated with the marine diesel engine were provoked during the data collection process of both profiles. The first fault-type is clogging of the air filter, while the second fault-type is a malfunction of the turbocharger.

In our previous work, a fault-type independent spectral anomaly detection algorithm was proposed to detect the fault

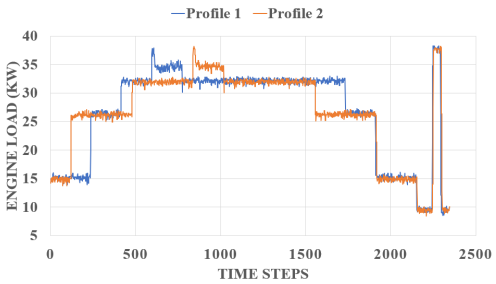


Fig. 3. Profile 1 vs. profile 2.

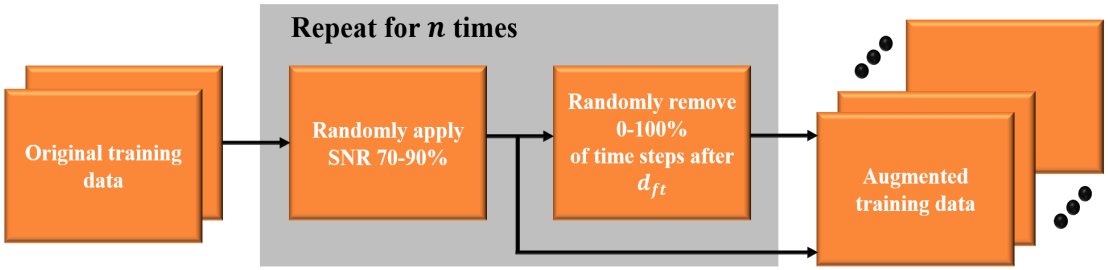


Fig. 4. The proposed data augmentation technique for run-to-failure time-series data.

TABLE I
THE FOUR ORIGINAL RTF DATA SETS COLLECTED FROM THE MARINE DIESEL ENGINE.

Data set	Profile	Usage	d_{ft}	Last RUL target	Time steps
Air filter	1	Train/val	1,660	0	2,346
Turbo	1	Train/val	1,347	0	2,346
Air filter	2	Test	1,483	106	2,240
Turbo	2	Test	1,347	490	1,856

time step d_{ft} of both fault-types in both profiles [3]. The detected d_{ft} is used in this case study to automatically construct RUL targets based on the piece-wise linear degradation model, an approach heavily validated in [7]. The air filter fault and the turbo fault in profile 1 are used as the training set for the SkipRnet. The degradation in the training set grows in magnitude until failure. Consequently, the last RUL target = 0. Profile 2 is subjected to different engine loads, and hence, it will be used as the test set. However, the degradation in the test set should end sometime before failure in order to verify that the SkipRnet is able to predict the RUL. Accordingly, a random amount of time steps is removed in both the air filter fault and the turbo fault in profile 2. Table I summarizes the data sets used in this case study. All data sets include 47 input features, e.g., engine load, engine speed, flow, pressure, and temperature measurements. See [3] for a detailed description of the two fault-types and see [25] for analysis of the input features.

It is worth noting that real-life RTF data sets are normally accumulated and collected through months, or perhaps even years. In this case study, however, the RTF data sets are collected more rapidly due to time constraints. Even though the collected RTF data sets only consist of 2,346 time steps, the real degradation patterns are assumed to remain. One time step equals 0.5 seconds.

B. Data augmentation and normalization

Each feature in the training set is scaled with zero mean and unit variance (z-score) normalization:

$$\hat{x}_n = \frac{x_n - \mu}{\sigma} \tag{9}$$

where x_n is the feature measurement, $n = 1, 2, \ldots, 47$, and μ and σ is the mean and standard deviation of that feature, respectively. The normalization statistics obtained from the training set will also be applied to the test set.

It is well-known that DNNs have the property that if you feed them more data they get better and better [26]. This fact also holds for DNNs constructed to provide RUL predictions. Therefore, scholars have usually used the publicly accessible Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) data set, which consists of numerous simulated RTF data sets of aircraft gas turbine engines, to train and validate their proposed DNNs. The C-MAPSS data set is created and provided by NASA [27] and acknowledged as the benchmark data set for prognostics algorithms that address DNNs within the PHM research field. In real-world applications, however, RTF data might be more time-consuming and difficult to acquire. In such applications, data augmentation techniques come in handy.

A limitation of DNNs today is the danger that they will learn only exactly what we ask them to learn. For instance, in this case study, we will only use RTF data from profile 1 as the training set. So, the danger is that the SkipRnet will only learn statistics from profile 1 and not be able to generalize to profile 2. Thus, as seen in Figure 4, this paper proposes a data augmentation technique for RTF data to increase the generalization power of the SkipRnet. First, random white Gaussian noise, g, is applied to each $\hat{x}_n$ in the original training set with a random signal-to-noise-ratio (SNR) between 70 and 90%:

$$SNR(\%) = \frac{P_{signal}}{P_{noise}} \cdot 100 \tag{10}$$

where P_{signal} and P_{noise} are the average power of the signal and the noise, respectively, and calculated as follows:

$$P_{signal} = \frac{1}{T_t} \sum_{t=1}^{T_t} \left(\sqrt{\frac{1}{n}\left(\hat{x}_1^2 + \cdots + \hat{x}_n^2\right)} \right) \tag{11}$$

$$P_{noise} = \frac{1}{T_t} \sum_{t=1}^{T_t} \left(\sqrt{\frac{1}{n}\left((\hat{x}_1 + g)^2 + \cdots + (\hat{x}_n + g)^2\right)} \right) \tag{12}$$

where T_t is the total time step length of the original training set and n is the number of input features. The resulting noisy data set will exhibit similar statistics to profile 1, but differ based on the SNR. The aim is to increase the range of statistics

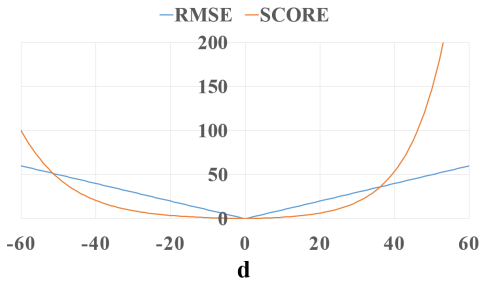


Fig. 5. RMSE vs. SCORE

that the SkipRnet will learn in the training procedure. Next, following [28], a random amount of time steps are removed after d_{ft} to also include some time-series that will end some time prior to failure. Thus, the SkipRnet is forced to learn distributions that are more similar to a real-time PHM system, where the main goal is to accurately predict the available time before operational failure. The proposed data augmentation technique is repeated for 0, 10, 20, 30, 40, and 50 times for each fault-type in the training set. This results in six different scenarios of 0, 20, 40, 60, 80, and 100 augmented training data sets.

C. Performance indicators

In this case study, the root mean square error (RMSE) is used as one of the performance indicators. The training procedure can be considered as a regression task since the goal is to train the SkipRnet to predict the constructed RUL targets with high accuracy at each time step. The RMSE is defined as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} d_i^2} \tag{13}$$

where n is the total number of constructed RUL targets and $d_i = RUL_{predicted,i} - RUL_{target,i}$. Additionally, the maintenance scoring function (SCORE), provided in [27], is used as a performance indicator for the last RUL targets in the test set:

$$SCORE = \begin{cases} \sum_{j=1}^{m} e^{-(\frac{d_j}{13})} - 1, & \text{for} \quad d_j < 0 \\ \sum_{j=1}^{m} e^{(\frac{d_j}{10})} - 1, & \text{for} \quad d_j \geq 0 \end{cases} \tag{14}$$

where m is the total number of last RUL targets in the test set and $d_j = RUL_{predicted,j} - RUL_{target\,last,j}$. In SCORE, the punishment for late RUL predictions, when $d_j > 0$, is greater than early RUL predictions, when $d_j < 0$. This indication is especially suited for the SkipRnet as late RUL predictions could lead to a potential disaster for autonomous ferries. Late RUL predictions are vulnerable to engine failure since maintenance operations would be scheduled too late. Early RUL predictions, however, pose less risk of engine failure as maintenance operations would be scheduled too early.

TABLE II
HYPER-PARAMETERS.

Hyper-parameter	Method/Value/Values
Activation function in FNN layers	ReLU
Activation function in LSTM layers	tanh
Dropout	0.5
Hidden units in all hidden layers	64, 128, 192
Learning rate first LSTM layer	$5 \cdot 10^{-4}$
Learning rate remaining layers	$1 \cdot 10^{-4}$
$l2$ regularization	$1 \cdot 10^{-5}$
Mini-batch size	5
Optimization algorithm	SGD
Optimizer	Adam
Seed	12345
Weight initialization	Xavier

The main objective for both performance indicators is to reach the lowest value possible, namely, when $d_i = 0$ and $d_j = 0$. The RMSE and the SCORE are illustrated in Figure 5.

D. Network configuration and training

DNNs introduce a big search space of hyper-parameters, which can be laborious to optimize in the training process. Thankfully, recent RUL prediction research studies [7], [9], [10] can be used as a guide to determine which hyper-parameters to include and which to omit. Similar to [7], the rectified linear unit (ReLU) activation function [29] is used in FNN layers and the tanh activation function is used in LSTM layers. Additionally, to better maintain low-level degradation features, the learning rate in the first LSTM layer is a half order of magnitude higher than the learning rate in the remaining layers. The $l2$ regularization coefficient is $1 \cdot 10^{-5}$. As recommended in [23], the dropout retaining probability is 0.5. In DL4J, time-series have three dimensions: [numExamples, inputSize, timeSeriesLength], where numExamples is the number of time-series included in a mini-batch, inputSize is the number of input features, and timeSeriesLength is the total time step length in a mini-batch. The mini-batch size is selected to be five RTF data sets for all augmented data set scenarios, except the first scenario. There are only two RTF data sets in the original training data, so no mini-batch is used in the first scenario. As in [7], [9], [10], stochastic gradient descent (SGD) is used as the optimization algorithm together with adaptive moment estimation (Adam) as the learning rate method [30]. Xavier weight initialization [31] is used in all layers. To ensure the training results are reproducible, the seed is fixed to a value of 12345 in all experiments. The hyper-parameters are given in Table II.

In this case study, the most important hyper-parameter is considered to be the total number of parameters, which relates to the number of hidden units in each layer. Thus, the number of hidden units will be tuned through a 7-fold cross-validation procedure for each augmented data set scenario. Each scenario is divided into 80% training and 20% cross-validation, randomly. To prevent overfitting, early stopping is used to monitor the RMSE performance on the cross-validation set for each fold. If the number of epochs with no reduction on the RMSE gets greater than four, the training process is aborted. Then, the SkipRnet, in the epoch with the lowest RMSE on the cross-validation set, is saved. In the

TABLE III
THE AVERAGE RMSE OF A 7-FOLD CROSS-VALIDATION PROCEDURE FOR EACH AUGMENTED DATA SET SCENARIO.

Configuration	Total params	Augmented data sets	RMSE train	RMSE cross	Avg. epoch time
SkipRnet 64	73,985	0	29.47	29.47	0.64
		20	31.45	34.64	2.86
		40	33.37	37.20	5.28
		60	34.84	33.58	7.59
		80	34.52	31.97	10.21
		100	41.84	31.01	12.53
		Avg.	34.25	32.98	**6.52**
SkipRnet 128	270,849	0	31.26	31.26	0.69
		20	26.75	28.83	2.94
		40	29.46	29.13	5.48
		60	31.66	27.22	7.85
		80	34.42	27.70	10.66
		100	39.19	31.55	13.02
		Avg.	**32.12**	**29.28**	6.77
SkipRnet 192	590,593	0	57.10	57.10	0.75
		20	30.00	30.20	3.11
		40	35.63	27.73	5.92
		60	33.81	31.42	8.54
		80	52.31	32.85	11.62
		100	43.70	37.10	14.32
		Avg.	42.09	36.07	7.38

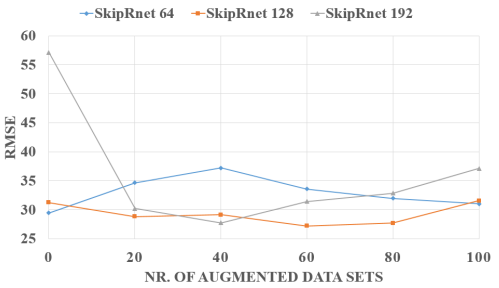


Fig. 6. The average cross-validation RMSE accuracy of all folds for each augmented data set scenario.

end, the average cross-validation RMSE of all folds for each augmented data set scenario is calculated.

V. EXPERIMENTAL RESULTS AND DISCUSSIONS

The objective of this paper is to explore whether data augmentation and skip connections are advantageous for fault prognostics in autonomous ferries. First, the average RMSE performance for all augmented data set scenarios was compared to the SkipRnet, with different numbers of total parameters. Then, the most robust configuration of the SkipRnet and its four different paths, L1F1, L1F2, L2F1, and L2F2, are compared using both RMSE and SCORE on the test set. Finally, the SkipRnet trained with and without data augmentation is illustrated in a real-time test.

A. Cross-validation

The goal of the cross-validation is to acquire the most robust configuration, in terms of total parameters, of the SkipRnet. That is, to achieve the configuration that best reflects the degree of complexity in the cross-validation sets in all augmented data set scenarios. The first scenario includes zero augmented data sets, and hence, the SkipRnet is only trained and validated on the original training set. Thus, the first scenario is assumed to exhibit the lowest degree of complexity of the six scenarios. In contrast, the scenario with 100 augmented data sets is assumed to exhibit the highest degree of complexity.

As seen in Table III and Figure 6, the SkipRnet with 64 hidden units in all hidden layers includes too few parameters to model the augmented data set scenarios. Instead, it provides the lowest cross-validation RMSE on the original training set. When it comes to the SkipRnet with 192 hidden units, the total number of parameters is increased almost eight-fold. This number of parameters is clearly too big compared to the number of examples in the original training set. That said, it provides a major decrease in cross-validation RMSE in all augmented data set scenarios, where both the number of examples and the complexity has increased. As seen in Table III, the SkipRnet with 128 hidden units provides the lowest average RMSE for all augmented data sets scenarios. Therefore, this SkipRnet is considered the most robust configuration and will be further used in the test on profile 2.

The training time should also be taken into consideration in a lot of applications. As seen in Table III, the average training time per epoch, which is stated in seconds, is very similar between all three configurations of the SkipRnet. As a consequence, this comparison neglects the training time.

B. Real-time test

Autonomous ferries will be subjected to unpredictable environmental conditions. Ergo, the marine diesel engine is prone to various operating conditions. This is why the SkipRnet needs to provide high generalization power towards engine load profiles that it has not seen before, which is profile 2 in this study. Therefore, together the air filter fault and the turbo fault in profile 2 comprise the test set.

The SkipRnet can be considered as an ensemble of four independent DNNs, L1F1, L1F2, L2F1, and L2F2, as referred to in Figure 1. As a result, the SkipRnet should provide better or as good RMSE performance as the four DNNs in all augmented data set scenarios. In order to verify this, each baseline DNN is also trained on each augmented data set scenario through the above-mentioned 7-fold cross-validation procedure.

In this comparison, the trained SkipRnet and baseline DNNs for each fold in each augmented data set scenario are employed to predict the RUL at each time step on the test set. In other words, this comparison can be considered as a real-time test because this is how the networks would potentially be employed in a real-life data-driven PHM system for autonomous ferries. Both RMSE and the SCORE are used as the performance indicators. Table IV, however, shows the average RMSE and SCORE of the seven SkipRnets, L1F1s, L1F2s, L2F1s, and L2F2s in each augmented data set scenario. As seen in Figure 7, the SkipRnet provides the lowest RMSE for each augmented data set scenario. As expected, L1F1 provides the worst overall RMSE due to having the lowest number of parameters. Interestingly, L2F2 provides worse overall RMSE compared to the SkipRnet, even though L2F2 has the same

TABLE IV
THE AVERAGE RMSE AND SCORE PERFORMANCE ON THE TEST SET.

Network	Total params	Augmented data sets	RMSE test	SCORE
L1F1	122,753	0	315.95	8,192.14
		20	285.09	67.63
		40	245.09	218.73
		60	254.61	731.32
		80	279.81	600.80
		100	256.86	226.77
L1F2	139,265	0	213.26	1,796.67
		20	194.51	18,086.95
		40	207.36	5,444.71
		60	194.55	3,023.56
		80	189.71	6,939.86
		100	181.01	177.45
L2F1	254,337	0	139.86	5.1
		20	122.14	320.60
		40	132.82	2,845.32
		60	156.84	69.38
		80	145.94	72.31
		100	135.03	88.35
L2F2	270,849	0	182.37	420.17
		20	218.03	34.75
		40	197.05	201.29
		60	177.41	16.90
		80	170.60	205.12
		100	158.41	66.65
SkipRnet	270,849	0	137.89	8.59
		20	85.76	9.92
		40	106.11	17.23
		60	72.98	70.28
		80	91.85	59.50
		100	97.86	397.26

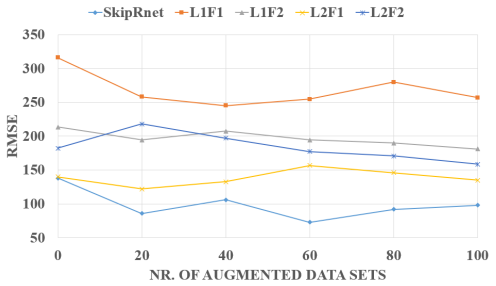


Fig. 7. The average RMSE on the test set for each augmented data set scenario.

number of parameters. A logical explanation for these findings is the advantage of the skip connections. For each time step in the test set, the SkipRnet has the ability to utilize the strengths and reduce the weaknesses of four DNNs. In other words, for each time step, the SkipRnet has the ability to utilize different numbers of parameters in the range between 122,753 and 270,849. Therefore, the SkipRnet is able to handle a wider range of complexities in new field data compared to DNNs without skip connections.

In addition to the RMSE, the SCORE is also important to consider in real-life data-driven PHM systems suitable for autonomous ferries. A reliable and low SCORE performance close to the end of the marine diesel engine's lifetime has great significance, as this period is critical in order to schedule maintenance operations. As seen in Figure 8, the SkipRnet provides satisfactory SCORE performance on the test set when

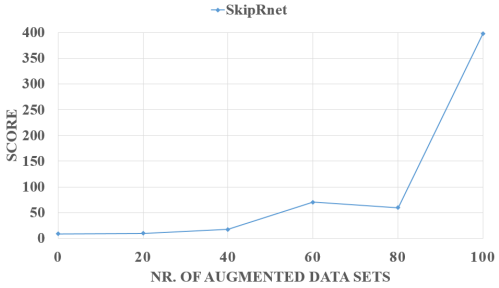


Fig. 8. The average SCORE on the test set for each augmented data set scenario.

trained on 0, 20, and 40 augmented data sets. However, as seen in Table IV, the SkipRnet provides the lowest RMSE on the test set when trained on 20 and 60 augmented data sets. Consequently, the SkipRnet provides the best overall RUL performance on the test set when trained on 20 augmented data sets.

Figure 9 compares the RUL predictions on both the air filter fault and the turbo fault in the test set when the SkipRnet is trained on 20 augmented data sets and the original training set. This comparison proves the advantage of the proposed data augmentation technique. It clearly increases the generalization power of the SkipRnet toward an engine profile it has never seen before. Additionally, the SkipRnet provides high RUL prediction performance, especially close to the end of the engine's lifetime, for both fault-types. However, the predictions are kind of noisy, but this is expected due to the drastic changes in engine loads. As a consequence, confidence bounds have to be incorporated to increase the reliability of the RUL predictions in a real-life data-driven PHM system. Maintenance decisions based on prognostics information should be anchored in confidence bounds rather than a particular RUL prediction [32].

VI. CONCLUSION AND FUTURE WORK

This paper has analyzed and proposed a novel data augmentation technique and the SkipRnet for fault prognostics of the marine diesel engine in autonomous ferries. In order to do so, RTF data of two fault-types in two different engine load profiles have been used during the experiments. RTF data in profile 1 is used for training and validation, while RTF data in profile 2 is used for real-time testing. Hence, the data augmentation technique is used to construct six different augmented data set scenarios based on the first profile. High generalization power towards engine load profiles that the SkipRnet has never seen before is of high value since the marine diesel engine will be subjected to unforeseen operating conditions due to variable environmental conditions the autonomous ferry will encounter.

In the validation of the SkipRnet, the importance of tuning the total number of hyper-parameters has been shown. The SkipRnet can be considered as an accumulation of four independent DNNs due to its skip connection. Thus, when included in future PHM systems for autonomous ferries, the

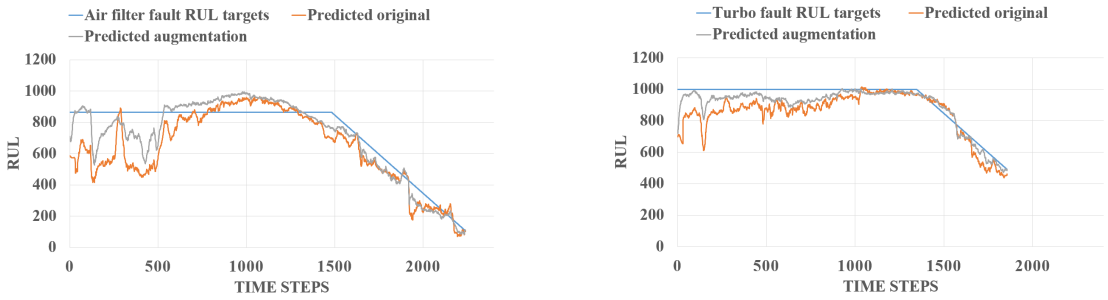


Fig. 9. The prediction results on the air filter fault and the turbo fault in the test set when the SkipRnet is trained on 20 augmented data sets and the original training set.

SkipRnet has the ability to tackle a wider range of complexities in new field data compared to traditional DNNs without skip connections. Additionally, the SkipRnet improved its generalization power when trained on 20 augmented data sets towards engine load profiles it had never seen before. The advantage of data augmentation and skip connections is clear.

Accurate and reliable fault prognostics rely on the accessibility of RTF data. Manufacturers and shipowners need to start saving and sharing their RTF data in order to gain the true benefits of data-driven PHM systems. Based on the findings in this paper, it is not an understatement to claim that the more data you feed to DNNs, constructed for fault prognostics, the better they become at providing RUL predictions.

Future data-driven PHM systems need to be included in the building and design phase of autonomous ferries. It will be more difficult and time-consuming to install such systems on an already operational autonomous ferry due to the diversity of equipment and system manufacturers operating today. Our future work will address these factors.

ACKNOWLEDGMENT

This work was supported by the Norwegian University of Science and Technology within the Department of Ocean Operations and Civil Engineering under project no. 90329106. The authors would like to thank Digital Twins For Vessel Life Cycle Service and the Research Council of Norway, grant no. 280703.

REFERENCES

[1] O. Levander, "Autonomous ships on the high seas," *IEEE Spectrum*, vol. 54, no. 2, pp. 26–31, 2017.

[2] A. L. Ellefsen, V. Æsøy, S. Ushakov, and H. Zhang, "A comprehensive survey of prognostics and health management based on deep learning for autonomous ships," *IEEE Transactions on Reliability*, vol. 68, no. 2, pp. 720–740, 2019.

[3] A. L. Ellefsen, P. Han, X. Cheng, F. T. Holmeset, V. Æsøy, and H. Zhang, "Online fault detection in autonomous ferries: Using fault-type independent spectral anomaly detection," *Under review in IEEE Transactions on Instrumentation and Measurement*, pp. 1–1, 2020.

[4] P. W. Kalgren, C. S. Byington, M. J. Roemer, and M. J. Watson, "Defining phm, a lexical evolution of maintenance and logistics," in *2006 IEEE Autotestcon*, Sept 2006, pp. 353–358.

[5] B. Sun, S. Zeng, R. Kang, and M. G. Pecht, "Benefits and challenges of system prognostics," *IEEE Transactions on Reliability*, vol. 61, no. 2, pp. 323–335, 2012.

[6] O. Bektas, J. A. Jones, S. Sankararaman, I. Roychoudhury, and K. Goebel, "A neural network filtering approach for similarity-based remaining useful life estimation," *The International Journal of Advanced Manufacturing Technology*, vol. 101, no. 1-4, pp. 87–103, 2019.

[7] 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.

[8] Y. Lei, N. Li, S. Gontarz, J. Lin, S. Radkowski, and J. Dybala, "A model-based method for remaining useful life prediction of machinery," *IEEE Transactions on Reliability*, vol. 65, no. 3, pp. 1314–1326, 2016.

[9] 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.

[10] H. Miao, B. Li, C. Sun, and J. Liu, "Joint learning of degradation assessment and rul prediction for aeroengines via dual-task deep lstm networks," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 9, pp. 5023–5032, Sep. 2019.

[11] M. Xia, T. Li, T. Shu, J. Wan, C. W. de Silva, and Z. Wang, "A two-stage approach for the remaining useful life prediction of bearings using deep neural networks," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3703–3711, June 2019.

[12] A. S. Zymaris, Ø. Å. Alnes, K. E. Knutsen, and N. M. Kakalis, "Towards a model-based condition assessment of complex marine machinery systems using systems engineering," in *Proc. 3rd Eur. Conf. Prognostics Health Manage. Soc.*, 2016, pp. 1–15.

[13] E. Bjørlykhaug and O. Egeland, "Vision system for quality assessment of robotic cleaning of fish processing plants using cnn," *IEEE Access*, vol. 7, pp. 71 675–71 685, 2019.

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

[15] 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, April 2019.

[16] P. R. de Oliveira da Costa, A. Akçay, Y. Zhang, and U. Kaymak, "Remaining useful lifetime prediction via deep domain adaptation," *Reliability Engineering & System Safety*, vol. 195, p. 106682, 2020.

[17] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 9–48.

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

[19] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with lstm," in *9th International Conference on Artificial Neural Networks ICANN 99*, vol. 2, Sept 1999, pp. 850–855.

[20] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

[21] F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count," in *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN)*, vol. 3. IEEE, 2000, pp. 189–194.

[22] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2017.
[23] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.
[24] "Eclipse deeplearning4j development team, deeplearning4j: Open-source distributed deep learning for the jvm," *Apache Software Foundation License 2.0, http://deeplearning4j.org*, 2020.
[25] X. Cheng, A. L. Ellefsen, F. T. Holmeset, G. Li, H. Zhang, and S. Chen, "A step-wise feature selection scheme for a prognostics and health management system in autonomous ferry crossing operation," in *2019 IEEE International Conference on Mechatronics and Automation (ICMA)*, Aug 2019, pp. 1877–1882.
[26] N. Jones, "Computer science: The learning machines," *Nature*, vol. 505, pp. 146–148, 2014.
[27] 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, Oct 2008, pp. 1–9.
[28] L. Jayasinghe, T. Samarasinghe, C. Yuen, J. C. N. Low, and S. S. Ge, "Temporal convolutional memory networks for remaining useful life estimation of industrial machinery," *arXiv preprint arXiv:1810.05644*, 2018.
[29] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, vol. 15, 2011, pp. 315–323.
[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[31] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.
[32] J. Z. 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.