PhD-FSTM-2022-062
The Faculty of Sciences, Technology and Medecine

Dissertation

Presented on 27^{th} April 2022 in Luxembourg

to obtain the degree of

Docteur de l'Université du Luxembourg en Informatique

by

Paul-Lou BENEDICK
Born on 19^{th} January 1994 in Sarrebourg (France)

TOWARDS A UNIFIED AND ROBUST DATA-DRIVEN APPROACH. A DIGITAL TRANSFORMATION OF PRODUCTION PLANTS IN THE AGE OF INDUSTRY 4.0

Dissertation defence committee

Dr. Nicolas NAVET, Chairman
Professor, University of Luxembourg, Luxembourg

Dr. Jean-Philippe GEORGES, Vice-Chairman
Professor, University of Lorraine, Nancy, France

Dr. Pierre-Alain MULLER,
Professor, University of Haute-Alsace, Mulhouse-Colmar, France

Dr. Jérémy ROBERT,
Digital Technology Expert, Cebi Luxembourg S.A., Steinsel, Luxembourg

Dr. Yves LE TRAON, Dissertation Supervisor
Professor, University of Luxembourg, Luxembourg

Abstract

Nowadays, industrial companies are engaging their global transformation toward the fourth industrial revolution, often called Industry 4.0. The main objective is to increase the Overall Equipment Effectiveness (OEE), by collecting, storing and analyzing production data. At its heart, the industry 4.0 is a data-driven paradigm: it relies on the production data / data measurements produced by heterogeneous industrial machines to maximize the OEEs. However, the heterogeneity of industrial equipment requires a common approach to collect and treat data. Several challenges have to be tackled to propose a unified data-driven approach to rely on, from the low-layers data collection on the machine production lines using Operational Technologies (OT), to the processing and more importantly the analysis of the data using Information Technologies (IT). This is all the more important for companies having decades of existence – *as Cebi Luxembourg S.A., our partner in a Research, Development and Innovation project subsidised by the ministry of the Economy in Luxembourg* – to upgrade their on-site technologies and move towards data-driven and more efficient business models. For the past decade, Artificial Intelligence (AI) has become a prominent technology for decision making industrial data analysis, thanks to the huge amount of (sensors-based) univariate time-series generated by factory equipments in production plants. However, a high data volume alone is not sufficient for AI to work properly and to make optimal decisions. This also requires a good data quality. Indeed, good theoretical performance and high accuracy can be obtained when trained and tested in isolation, but AI models may still provide degraded performance in real/industrial conditions where data is subject to measurement noises and errors in the collection process.

As such, defining a data-driven solution for industrial plants faces two significant challenges:

- Industrial production systems are vertically-oriented closed systems that make difficult their communication and their cooperation with each other,

and intrinsically the data collection.

- Industrial actors used to rely on deterministic processes for their predictability. Introducing AI - *that can be classified as stochastic* - in the industry requires a full understanding of the potential deviation of the models in order to be aware of their domain of validity.

This dissertation proposes a unified strategy for digitizing industrial systems and introduces methods for evaluating the performance and the robustness of AI models that can be used in such data-driven production plants.

In the first part of the dissertation, we propose a three-steps strategy to digitize industrial systems, called TRIDENT, that enables industrial actors to implement data collection on production lines, and *in fine* to monitor in real-time the production plant. Such strategy has been implemented and evaluated on a pilot case-study at *Cebi Luxembourg S.A.* production plants. Three protocols, Open Platform Communications - Unified Architecture (OPC-UA), Message Queuing Telemetry Transport (MQTT) and Open Messaging Interface (O-MI)/Open-Data Format (O-DF), are used for investigating their impact on the real-time performance. The results show that, even if these protocols have some disparity in terms of performance, they are suitable for an industrial deployment. This strategy has now been extended and implemented by our partner - *Cebi Luxembourg S.A* - in its production environment.

In the second part of the dissertation, we aim at investigating the robustness of AI models in industrial settings. We then propose a systematic approach to evaluate the robustness under perturbations. Assuming that i) real perturbations - *in particular on the data collection* - cannot be recorded or generated in real industrial environment (that could lead to production stops) and ii) a model would not be implemented before evaluating its potential deviations, limits or weaknesses, our approach is based on artificial injections of perturbations into the data sets, and is evaluated on state-of-the-art classifiers (both Machine Learning and Deep Learning) and data sets (in particular, public sensors-based univariate time series). **First**, we propose a coarse-grained study, with two artificial perturbations - *called swapping effect and dropping effect* - in which simple random algorithms are used. This already highlights a great disparity of the models' robustness under such perturbations that industrial actors need to be aware of. **Second**, we propose a fine-grained study where instead of testing randomly some parameters' values, we used Genetic Algorithms to look for the models' limits. To do so, we define our multi-objectives optimisation problem with a fitness function as: maximising the impact of the perturbations (i.e. decreasing the most the model's accuracy), while minimising the changes in the time-series (with regards to our two parameters). This can be seen as an adversarial case, where the goal is not to exploit these weaknesses in a malicious way but to showcase their effects and notify future

practitioners of their consequences for production systems. Based on such a study, methods for making more robust the model and/or for observing such behaviour on the infrastructure could be investigated and implemented if needed. The tool developed in this latter study is therefore ready for being used in a real industrial cases, where data sets and perturbations can now be fitted to the scenario.

Acknowledgments

Throughout this thesis, I have received a lot of support and assistance.

First of all, I would like to thank Pr. Yves Le Traon who allowed me to pursue my PhD studies in his group and under his supervision. He believed in my research and provided me with valuable feedback. Of all the people in my thesis, I am especially grateful to my co-supervisor and daily advisor Dr. Jérémy Robert for his patience and advice, as well as his support. He taught me how to conduct proper experiments and present my results to others. The debates we had allowed me to better understand the field, but also to become a better researcher altogether. I am also thankful to my co-authors for their efforts and feedback that contributed to make this thesis stronger, as well as all my colleagues from SERVAL (SnT) for their expertise and help when I needed it. I also want to acknowledge all the jury members for their interest in my research and the time invested in my dissertation.

I am thankful to the National Research Fund (FNR) of Luxembourg for trusting in the initial ideas that lead to this dissertation by granting the funding that supported my thesis. Equally, I would like to express my gratitude to Cebi Luxembourg, and especially Guillaume Pollicarpo, who was at my side for four years and offered me a real industrial playground. It brought an additional dimension in my research while developing my knowledge and skills.

Finally and more personally, I thank my partner Tiphaine, for her love, constant support, and patience during this thesis. I also thank my father Roland, for always giving me the best possible conditions to pursue my ventures. From the bottom of my heart, I thank my sister Aurélie, who contributed to this journey since the very beginning, in a way she could not even understand. I would also like to express my gratitude to my brother-in-law Joffrey for having been a model for me. I would also like to thank my goddaughter Camille and my nephew Léo, for giving me so much love and joy. I want to thank Valentin Poirot, a former classmate and PhD student with whom I shared great moments and discussions. Also, I would like to make a special mention to Julien Polge, with whom I shared

the last nine years as a student. Last but not least, my special thanks to my very best friends, Florian, Joris, Baptise, Alexandre and Lucas, for their unconditional support, help and the great times we had.

Contents

1

Introduction

The industry is the part of an economy that transforms raw materials in finished or semi-finished products. In 2020, it represented about 26 % (including 15 % in manufacturing) of the worldwide Gross Domestic Product (GDP) according to the World Bank national accounts and OECD National Accounts data. Since its inception, the industry has evolved a lot, and the main changes have been conceptualized in four industrial revolutions, as depicted in Figure 1.1.

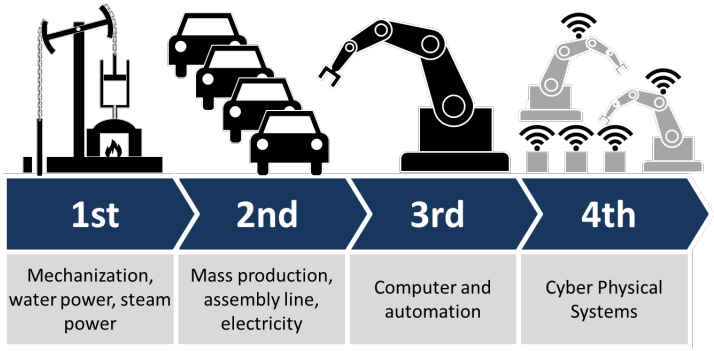


Figure 1.1: The four Industrial Revolutions - *Christoph Roser at AllAboutLean.com*

The **first industrial revolution**, that happened in the late 18^{th} century, is the transition from hand production methods to machines, thanks to the increasing use of steam and water powers. The result was productivity improvements, adding a new dimension to the industrial output. The **second industrial revolution** followed a hundred years later, around the turn of the 20^{th} century and was a period of rapid industrial development. This era introduced electrically-powered assembly lines enabling mass production. The **third industrial revolution** has started around 1970. It is characterised by the use of electronics in industries. Computers and Programmable Logic Controller (PLC) have been developed and deployed on assembly lines in order to automate the manufacturing processes. This automation helped increasing the efficiency of the production floors. Nowadays, we are in the **fourth industrial revolution**, the so-called Industry 4.0. This term was introduced by the German Federal Ministry of Education and Research which involves the technical integration of CPS into logistics and manufacturing and the usage of Internet of Things (IoT) technologies in industrial processes. From its origin, Industry 4.0 - *derived from the German term Industrie 4.0* – is used as a synonym for Cyber Physical Production Systems (CPPS), i.e. CPS applied to the manufacturing sector [VHH16].

Industry 4.0 is a concept that encompasses different objectives in order to satisfy the new customers' needs, as a strong product customisation and, therefore, the necessity for companies to have a highly flexible and traceable production [SKK+15, LFK+14, MSS18] while increasing its efficiency - *commonly evaluated in*

the industry by a Key Performance Indicator (KPI) called Overall Equipment Effectiveness (OEE) -. To reach such objectives, industrial and academic researchers define requirements and principles characterizing the Industry 4.0 that can be summarized from a high-level viewpoint - based on [VHH16] and [KWLJ17] - as follows:

i. RELIABILITY: production systems have to run even if any perturbations are encountered;
ii. COMMUNICATION: information systems of the industry (business applications, machines, humans) have to be connected and be able to communicate with each other by providing a high-level of interoperability;
iii. SECURITY: information systems have to offer data confidentiality, integrity and availability, while identifying assets (machines or humans) in an unique manner;
iv. DATA ANALYSIS: data coming from production floors have to be deeply analysed to make smarter decisions at production or factory level by using Machine-Learning (ML) or Deep-Learning (DL) algorithms that are able to deal with a huge amount of data in real-time;
v. ASSET ADMINISTRATION SHELL (AAS): the different industrial assets have to be modelled by using a standardised and interoperable semantic (i.e. providing cross disciplines data integration) in the virtual/digital worlds, so as to build powerful and relevant Digital Twins.
vi. SELF-CONFIGURATION: production systems have to be able to automatically discover and/or configure themselves (e.g. when devices are added or replaced).

However, even if researchers and engineers have already proposed methods and solutions to meet such requirements, it remains difficult for industrial companies to step into Industry 4.0. In fact, some manufacturing companies - *especially companies having decades of existence* - do not have the maturity and/or the expertise, neither in terms of interoperability nor in data analysis, to take benefits of these technologies. Industrial actors tends therefore to create partnerships with academic research centers in order to be helped/advised in their digital transformation. As an example, **Cebi Luxembourg S.A**. - *a manufacturer of electromechanical components for automotive equipments and household appliances sector* - contacted, in 2017, the **Interdisciplinary Centre for Security, Reliability and Trust (SnT)** to start a partnership about their biggest generational shift in its history. A Research, Development and Innovation (RDI) project[1] was officially signed in April 2018, in which **DataThings** - *a Luxembourgish start-up company which is specialized in data analysis and artificial intelligence* - is also involved.

[1] www.cebi.com/en/signature-ceremony-industry-40-project

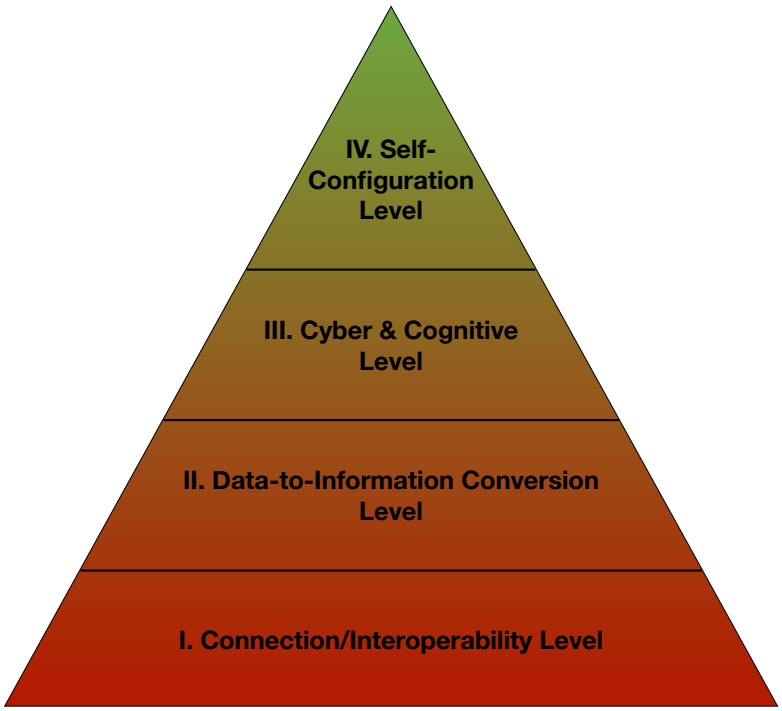


Figure 1.2: Functional Levels of Industry 4.0

The ultimate objective of this project is to lay down the foundations of potential (self-)configuration of manufacturing production units by Artificial Intelligence to handle changing demands, failures or non-optimal aspects of the production process. To do so, different functional levels - depicted in Figure 1.2 - have been defined and need to be reached in order to be Industry 4.0-ready:

- Level I - Connection/Interoperability: refers to the production unit itself, with proprietary PLC using fieldbus networks and the different control/commands specification of the machine.
- Level II - Data-to-Information conversion: refers to creation of interoperability and unified interface between the different machines of level I and the industrial application at the level III.
- Level III - Cyber & Cognitive: refers to the evaluation of the predictive models (learning algorithms) and the modelisation of expert knowledge.
- Level IV - Self-Configuration: refers to the definition of an AI-based system that make autonomous decisions.

In order to deal with each layer complexity, six work-packages (WP) have been defined. The SnT - *in particular with two PhD candidates* - is strongly involved in three of them. One PhD student is focused on "Cybersecurity in Industry 4.0" as

part of the WP3 and *I* am focused on the topics "Connected Manufacturing" and "Cognitive Manufacturing" as part of WP2 and 4.

These topics are of the utmost import since the production units of Cebi are (- *at the beginning of the project* -) neither connected to each other nor to the entreprise network. Indeed, such production units rely on proprietary PLC that do not enable communications between the Operational Technologies (OT) and Information Technologies (IT). In fact, legacy controllers do not have open/standardized interfaces to communicate with different business applications (e.g. monitoring or more advanced data-driven applications). So far, production monitoring was done manually (which is error prone and not in real-time), and the decision-making was based only on such - potentially inaccurate - data. This scenario is not only encountered by our partner, but by many industrial actors that have legacy production units.

In addition, industrial companies historically used to implement deterministic processes. Introducing stochastic approaches like AI - either in the production processes or at business level - demands a full understanding and trust on the ability of the underlying technology to be robust. AI models can reach excellent performance on state-of-the-art scenario regarding Time Series Classification (TSC) tasks [SL17, DBK$^+$19, FFW$^+$19]. Nonetheless, some of regrettable events had happened in the past, e.g., when a pedestrian was killed by a self-driving car in Arizona [ped18], that may impact the willingness and trust to use such technologies in industrial settings. Indeed, it is not so difficult to imagine similar unexpected events that could happen in the factory if perturbations occur. With regards to the "communication" aspect, network perturbations such as delays or losses can appear when connecting more and more agents collecting and/or analysing data. Such perturbations can tamper the data and lead to an inaccurate analysis, resulting in inappropriate actions from the different actors [LDSP18, GAD17, SLS$^+$18]. As a consequence, *it is important to study the robustness of AI models* before deploying them in industrial data-driven production floors.

To sum up, the problem, addressed in this thesis, is then twofold: **1)** Industrial production systems are vertically-oriented closed systems that make difficult their communication and their cooperation with each other, and intrinsically the data collection. **2)** Industrial companies used to implement deterministic processes. Introducing AI - *that can be classified as stochastic* - in the industry requires a full understanding of the potential deviation of the models in order to be aware of their domain of validity.

In order to tackle these challenges, this dissertation proposes a unified strategy for digitizing an industrial system and methods for evaluating the performance

and the robustness of AI models that can be used in such data-driven production plants.

This dissertation is a compilation thesis, i.e. each chapter is an article published or to be submitted, linked to each others by a foreword. The thesis consists of two parts related to I. Unified approach for data-collection and II. The study of the ML models robustness in industrial settings. In the first part, chapter II is refering to an analytical performance evaluation of O-MI/O-DF and MQTT protocols to provide interoperability in the case of a remote industrial monitoring application. Chapter III refers to the design and the evaluation of TRIDENT, a three-steps strategy to provide interoperability between heterogeneous vertically-closed systems. In the second part, chapter IV presents a systematic approach to evaluate the robustness of Machine Learning models in industrial settings, with a coarse-grained experimental study. Chapter V extends this approach to a fine-grained evaluation of the models' robustness, and proposes elements for the robustification of intelligent data analysis tools such as Machine Learning.

Part I

Towards a unified data-driven methodology to digitise an industrial production plant

2

O-MI/O-DF vs. MQTT: a performance analysis

Foreword

With the advent of the Industry 4.0 and in order to meet the new needs in terms of production flexibility while willing to increase the OEE, some of legacy industrial companies, and especially the ones that have decades of existence, want to upgrade their production tools. The first step of this transition is to enable the data collection with a unified (data-driven) approach in order to monitor the whole production floor in an interoperable and standardised manner. In the meantime, researchers and public actors (e.g., Europe H2020) are working on these industrial digitisation topics, and have developed protocols to provide connectivity and interoperability between vertically-closed silos.

However, industrial actors have to fully trust these technologies before implementing them and benefiting from them. They have to be sure that it will fulfill their requirements in terms of performance, while having a deep understanding of how it will be scalable in a real production floor (in order to design the future architecture). In order to investigate the perks and the potential trustworthiness of these technologies, collaborations between industrial and academic worlds are rising to help define the different requirements of the companies and to share the knowledge of the researchers. It has been the case between the **SnT** and **Cebi Luxembourg S.A.**, our industrial partner, established in 1976, in the midst of the third industrial revolution. The company started manufacturing temperature switches for the household appliance market and designing electromechanical components for the automotive market. In order to propose new products to the customers and to enrich their catalog, Cebi Luxembourg S.A. - *as other manufacturers*- extended and upgraded their production plant with time, leading to a heterogeneous production floor composed of neither connected nor interoperable PLC from different ages and technologies. Today, they expect to step into the fourth industrial revolution, not by replacing and redesigning its actual production system, but by upgrading it to be Industry 4.0-ready. The first step towards this digital readiness is considered as the ability to monitor in a unified manner the production tools, by collecting the different data available on the machines (and more precisely on the PLC) and understanding the impact of the monitoring process on the network.

To do so, we designed proof-of-concepts in order to test and evaluate the different existing approaches in real-life scenario (and then trusting such data-driven approaches). This proof-of-concept aims at raising awareness and providing performance indicators to our partner, so that he can decide to rely on it or not (e.g., regarding performance and scalability aspects). In

that sense, the first task was to study the ability of two well-established protocols in the research community (i.e. O-MI/O-DF and MQTT) to provide connectivity and interoperability in the production floor of our partner. Therefore we propose an analytical model that describes the traffic-load generated on the network of these two protocols (i.e., O-MI/O-DF and MQTT) for different communication modes. This analytical model provides a detailed description of the payload generated by the different network layers of data exchange using those two protocols. It has also been applied to a real industrial case study for which we computed the traffic-load generated for monitoring pilot production tools of our partner.
This chapter presents our study and has been published as a peer-reviewed conference paper titled *"O-MI/O-DF vs. MQTT: a performance analysis" [BRLTK18]* at IEEE Industrial Cyber-Physical Systems (ICPS) conference in 2018.

Contents

2.1 Abstract

Over the past decade, a flourishing number of concepts and architectural shifts appeared such as Industrial Internet of Things (IIoT), Industrial CPS or even Industry 4.0. Unfortunately, today's IoT as well as Industry 4.0 environments, look more like a collection of isolated "Intranets of Things", also referred to as "vertical silos", rather than a federated infrastructure. Breaking down these silos is a key challenge in both the IoT and Industry 4.0 communities. This paper is intended to present and discuss two open and standardised messaging protocols designed for IoT applications, namely: MQTT and O-MI/O-DF. First, a traffic load's analytical model derived from the MQTT standard specifications is presented. Second, a comparison study between MQTT and O-MI/O-DF standards is carried out based on a real-life industrial implementation. This study brings a deep understanding of the extent to which these protocols are performant (from a traffic load perspective) and how they can impact on future architectural designs.

2.2 Introduction

Over the past decade, a flourishing number of concepts and architectural shifts appeared such as IoT, Cyber-Physical Systems (CPS) or Internet of Everything (IoE). Applying these concepts to the industrial application scenarios leads to the definition of the following terms: IIoT, Industrial CPS or even Industry 4.0.

Currently, the Industrial Internet consortium is essentially driven by US enterprises, meanwhile in Europe similar initiatives have different names: 'Industrie 4.0' in Germany, 'Smart Factory' in the Netherlands, 'Usine du Futur' in France, etc. Those disciplines have become a technological focus area for academia, industry, and governmental organisations, as stated by the number of papers found on Google Scholar: 287 papers – from 2013 – using 'Industrial Internet of Things' as title keywords, and 1160 papers using "Industry 4.0". Strictly speaking, differences between the aforementioned terms could be elicited. Nevertheless, the term Industry 4.0 is used throughout this document for consistency purposes.

One of the important aspects of the Industry 4.0 is to increase connectivity between the technologies, systems and processes [KWLJ17]. Indeed, most of the companies that have been created decades ago, have heterogenous, non-interoperable and proprietary systems that are expensive to train on and maintain. Adding to that, future production systems have to be developed considering the need for strong product individualisation/customisation and, therefore, the necessity for high flexible production processed [SKK+15]. It includes the industrial communications networking and IT infrastructures. Those infrastructures are still

based on the building automation pyramid [Sau10], which are still too rigid for meeting flexibility and adaptability requirements. From a long-term perspective, networked Things and IO modules (at the field level) will be identifiable and accessible through the Internet, constituting a dynamic global network infrastructure with self-configuring capabilities. Even if a first level of interoperability is achieved at the field level by relying on open technology standards such as Ethernet-based solutions, it still remains to manage one of the most critical obstacles, namely the *vertical silos'* model that shapes today's IoT [KRH+16]. Data is not anymore dedicated to a particular use but expected to be connected to many other organizational information systems to support various types of activities, spanning from production, to business, and to services. One solution to solve this problem is to rely, as much as possible, on open communication standards at the Application layer, where both technical and semantic interoperability are tackled [Tur07].

While many manufacturing companies are willing to move towards the Industry 4.0 paradigm, their IT infrastructure, often dating back from the early 70's or 80's, prevent them from taking full advantage of that paradigm. As a result, it is of the utmost importance to help them to efficiently step into the Industry 4.0 by proposing efficient network infrastructure frameworks to connect all the existing vertically-oriented closed systems (production units, metrology station, ...). To do so, it is important for system designers to be aware, beforehand, about the traffic load that a system (through its gateway) will generate depending on the adopted communication protocol. The contribution of this research is twofold: i) propose an analytical model of the traffic load (and efficiency ratio) based on the MQTT standard specifications ii) carry out a comparison analysis with an other well-know IoT communication protocol, and particularly the O-MI/O-DF standards (a first analytical model being proposed in [RKLTF16]). These two protocols have been selected for this study not only because they can be used for Machine-to-Machine communications, but also due to their intrinsic capabilities. The first one implements an aggregation-like mechanism, meaning that all the data are bundled in the same message, whereas the second is not fully compliant with this model (using one response message per data item/topic). This study will allow us to draw conclusions about the efficiency of each protocol and their underlying mechanisms. The originality of this research, compared with the existing literature where several experimental analysis have been conducted with regard to MQTT [NSP17, DCJ15], lies in the fact that no comparison study between MQTT and O-MI/O-DF has been proposed and quantified yet.

The rest of the chapter is organized as follows. Section II presents the mathematical models of the traffic load for both protocols. In Section III, a real industrial case-study is described and analysed. Section IV discusses on how to use these study results in an industrial CPS. Finally, Section V concludes this chapter.

2.3 Traffic-Load Analysis

2.3.1 Introduction of O-MI/O-DF & MQTT

O-MI [Gro17b] and O-DF [Gro17a] standards, which have been specified and published by *The Open-Group* standardization fora, are independent entities that reside in the Open Systems Interconnection (OSI) Application layer, respectively specified at the 'communication' and 'format' levels [FKB14]. O-MI provides a generic Open API (specifying different interfaces such as Read, Write… as summarized in Figure 2.3) for any REpresentational State Transfer (REST)ful IoT information system. IoT gateways that implement O-MI can act both as a "server" and "client", meaning that communications are established in a peer-to-peer manner. O-DF standard can be combined on top of O-MI - *although not mandatory* - for describing 'Things' in a generic manner. Note that more specific vocabularies can also be added to the O-DF structure, as discussed in [RKK+17].

MQTT is a connectivity protocol for IoT and Machine-to-Machine communications. Standardised by the OASIS body, it also resides in the OSI Application layer, relying on the Publish/Subscribe model (i.e., clients communicate with a broker in a peer-to-peer manner). The data are described according to a string-based (hierarchical) topic (e.g., "smartHouse/temperature"). MQTT distinguishes between three Quality of Service (QoS) levels at the application layer, since it relies on Transmission Control Protocol (TCP)/Internet Protocol (IP) for the lower ones. In addition and contrary to O-MI, MQTT is a connection-oriented protocol, meaning that clients need to setup a connection with the broker before publishing or subscribing any data/topic.

2.3.2 MQTT: a traffic load analytical model

MQTT is independent of the lower layers (Medium access, Link layer, Network and Transport). Thus, the size of the request, or response, or acknowledgement, respectively denoted by S_{req}, S_{resp}, S_{ack} can be formalized as in Eq. 2.1.

$$S_{\text{req}} = \ell_{\text{low-layer}} + \ell_{\text{app-layer}} \tag{2.1}$$

Application layer

According to the OASIS standard specifications [Sta14], the MQTT header is composed of both a fixed and variable part as emphasized in Figure 2.1. The fixed part corresponds to the first byte and enables to specify the message type and QoS level. The variable part defines data-related information such as the header's length, topic's name, or data payload. The application layer size can therefore

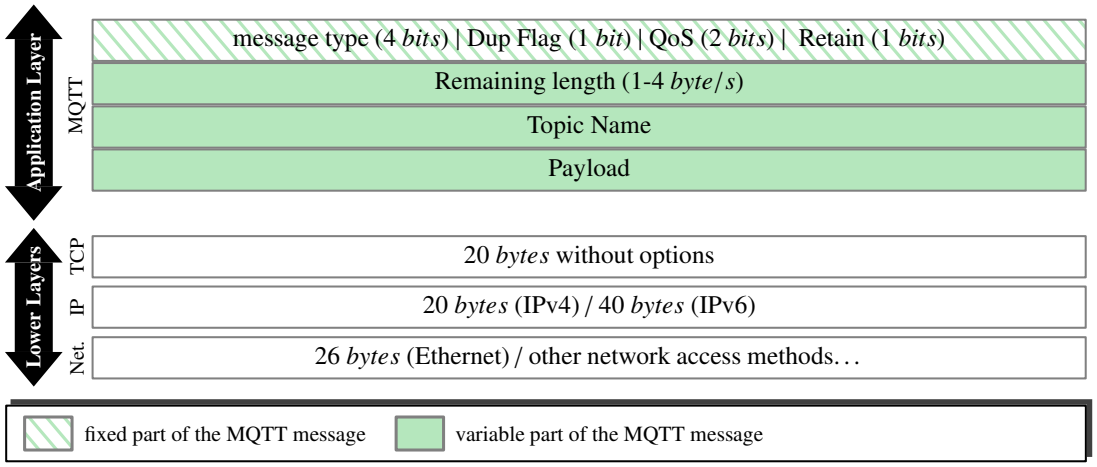


Figure 2.1: MQTT message size

be written as in Eq. 2.2, where the remaining length is defined as the sum of the data and topic sizes. This value (in decimal) needs to be coded over 1 to 4 *bytes* according to the standard. As a consequence, the number of bytes needed for coding this decimal value can be expressed as in equation 2.3). Let us note that only 7 bits out of 8 are used to code the remaining length, as the first bit (called "*Continuous bit*") enables to specify whether or not there is a subsequent byte for this field.

$$\ell_{\text{app-layer}} = 1 + \ell_{\text{length}} + \ell_{\text{topic}} + \ell_{\text{data}} \tag{2.2}$$

$$\ell_{length} = \left\lceil \frac{\left\lceil \frac{\ln(\ell_{topic}+\ell_{data})}{\ln(2)} \right\rceil}{7} \right\rceil \tag{2.3}$$

Lower layers

As previously mentioned, MQTT takes place over TCP/IP connections. The length of the (network) IP header depends on whether IPv4 or IPv6 is in use, as emphasized in Figure 2.1. In our model, we consider Ethernet as the underlying network access protocol, but other protocols could be considered as well (e.g., IEEE 802.15.4). As a result, $l_{low-layer}$ is either equal to 66 bytes $(26 + 20 + 20)$ or 86 bytes $(26 + 40 + 20)$.

Traffic Load & Efficiency Ratio

As outlined in red in the Figure 2.2, only the data exchanges phase (i.e. *Publish/Subscribe* messages) are considered in this study, which corresponds to a QoS

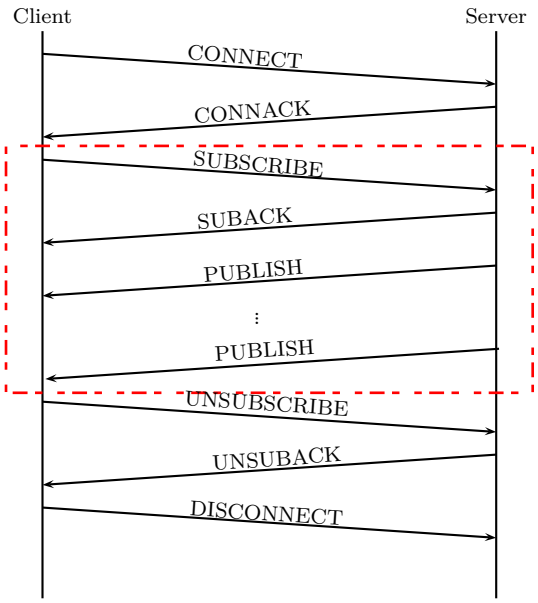


Figure 2.2: MQTT sequence diagram (considering a QoS level=0)

Table 2.1: Variables used for O-MI/O-DF formulas

Protocol	Variable	description
HTTP	ℓ_{url}	URL length
	ℓ_{reason}	HTTP reason-phrase length
O-MI	ℓ_{ttl}	O-MI TTL field length
	ℓ_{rc}	O-MI Return code length
	ℓ_{reqID}	Request ID length
	ℓ_{int}	Subscription Interval length
	ℓ_{call}	Callback address length
O-DF	ℓ_{objID}	Number of digits of Object's ID
	ℓ_{name}	Number of digits of InfoItems' name
	ℓ_{value}	Number of digits of Value

level equal to zero. This is the best case for minimising the number of exchanges, and accordingly the traffic load. Connection and disconnection phases are not taken into account, but could easily be added since MQTT messages use the same header. Given this, let T be the number of topics that can be subscribed to (i.e., number of 'subscription' messages); 'Suback' be the message (of a size S_{suback} - *as defined in Eq. 2.1* - with $\ell_{data} = 0$ *bytes* in Eq. 2.2) used by the broker to

acknowledge each subscription; and P be the number of 'Publish' messages sent by the MQTT broker to all clients having subscribed to a given topic. It can be noted that $T \leq P$ since MQTT can use an aggregation mechanism in the subscription process. It means that the subscriber can directly subscribe to all the data hierarchy available at the broker level using the `#` character (e.g., `/smartHouse/#/` means that the request subscribes to all topics under the `smartHouse` topic) or using the wildcard + (e.g., `/smartHouse/+/temperature/` means that + will be replaced by any available Object). However, it cannot be used in the *Publish* messages, and it is therefore necessary to send as many messages as topics. Given all these parameters, the traffic load can be expressed as in the equation 2.4.

$$TL(T,P) = \sum_{t=1}^{T} \Big(S_{\text{req}}(t) + S_{\text{suback}}(t) \Big) + \sum_{p=1}^{P} S_{\text{resp}}(p) \tag{2.4}$$

Let us note that equation 2.4 (or Eq. 2.1 to be exact) does not take into account the lower layer constraints, and particularly the network access method in terms of Maximum Transmission Unit (Maximum Transmission Unit (MTU) - 1500 *bytes* in Ethernet). Indeed, if $\ell_{\text{app-layer}} > MSS$ (Maximum Segment Size - 1460 bytes with IPv4/TCP), the number of frames is expressed as $n = \left\lceil \frac{\ell_{\text{app-layer}}}{MSS} \right\rceil$. Eq. 2.1 can therefore be refined as in Eq. 2.5 to express the total length of data transmitted by the network (either for the request L_{req}, response L_{resp} or the application acknowledgement L_{suback}).

$$L_{\text{req}} = (n-1) \cdot (\text{MTU} + \ell_{net}) + S_{\text{req}} - (n-1) \cdot \text{MSS} \tag{2.5}$$

As MQTT relies on TCP as transport protocol, it is also important to take into account the exchanges added by TCP such as the opening and closing TCP connections and segment acknowledgments. However, the transient states of TCP opening and closing operations are not considered in this study as one or more MQTT messages can be transmitted over a same TCP connection. The overall traffic load can therefore be defined as in Eq. 2.6, with L_{ack} defined as in Eq. 2.7

$$TL(T,P) = L_{\text{ack}}(T,P) + \sum_{t=1}^{T} L_{\text{req}}(t) + L_{\text{suback}}(t) + \sum_{p=1}^{P} L_{\text{resp}}(P) \tag{2.6}$$

$$L_{\text{ack}}(T,P) = \left[\sum_{t=1}^{T} \left(\left\lceil \frac{n_{\text{req}}(t)}{m} \right\rceil + \left\lceil \frac{n_{\text{suback}}(t)}{m} \right\rceil \right) + \sum_{p=1}^{P} \left(\left\lceil \frac{n_{\text{resp}}(p)}{m} \right\rceil \right) \right] \cdot \ell_{\text{low-layer}} \quad (2.7)$$

Eq. 2.7 takes into account the TCP acknowledgments, which can be sent either immediately a segment is received, or after several segments are received, or inside a new data transmission (piggybacking). This is taken into consideration in our study by defining the m variable as the number of received segments after which an acknowledgement is sent. Indeed, the real value of m depends on the TCP behaviour that changes, in practice according to the operating system and the TCP configuration. Let us note that L_{ack} considers the acknowledgements for all messages used in the data exchanges .

Let us note that based on the traffic load, it is straightforward to define the efficiency ratio of the protocol. As this parameter is computed in the industrial case-study, Eq. 2.8 provides the generic expression of the efficiency ratio.

$$ER = \frac{\ell_{\text{payload}}}{TL} \quad (2.8)$$

2.3.3 O-MI/O-DF: a reminder of the traffic load analytical model

A specific methodology has been introduced in [RKLTF16] to build the analytical model of the O-MI/O-DF standards. The same methodology is followed in this study to build the analytical model of the MQTT standard. As explained before, O-MI relies on the Hypertext Transfer Protocol (HTTP) protocols. The application layer size can thus be expressed as in Eq. 2.9.

$$\ell_{\text{app-layer}} = \ell_{\text{HTTP}} + \ell_{\text{O-MI}} + \ell_{\text{payload}} \quad (2.9)$$

Figure 2.3, associated with Table 2.1, remind the formulas needed for computing the size of (i) the HTTP header (i.e., ℓ_{HTTP}); (ii) the O-MI requests/responses (i.e. $\ell_{\text{O-MI}}$), (iii) the O-DF payload (i.e. ℓ_{payload}). Let us note that the traffic load is computed thanks to the expression $TL = L_{req} + L_{rep} + L_{ack}$ (since O-MI/O-DF will use only one message for the request and one for the response from an application perspective), where L_{ack} is defined as follows: $L_{\text{ack}} = \left(\left\lceil \frac{n_{\text{req}}}{m} \right\rceil + \left\lceil \frac{n_{\text{resp}}}{m} \right\rceil \right) \cdot \ell_{\text{low-layer}}$, since there is no application acknowledgments.

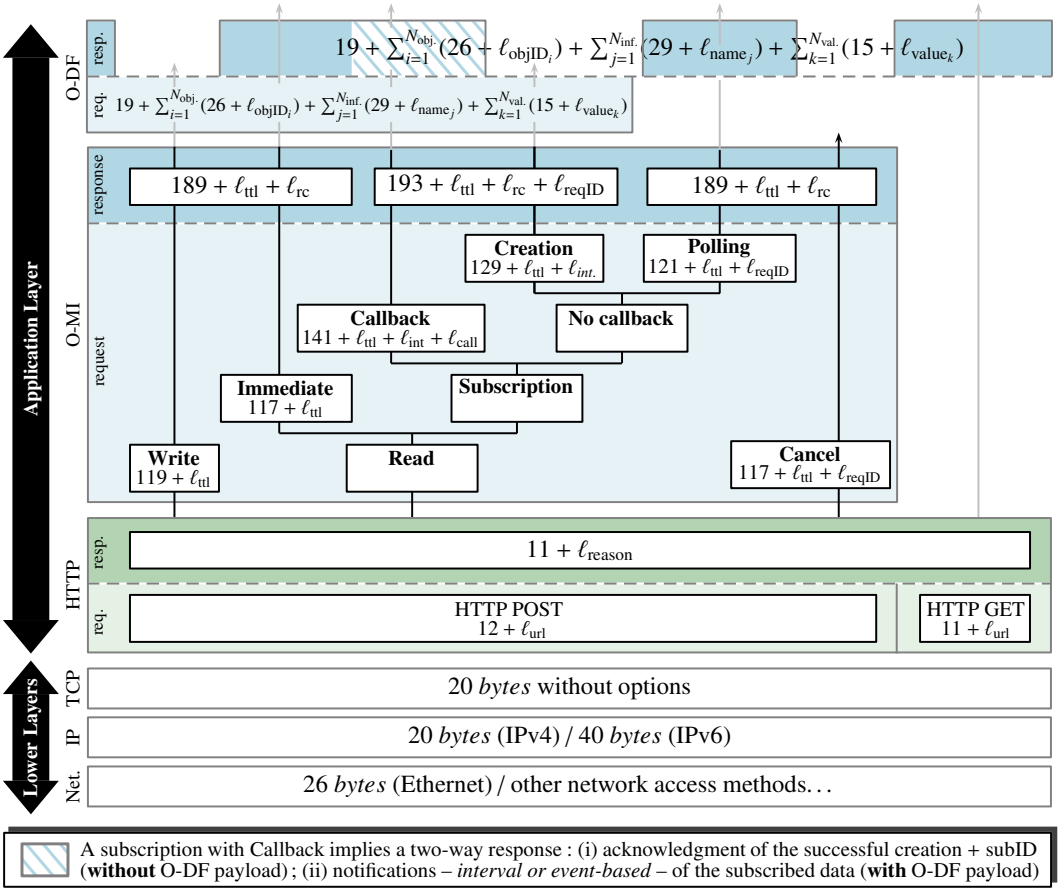


Figure 2.3: O-MI/O-DF request/response message size [RKLTF16]

2.4 Industrial case study: O-MI/O-DF vs. MQTT

The overall use case is depicted in Figure 2.4, which involves a company expecting to publish industrial plant-related information through an O-MI server or a MQTT broker in order to create a basic application for monitoring the production environment. Let us assume that only two vertically-oriented closed and proprietary systems exist: one for the plant metrology (based on the technology Saveris) and the second one for the production itself (information can be accessed directly from each production unit through a PLC). An agent (or wrapper) has been developed and implemented on the O-MI server and the MQTT broker for translating the information coming from these two systems and making them compliant with the O-MI or MQTT standards. The performance evaluation and comparison take place between the O-MI server/MQTT broker and the application. As the application is intended to be used for monitoring these two systems, all the data needs

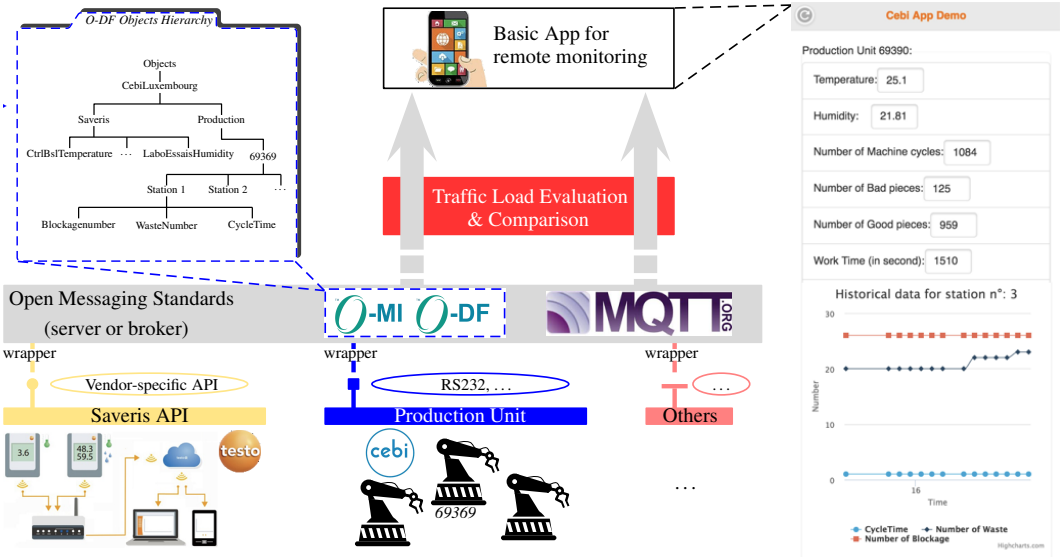


Figure 2.4: Industrial Use Case Scenario

to be collected on this application.

Even though the software agents between the proprietary systems and the server/broker are not presented in this study, it is necessary to define the data structure that needs to be communicated in the application. Figure 2.4 shows also the generic O-DF Objects hierarchy built for this scenario. This hierarchy highlights that `Saveris` and `Production` are defined as O-DF 'Object', inside the main Object `CebiLuxembourg` (corresponding to the name of the company). In the `Saveris` Object, the Object properties (called InfoItems) have been defined so as to correspond to the sensor information. In the `Production` Object, different O-DF 'Object' are nested; for instance, `69369` corresponds to the number assigned to the considered production unit, which is divided into many stations. And finally, InfoItems named '`Blockagenumber`', '`WasteNumber`', ..., provide information about the status of each station. To be consistent with this structure as well as for comparison purposes, the MQTT topics are based on the O-DF paths defined by this hierarchical structure, as MQTT does not specify, impose, nor recommend any data structure. For example, the topics are: '`Objects/CebiLuxembourg/Saveris/CtrlBslTemperature`', ..., '`Objects/CebiLuxembourg/Saveris/LaboEssaisHumidity`', '`Objects/CebiLuxembourg/Production/Station 1/Blockagenumber`, and so on. Overall, it corresponds to 21 Objects and 78 InfoItems in the O-DF structure, and to 78 topics in MQTT, which

Table 2.2: Overall Traffic Load (in bytes) on the Industrial setting: O-MI/O-DF vs. MQTT

Layers	**O-MI/O-DF**	**MQTT**
Payload (value)	503	503
Data presentation (i.e. O-DF structure or MQTT Topics)	5341	2954
Messaging protocol (i.e. O-MI or fixed part & length of MQTT)	315	160
HTTP - *if needed*	48	0
TCP	240	3200
IP (v4)	240	3200
Network access methods (Ethernet)	312	4160
Overall Traffic Load (*Efficiency Ratio*):	**6999 (*83.5%*)**	**14177 (*24.4%*)**

makes it easy to compare both standards. The first analysis is based on the industrial setting above-introduced, and the second one evaluates the impact of using aggregation-like mechanisms.

2.4.1 Industrial setting analysis

In this scenario, we assume that the application – for plant monitoring – periodically requests all the data hierarchy/topics. The application sends O-MI `Immediate Read` requests (one of the operations defined in Figure 2.3) to the O-MI server, by specifying either the whole O-DF structure or only part of it as payload. In order to minimise the request size, the O-DF root 'Object' is only embedded as payload in the O-DF read request. Following this request, response messages containing the "Values" of the requested InfoItems are pushed to the application. On the other hand, the application subscribes to all MQTT topics by sending a `Subscribe` request by containing the following aggregated topic `Objects/xx...xx/#` in order to minimise this request as well (otherwise, as many requests as topics should be sent). Following this request, the MQTT broker pushes a `Publish` response containing the "Values" of each topic (each time it receives a notification from the system).

Based on this scenario, the analytical model of the traffic load detailed in the

section II are applied and the associated results are shown in the Table 2.2. The following conclusions can be drawn:

- The overall traffic load is much less important (around twice less) when using O-MI/O-DF rather than MQTT. Indeed, the O-DF structure (especially in the response) is based on an aggregation-like mechanism that plays a major role in minimising the overall traffic load (compared with MQTT, which does not implement such kind of mechanism). Even though the overall length of the MQTT data presentation (considered only as the topics) is also much less important than in O-MI/O-DF, the transport of these data needs to use as many TCP segments as topics (i.e., 80 by counting the subscription request and the application acknowledgement of this subscription without taking into account the TCP acknowledgements) instead of using only 6 TCP segments when using O-MI/O-DF. In addition, we considered that for each received TCP segment, a TCP acknowledgment is sent (i.e. $m = 1$). As a result, the total number of TCP segments is equal to 12 for O-MI/O-DF and 160 for MQTT, and then the length of the low layers is computed accordingly (see rows 'TCP', 'IP','Network access method' in Table 2.2).
- From an efficiency ratio perspective, it can be noted that this ratio is more than three times more important for O-MI/O-DF than for MQTT. Let us note that the data presentation part (in addition of the values themselves) is also considered as payload for computing this ratio (i.e. for O-MI/O-DF: $(503 + 5341)/6999 = 83.5\%$; for MQTT $(503 + 2954)/14177 = 24.4\%$). Indeed, we consider that the generic data model (with the use of semantic vocabularies) introduced in O-DF is beneficial for representing IoT data/service in order to integrate more complex reasoning out of them. Even though MQTT can give some information about the hierarchy (through the topic structuration), MQTT does not allow to add metadata to the sensors' values (e.g, unit of the value, accuracy of the sensor, and so on).

2.4.2 Aggregation- vs. non aggregation-like mechanism

In this section, the objective is to set up the size of the data structure – *O-DF parameters such as the Object's name, InfoItems's name and values, as well as MQTT parameters such as the topics' name and values* – in order to assess the impact of the number of O-DF InfoItems or MQTT topics on the Traffic Load, and accordingly the impact of using an aggregation-like mechanism. To do so, it is important to increase only one parameter (i.e. the number of InfoItems/topics) at the same time. Based on the lengths defined in the real-life scenario, the average length for each parameters is considered, namely $\ell_{objID} = 8\ bytes$, $\ell_{name} = 14\ bytes$ and $\ell_{value} = 6\ bytes$ (i.e., that all objects name/InfoItems name/values have re-

spectively the same length. As the request does not change with the previous scenario (requesting all available information), it means that the first response only contains one InfoItem for O-MI/O-DF and only one topic for MQTT, the second only two (first) InfoItems/topics, and so on, until reaching the whole InfoItem hierarchy/topics (78 in total).

Figure 2.5 gives insight into the Traffic Load evolution, along with the number of frames sent according to the number of O-DF InfoItems or MQTT topics. It can thus be noted that:

- If the number of InfoItems/topics to be collected is inferior to 10, then the aggregation mechanism of O-MI/O-DF is not efficient in terms of traffic load since the traffic load in MQTT is less important in this case.
- On the contrary, if the number of InfoItems/topics to be collected is superior to 10, then the aggregation mechanism of O-MI/O-DF is more efficient.

However, this conclusion should also be put into perspective with the fact that, the higher the number of MQTT topics, the higher the number of messages. But at the same time, the higher the number of O-DF InfoItems, the bigger the size of the O-MI/O-DF response. This is also important to be noted, as the traffic load will significantly increase in case of frame error occurrences (since TCP needs to retransmit the frames in error). All these considerations must be taken into account when designing a network, including more complex functionalities such as the network control (e.g., for adapting it to the demand).

2.5 Conclusion: Towards Performance-driven network designs in Industrial CPS

Industrial CPS is gaining a growing attention in both the academic and industrial sectors. With an increasingly trend to digitalize all aspects of companies, it is an important shift of paradigm that is leading to a new way to design the industry. Lower level systems (in the automation pyramid) needs to be visible at the higher level, but also accessible from the outside world for creating more flexible, agile and smarter services for the industry. It consists of developing digital shadow(s) of the vertically-oriented closed and proprietary systems by relying to the best extent possible on open and standardised technologies, standards and frameworks. All these shadows form a complex and interlinked System, also referred to as "System-of-Systems" in the scientific community. Those systems need to communicate and cooperate, in real time, with each other and potential human beings. To do so, it is of the utmost importance to have a deep understanding to what extent these new protocols perform, and to what extent they impact on the network performance (e.g., in terms of traffic load). This is important, as it helps system designers

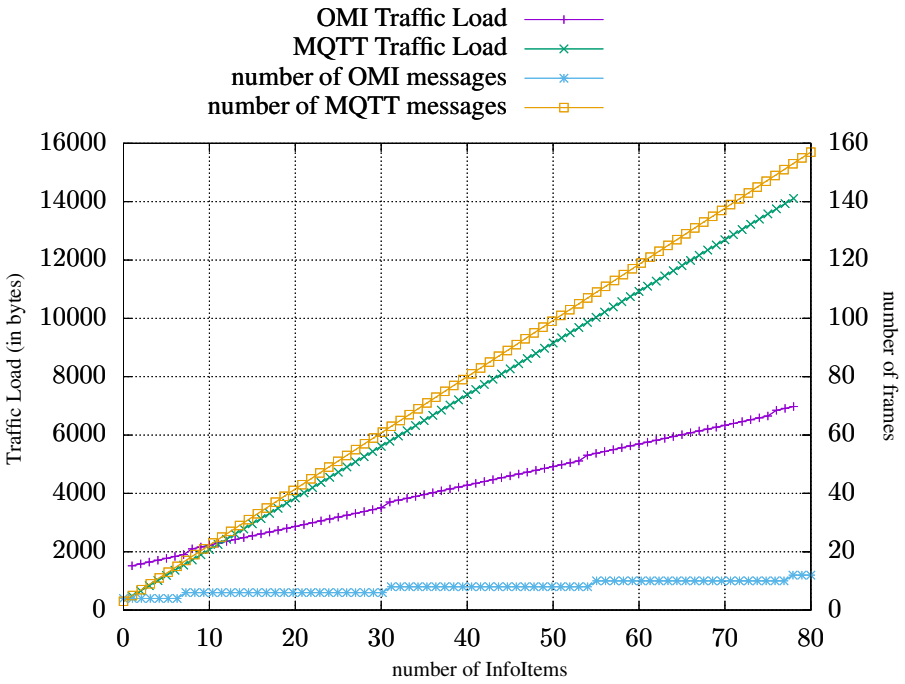


Figure 2.5: Traffic Load evolution vs. Number of O-DF InfoItems

to properly (re-)think and adapt the design of the network by assessing the compliance with the QoS requirements (freshness, delay, …), while supporting more dynamic structures. Dynamic refers here to the fact that the network has the ability to adapt itself to new production demands or fault recovery purposes. For example, if the production planning is frequently modified due to the use of the "*order your personalised product online to get it tomorrow*" paradigm, the network needs to be online reconfigured. For the design of the network, it means that the control functions, which are currently running on field controllers, become distributed on software components rather than on dedicated hardware devices. Such capabilities are offered by paradigms such as Software-defined networking (SDN) and Network Functions Virtualization (NFV).

In addition to succinctly presenting two open and standardised protocols for Machine-to-Machine communications (MQTT and O-MI/O-DF) through their assessment in a real-life industrial case-study, this studys provides some conclusions and trends regarding the use of aggregation-like mechanisms. Those trends could be used, as first rules, to control what mechanisms (or protocols) should be used for transporting real-time data according to current status of the network (current traffic load, noisy environment, …) and potential demands (in terms of number of requested data items/topics). Those rules could be implemented in an IoT gateway, as a PONTE-like bridge (https://www.eclipse.org/ponte/), which han-

dles interoperability issues by selecting the appropriate protocol and associated request/response messages.

3

TRIDENT: A Three-Steps Strategy to Digitise an Industrial System for Stepping into Industry 4.0

Foreword

Chapter II demonstrated that O-MI/O-DF and MQTT were theoretically suitable for providing interoperability in a production plant and for feeding a monitoring application regarding a given scenario. We also demonstrated that, depending on the data hierarchy/format, the aggregation mechanism (implemented by O-MI/O-DF) can have a positive impact on the generated traffic-load. This study helps demonstrate the impacts of the intrinsic protocols' mechanisms on the network. This is particularly interesting to pay particular attention to it when designing a data collection architecture.

Industrial companies willing to step into Industry 4.0 - *and then looking for a unified data collection architecture* -, have two options:

1. (re)build from scratch a new plant with PLC of new generation, embedding communications interface that provides intrinsically connectivity and interoperability between the different data producers and consumers. In that case, everything can be designed from scratch to fulfill the different requirements in an optimal way, in particular by choosing the suitable equipments - *possibly off-the-shelf* - that provide the required functionalities and interfaces, both for business and operational applications. Even if designing a new ecosystem from scratch has some benefits (e.g. interoperability-, connectivity- by-design), it remains a hot research topic since it needs a straightforward, systematic and performant strategy to cover the different topics related to the Industry 4.0 by design.

2. Update an existing production plant composed of legacy devices that have to be interconnected using wrappers and gateways. Things can be even more complex than the first option, or at least require intermediate stages in the design and in the implementation of such a data-driven ecosystem. It can be due to the history of the company, from its legacy systems to the way the production plant itself has been designed and upgraded over time. Actually, manufacturers have evolved their productions tools in order to enrich their catalog and to fit with the changing requirements of their customers, or even with the legal aspects (regarding safety, security or quality of the production). This led the companies to deal with heterogeneous technologies that cannot be connected to the entreprise network in a plug-and-play manner. Launching a digitisation project in such a plant's configuration requires some retro-fitting actions (as inventory of existing devices, their behaviours, available communications capabilities and the data

they can provide/manage) in order to build a strong mapping between old data producers and the future servers that will interface with the applications that consume this data.

These different options do not have the same cost for the companies and do not require exactly the same strategies. However, irrespective of the preferred option, it is of utmost importance to have a strategy to digitise a production plant that is standardised, unified, generic and replicable with the less effort possible. Even if the finality of both options remains the same (*being able to develop and connect the different IT and OT applications with the same data-driven approach*), the strategy to meet the expectations can slightly differ. One will require less intermediate/preparation steps (i.e. retro-fitting, wrapper creations, …) to collect and consume the data, the other will require less physical and humans changes (new (expensive) machines, humans teaching processes, …). Note that, in this research, development and innovation project, our partner Cebi Luxembourg S.A. chooses the second option (i.e. update of the production plant).

In that context, we propose a three-steps strategy in order to digitise an industrial production plant. This strategy is based on three main steps: 1) the definition of the information model consisting in identifying the data of interest and formatting them in a unified modelling language, 2) the creation of IoT gateways, composed of wrappers that will act as translators between proprietary closed protocols and open standards that provide interoperability, and IoT servers that provide the interface with the applications, and 3) the creation of the applications (e.g. Monitoring, ML, …) that are fed by the collected data.

Although it is a real need for industrial companies, and especially for our partner, to have this straightforward data-driven approach, it is also important to demonstrate that such a strategy can be reliable and efficient for their applications. To do so, we evaluate this generic strategy (that can be instanced with any protocols) in terms of temporal performance, which is a critical performance indicator for industrial companies. We have chosen to conduct experiments on O-MI/O-DF and MQTT (i.e. the same protocols than the previous study), and also on OPC-UA due to its growing adoption within the industrial community. Actually, looking at the protocols that are pushed by the different industrial consortiums, PLC manufacturers or companies that have already stepped into the fourth industrial revolution, OPC-UA seems to be the protocol that tends to be the most adopted to provide connectivity and interoperability in production plants. There are al-

ready big controllers/machines brands selling PLC with embedded OPC-UA servers. Also, these protocols can be implemented following different communication modes (aggregation vs. non-aggregation mechanisms, client/server communication, publish/subscribe service) for which there are benefits and downfallss. The choices of such implementations are a cornerstone in the design of an architecture and can have a big impact on the performance of the different applications. The further evaluation of our three-steps strategy takes these different possible choices into account in order to give the insights of such technical decisions to companies desiring to implement it.
This chapter presents our study and has been published as a peer-reviewed conference paper titled *"TRIDENT: A Three-Steps Strategy to Digitise an Industrial System for Stepping into Industry 4.0"* [BRLT19] at the 45^{th} Annual Conference of the IEEE Industrial Electronics Society, 2019.

Contents

3.1 Abstract

Nowadays, industrial companies are engaging their global transition toward the fourth industrial revolution (the so-called Industry 4.0). The main objective is to increase the OEE, by collecting, storing and analysing production data. The challenge to be tackled is to collect and make available data from the production units in a real-time and standardised manner. This paper proposes a strategy to digitise an industrial system, that can be used regardless the industrial environment . This strategy is applied on a real case-study and deployed on an industrial assembly line. The evaluation has been led by measuring the performance of three standards (i.e. OPC-UA, MQTT and O-MI/O-DF) highlighted by both industrials and academics. The study points out the i) feasibility of applying our strategy and ii) the suitability of 2 out of 3 standards to meet the requirements (in particular, in terms of performance) of real-life industrial scenario.

3.2 Introduction

Nowadays, industrial companies are engaging their global transition toward the fourth industrial revolution (the so-called Industry 4.0). The main objective is to increase the (OEE), by collecting, storing and analysing production data. This digital shift brings a lot of requirements (e.g. in terms of network reliability or data-analytics) as detailed in [KWLJ17]. One of the most important requirement is the connectivity between all the components. Unfortunately, companies - *especially companies having decades of existence* - do not meet this requirement. Indeed, while we all dream about an ultra-connected ecosystem allowing the smooth transition toward this fourth industrial revolution, companies have still legacy, heterogeneous and proprietary systems and technologies making it even more complex.

To overcome smoothly this connectivity issue, industrial companies - *that have definitely the objective to digitize their industry* - may rely on IoT gateways for bridging the gap between the OT and IT worlds [SL11]. The gateways have to implement open and standardised Application Programming Interface (API)s so as to enable horizontal interoperability across vertically-oriented closed systems. The challenge to be tackled is therefore to collect and make available data from the production units in a real-time and standardised manner. This will enable to build efficient industrial applications (e.g. monitoring of production units) by retrieving those data in real-time.

To address this challenge, this study proposes i) a generic and reproducible strategy for defining an information model, that will be exploited for publishing

data coming from physical processes in a standardised way, and ii) an experimental performance evaluation of the standards OPC-UA, MQTT and O-MI/O-DF that are growing in the IoT world, as claimed by both industrial [RJ17] and academic communities. Note that, several studies have been led on those protocols with different purposes: i) for defining a roadmap to make the shift from Open Platform Communications (OPC)/Distributed Component Object Model (DCOM) to OPC-UA [HSK08], ii) for extending protocols with additional functionalities, such as a RESTful extension of OPC-UA [GPP16] iii) for conducting experimental evaluations [CCM10] in different scenarios and/or criteria [NDJ+16, PSK09, DdCGH16, NSP17, DCJ15, FFR+18, KRKLT17] iv) for ensuring a full interoperability at any level of the factory, by proposing to use the new Ethernet standards Time-Sensitive Networking (TSN) [BSB+19] or v) for proposing analytical models [RKLTF16, BRLTK18]. However, these papers do not propose neither a common methodology for collecting and publishing data into the IoT gateways, nor a common evaluation and comparison of the standards through real case-study and with the same criteria. That is why, after defining a strategy to digitise an industrial system, the strategy is evaluated on a real industrial pilot use case through a performance evaluation of the aforementioned standards. This case study is made possible thanks to the collaboration with CEBI Luxembourg S.A., a manufacturer of electromechanical components for automotive equipments and household appliances sector.

The rest of the chapter is organised as follows. The proposed methodology is detailed in 3.3. Our methodology is therefore assessed through the performance evaluation of the aforementioned standards in a real environment in 3.4. Finally, Section 3.5 concludes this study.

3.3 A strategy to digitise an industrial system

The purpose of this paper is to develop an effective strategy in order to digitise an industrial system (e.g. a production unit/cell) so as to make it Industry 4.0-ready. It will enable to offer production data to applications built on top of those digital shadows. We adopt a straightforward three-steps strategy called TRIDENT to incrementally enable the development of applications exploiting the process data and interacting with the industrial system. TRIDENT is voluntarily generic and high-level, not to say agnostic to a specific technique and/or protocol, since its goal is to provide a simple and common-sense structure for digitising a heterogeneous industrial system. TRIDENT has been successfully applied to our case study, and its simplicity is an advantage when interacting with stakeholders. As shown in Figure 3.1, the proposed methodology consists of several steps: 1) definition

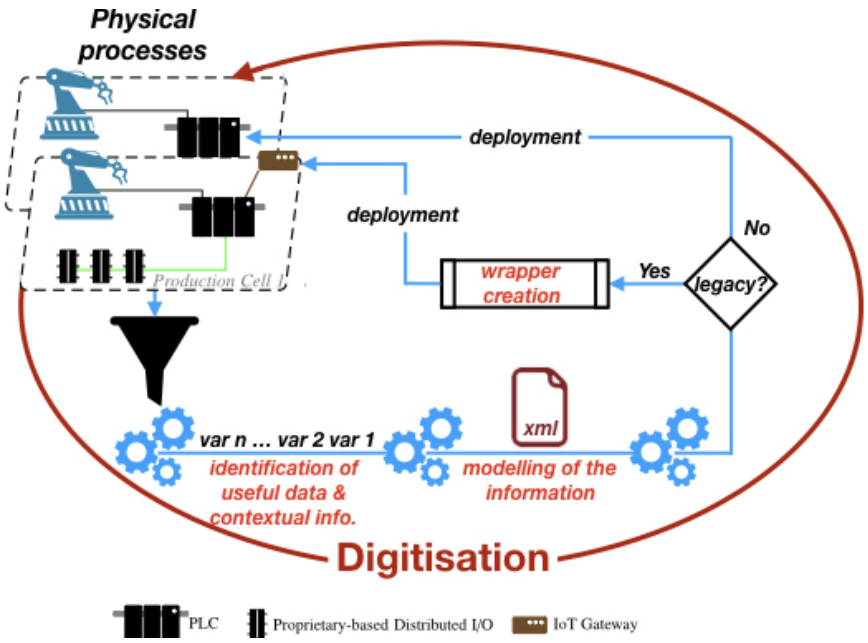


Figure 3.1: Strategy to digitise an industrial system

of an information model, 2) wrappers creation & deployment - *if needed* and 3) applications creation & deployment. The role of each steps are detailed in the following subsections.

3.3.1 Definition of an information model

TRIDENT suggests a classical first step to digitise an industrial system starting by establishing an information model of the system (that will be used by the digital shadow). This model relies on two models: i) a model of the industrial system topology (actuators, sensors and controllers), ii) a model of the processes running on it. Indeed, digital transformation cannot be done without an in-depth understanding of the current settings of the industrial system. The objective is to create a concrete (digital) representation that can be seen as an instance of a conceptual model (e.g. a meta-model or business model in the model-driven engineering terminology) with the relationships, constraints, rules, and operations specifying data semantics for Industry 4.0. TRIDENT suggests to have a current model corresponding to the current state of the factory and a targeted model (i.e. with all envisaged improvements in terms of new devices, information, ...). TRIDENT is model-agnostic, i.e. any information modelling technique (e.g. Unified Modeling Language (UML), Domain-Specific Language (DSL), ontologies) or any

process modelling language (e.g. activity diagram, statechart, Business Property Specification Language (BPSL)) can be used to obtain finally a structured and semantically rich information model.

In practical terms, it needs to make the inventory of current sensors/actuators/-controllers, the relevant data (in particular, the variables used in the controllers) and also the contextual information. Indeed, adding context will enable to make easier the automated reasoning that could be implemented in the applications. TRIDENT suggests that the (current and targeted) information model is defined in a structured way (e.g. in a hierarchical way), typically in a form of a file that can be manipulated and processed for the deployment on the physical process. The way to use this information model depends on the following aspects: i) if the process is already deployed or not, and ii) if the system on which it needs to be deployed relies on standards or proprietary technologies. In the first case, if the physical process is not yet implemented, this information model constitutes a mandatory element of specification that the manufacturer will leverage for implementation. In the second case, if the process is based on proprietary and closed solutions, the development of a wrapper is needed to be deployed on IoTs gateways. Otherwise, the information model can be deployed directly in the controller.

3.3.2 Creation of wrappers

The wrapper enables data translation from a proprietary to a standardised format. In some cases, it can also store data for different purposes (e.g. data logger at the edge). The main idea is to define *a minima* the two following microservices functions:

- Function 1 - *Update*: This function updates the data on the digital shadow from the data directly collected from the physical process. Usually, it is necessary to use proprietary protocol(s) for gathering data before translating them into the standardised data format selected for operating in the digital shadow.
- Function 2 - *Read*: This function responds to the users and/or applications request for reading related data. The type of request depends on the communication mode (client/server or publish/subscribe).

Let us note that those functions are related to the four basic functions of the Create, Read, Update, Delete (CRUD) model, that are usually used when building APIs. This model is also recommended by the Industrial Internet Consortium [RJ17] for managing the lifecycle of a data object. The first function 'Create' is related to the creation of the information model in the digital shadow. Its implementation will depend on the selected standard. The function 'Delete' is just a management operation of this information model, that can be managed by

the digital node, e.g. by using the information model file as input. In addition, those different functions can be located at different places in the Industry 4.0 architecture. In our vision, the wrapper is running on same device than the digital shadows (but it will depend on the resources available on the device, plus the requirement in terms of real-time).

3.3.3 Creation of applications

The creation of applications is depending on the communication mode implemented by the IoT gateway. In TRIDENT, since we access the data via READ functions offered by the wrapper (or directly by the industrial device), two communications modes have to be supported. The first communication mode is client/server (e.g. when using OPC-UA). In that case, the application requests in a regular basis some of the available data. The second mode occurs when the wrapper uses publish/subscribe mechanisms (e.g. when using MQTT). In that case, the application subscribes to the needed data, and will receive them in an event-driven manner (i.e. when data values change). TRIDENT requests to well understand the communication mode before developing any application, since it may impact the way to develop the temporary storage - *if needed* - for consuming such data. TRIDENT also recommends to properly timestamp the data that are collected in a uniform way, since data analytics may exploit timestamped information to analyse the industrial process and diagnose potential issues. Many industrial processes do collect the information in a non-uniform way (either timestamped by the controller, or by the data aggregator, or by the data consumer). Having a non-uniform and non-systematic way to timestamp data may lead to inaccurate information and further analysis. Indeed, those data can be used in multiple applications, e.g. in an User Interface (UI) for monitoring them, by an agent enabling to push them into a permanent database or even data analytics.

Next section demonstrates how this strategy has been applied in a real industrial case-study.

3.4 Industrial Case Study

In the framework of our partnership with CEBI Luxembourg S.A., we have access to the shop floor of the factory for deploying, testing and evaluating our strategy to digitise an industrial system. For this purpose, we selected one typical assembly line, that is not (yet) connected to the IT network. This assembly line consists of a PLC associated with sensors and actuators that are connected through a proprietary fieldbus (here, DeviceNet). All the information are processed

locally, either by the controller itself, or by local applications such as monitoring of the process through a Human-Machine Interface (HMI). The objective of this study is to apply our strategy on this specific application of monitoring by making available data in a standardised manner for this application, but also for future applications (e.g. analytics). Beyond presenting how to apply the strategy, performance is the first bottleneck for the applicability of such strategy and that is why we aim at evaluating the following protocols associated with an open available implementation: OPC-UA with two different versions open62541 developed in C & Eclipse Milo in java, MQTT with Eclipse Mosquitto and O-MI/O-DF with a java-based reference implementation (https://github.com/AaltoAsia/O-MI). We use an industrial IoT gateway (Kunbus Revolution PI). In addition, we tested a new PLC (OMRON NX1) that embeds an OPC-UA server that would enable i) to avoid building of wrappers and ii) potentially to make the next generation of assembly line interoperable with the IT network since relying on Ethernet.

3.4.1 Digitisation of an assembly line

Figure 3.2 shows the implementation of our strategy to digitise an assembly line of our partner. This section describes how to practically implement every step of the strategy presented in the previous section from a practical viewpoint:

Definition of an information model

As mentioned previously, it requires a deep understanding of the physical process (hardware and software). In our case, an assembly line is composed of several production stations that have a specific task to operate. Those kinds of information is what we called 'contextual information' in the previous section. Then, each station consists of several sensors and actuators that are listed in the PLC in terms of variables. Based on these both information, we built a hierarchical information model in the form of 'Machine number → station number → sensor/actuator name'. Note that metadata could be used to give additional information to a sensor value, e.g. unit or precision. From a practical perspective, the developed information model relies on the OPC-UA address space. Accordingly, we used the free software 'opc-ua modeler' to generate it and saving it as an XML file, that will be used as input of the server/broker running on the IoT gateway. For instance, this file can be directly used in the OPC-UA implementation 'open62541' that creates the information model (or address space in the OPC-UA terminology). However, for other implementations such as Eclipse Milo, we developed our own agent for processing the file to create the information model. In the specific case of MQTT, the information model (in the form of MQTT topics that are created from the eXtensible Markup Language (XML) file) is built when sending the first data update

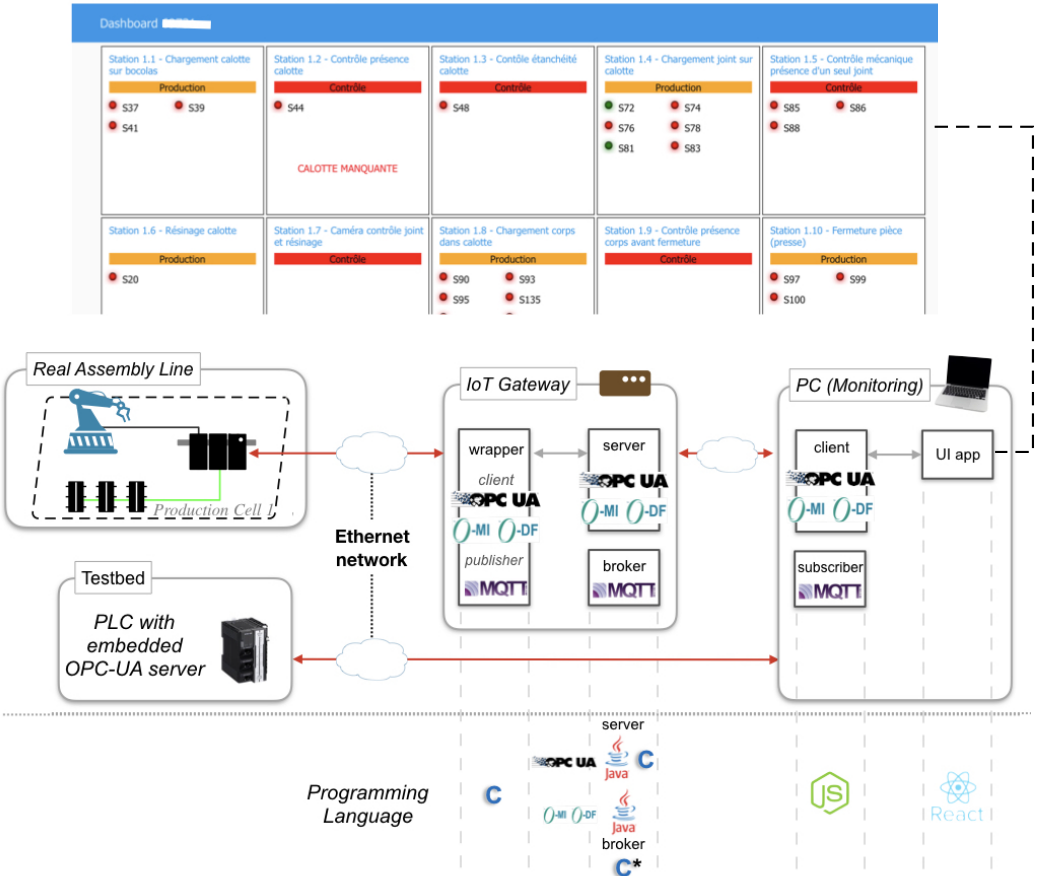


Figure 3.2: Implementation of our strategy to digitise an assembly line

managed by the wrapper. Finally, our information model for this assembly line consists of 64 variables, that have been considered as relevant. This information model could be extended in the future if needed. Regarding the embedded OPC-UA on the PLC OMRON NX1, the implementation of this information model is more complex. Indeed, it is impossible to select process variables for making them available through the OPC-UA server. Instead, we need to declare all the variables as global (i.e. that could be shared with the process) in a flat manner. This flat table will be added to a pre-defined hierarchical address space, meaning that the variables cannot be structured according to our information model, what constitutes a flaw in our perspective.

Wrappers creation & deployment

The objective of the wrapper is to collect data from the assembly line, and to translate them into the selected standards (i.e. OPC-UA, O-MI/O-DF and

Algorithm 1: Wrapper & Service Response Time (SRT) measurement

```
1  begin
2      Periodically
3          Start ← get(current time) in μs
4          Collect all the data from PLC
5          Update all the data on Server/Broker
6          End ← get(current time) in μs
7          SRT ← (End − Start)
8          if SRT < period then
9              wait(period - SRT)
10         else
11             continue // go to next period without waiting
12         end
13     end
14 end
```

MQTT in our experimentation). We developed all the wrappers in C. The main steps are i) to collect data every 100 ms (to meet at least the current specification of the monitoring application) by using the proprietary protocol (OMRON FINS in our case) and ii) to update those data on the server/broker, i.e. by sending write request on the OPC-UA and O-MI/O-DF server and publish message to the MQTT broker. Note that the various implementation of the servers and broker are using different programming languages that may have an impact on the overall performance and will be evaluated in the next subsection. Note that wrappers and servers/brokers are running on the same device (IoT gateway), meaning that exchange between them are made locally.

Applications creation & deployment

To create our monitoring application, we developed an agent (in node.js) that is able to gather all published data either by sending periodic requests to the server (OPC-UA or O-MI/O-DF) or by subscribing and receiving event data to/from the broker (MQTT). Then, a web-based UI has been developed relying on the REACT framework for displaying the monitored data. An overview of this interface is given in the Figure 3.2 (right part).

3.4.2 Experimental performance analysis

This section aims at evaluating the wrapper implementation associated with the standardised protocols OPC-UA, MQTT, and O-MI/O-DF on the aforemen-

tioned case-study. Our objective is to answer the following questions: i) Can we update the data on the IoT gateway at least every 100 ms as done with the current implementation (i.e. with the HMI directly connected to the PLC with a serial cable)? and ii) Can we refresh our UI at least at the same frequency? The first question is related to the wrapper efficiency whereas the second is related to the application efficiency.

Wrapper efficiency

For evaluating the efficiency of our wrappers, we focused on the SRT defined as the time taken for supplying the 'Update' service, as described by the algorithm 1. Note that, if the SRT is below the period (of 100 ms) (cf. line 8-9), we are waiting the difference between 100 ms and the recorded SRT to ensure that a data collection process is starting exactly every period. On the contrary, if the SRT exceeds the period (cf. line 10-11), we directly execute a new loop to minimise the jitter between two data collection. Figure 3.3 provides an aggregated view of the results for both different 12-minutes experiments, as a boxplot that provides the minimum, 1st percentile, median, 99st percentile, and maximum of the SRT measurements. Data have been cleaned by removing the measurement bias/outliers, that represents only 7 measures out of 7200 (i.e. less than 1‰) in each experiment. In addition, we used *tcpdump* for capturing network traffic so as to gain knowledge about the concurrent traffic (since we are plugging our IoT gateway on the existing embedded Ethernet network without any knowledge) and the throughput generated by our digitisation chain. The following conclusions can therefore be drawn:

Concurrent traffic: The analysis of the captures shows that few traffic is existing on this network. Indeed, there are only 'real-time' traffic between controllers (via the Ethernet/IP) representing only 26 $kbits/s$ and few management traffic (e.g. Address Resolution Protocol (ARP), Dynamic Host Configuration Protocol (DHCP), ...) representing less than 300 $bits/s$. This traffic load (wrt. the network capacity equals to 100 $Mbits/s$) will have very little impact on the data collection (wrapper).

Throughput for data collection: it corresponds to the requests/responses made with the proprietary protocols (Factory Interface Network Services (FINS)) to collect all the data (i.e. 64 *variables*) every 100 ms. Overall, the measured throughput generated by our wrapper is 32 $kbits/s$, that is relatively low compared to the network capacity. A typical DHCP frame size (request or response) is between 70 and 94 *bytes* (taking in average 7 ms as round-trip time) depending on the number of variables requested at the same time (aggregation mechanism). Indeed, if the variable numbers are not consecutives (depending on how the PLC variables have been defined), we need to use several frames (3 in our experiments) for collecting all the selected data. The throughput is the same whatever the

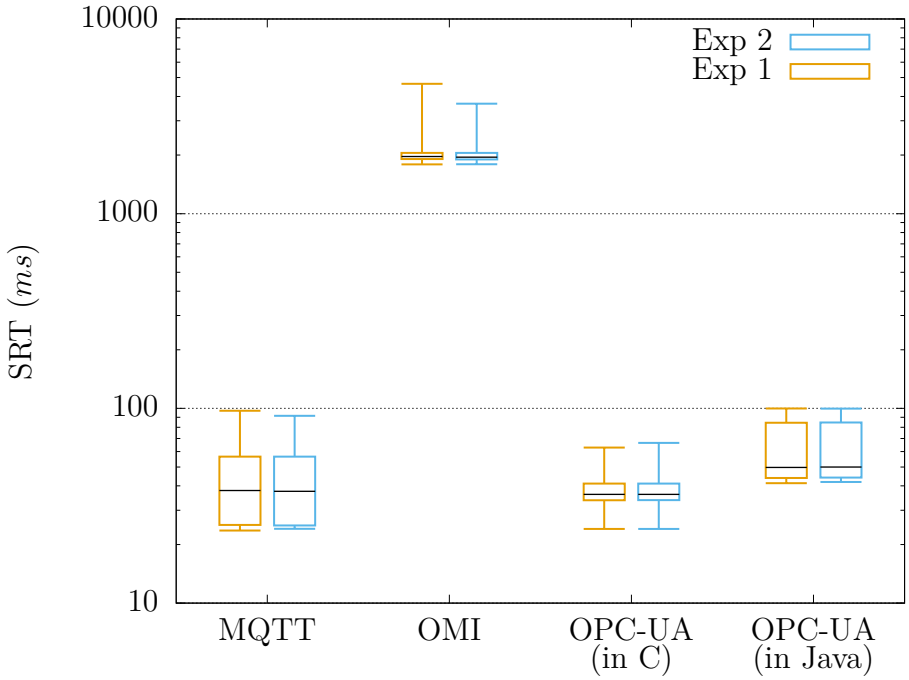


Figure 3.3: SRT measurements on the wrapper (note the log-scale on the y-axis)

protocols used for updating the IoT gateway.

Protocols efficiency wrt. 100 ms-period:

- O-MI/O-DF: According to the experiments, this standard is not suitable for the considered industrial scenario since it does not meet the specified expectations in terms of SRT (largely greater than 100 ms) in both experiments (cf. Figure 3.3). Our first investigations highlight that the embedded database is the bottleneck of this implementation (at least for the version we used). Note that O-MI/O-DF was designed for meeting IoT requirements, that most of the time do not need to decrease the update time below seconds.
- OPC-UA: Both versions of the server (i.e. open62541 in C and Eclipse Milo in java) are suitable for this scenario, since all measures are below to 100 ms. However, it can be noticed that the C version gives better results than the java version since 98% of the values are in the $[24; 41]$ ms range (compared to $[41; 84]$ ms range provided by the java version). Note that the aggregation mechanism (at the application level), enabling to embed several variables (as much as in the DHCP requests) in the sameOPC-UA request(s), have been used in those experiments.
- MQTT: the results we have obtained are also satisfying for this protocol. All points are above 100 ms with 98% of the values are in the $[23; 56]$ ms range. As MQTT implements a publish/subscribe mechanism, each variable is sent

to the broker by using one 'publish' message. Despite the higher number of frames (compared to OPC-UA with the aggregation mechanism), the broker is able to manage them.

Overall, only OPC-UA (all versions) and MQTT standards are selected for going further in the analysis, and in particular from the application perspective.

Application efficiency

The analysis of the application efficiency is split in two parts: the first one focuses on the evaluation done on the real assembly line (legacy system), whereas the second is based on a testbed consisting of the PLC OMRON NX1 that is under test for a potential use in the next generation of assembly line.

*** legacy system**: Instead of looking at the SRT, we are interested to the inter-arrival time, i.e. time elapsed between the two consecutive receptions of the same variable. The choice enables the comparison between protocols. It is not possible to compute a SRT (where the service is a 'Read/Retrieve') in the case of MQTT, since it relies on the publish/subscribe mechanism. Indeed, once subscribing to each variable, the application does not need to generate other request; the broker pushes values to the application each time it receives a new one (publish by the wrapper). In addition, OPC-UA is here considered when giving the best results, i.e. by implementing the aggregation mechanism (at the application level). Figure 3.4 provides an aggregated view of the results for both 12-minutes experiments. Based on those results associated to the network captures, the following conclusions can therefore be drawn:

Throughput for data reading: it corresponds to either the requests/responses made with OPC-UA to read all the data on the IoT gateway or the MQTT publish messages sent by the IoT gateway. Overall, the measured throughput is 210 $kbits/s$ when using OPC-UA with aggregation mechanism at the application level (as previously explained) and 550 $kbits/s$ in average when using MQTT. The throughput of MQTT varies (between 469 and 631 $kbits/s$) because of the Nagle's algorithm (implemented on Linux-based systems by default). Indeed, this algorithm enqueues all new data received by the TCP layer in the same segment until the TCP- acknowledgement of the previous segment is received.

Protocols efficiency wrt. 100 ms-period: Ideally (i.e. without network delay, processing time, …), the inter-arrival time is equal to the 100 ms period. However, in real applications, those parameters impact the inter-arrival, as shown in Figure 3.4.

- MQTT: 98% of the values are between 84 and 116 ms. Indeed, those inter-arrival times are directly linked with the performance of the wrapper itself, since the data are pushed to the application each time the broker receive a new data from the data sources (i.e. our wrapper). It can be noticed that some values are below 100 ms, which can be explained by the waiting time

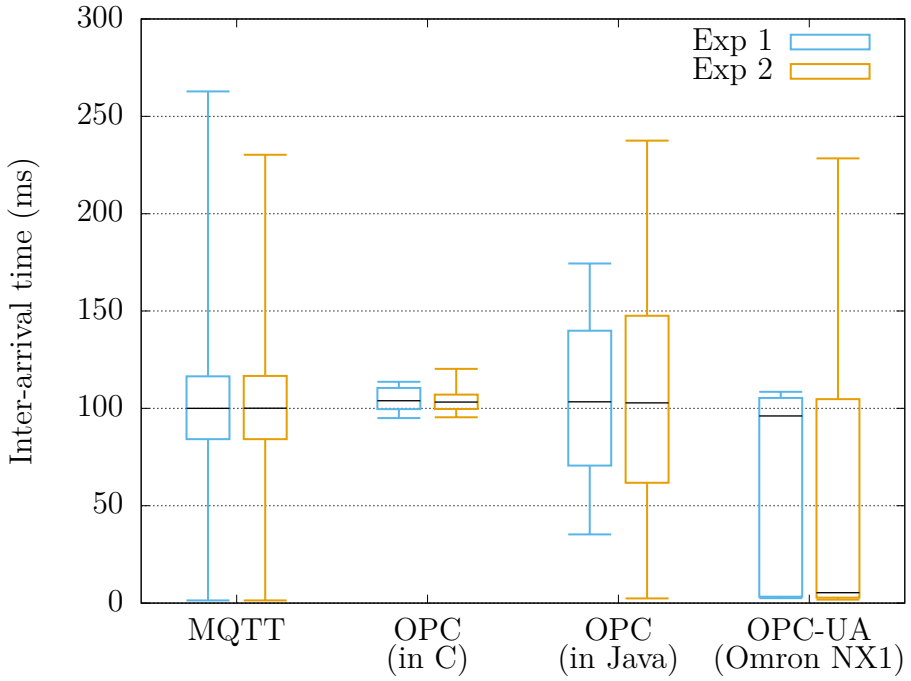


Figure 3.4: Inter-arrival measurements on the application

(less than a period) defined in the algorithm 1 line 8-9. The inter-arrival time will be more important than the period in the next loop. However, if the SRT exceeds the period of 100 ms, we can obtain higher inter-arrival values as seen with the maximum (around 220 ms) in the Figure 3.4. Note that this situation has been rare in our experiments (only one time).

– OPC-UA: The server developed in C is able to respond quickly to all the requests since the 98% of the inter-arrival times are in the $[100-110]$ ms with a maximum of 120 ms. The server developed in java has more difficulties to process all the requests (between 70 and 140 ms). Note that it is possible to have values below 100 ms since, as for the wrapper (algorithm 1 line 8-11), the application code starts a new loop if the previous loop took more than one period. Even if these situations have been rare as well for OPC-UA, we wanted to highlight this fact, that may occur in this kind of infrastructure.

Overall, open62541 implementation of OPC-UA has best results in terms of inter-arrivals times, MQTT and Eclipse Milo (OPC-UAs in java) are also suitable for this industrial applications due to the scarcity of the non expected situations.

*** next generation of assembly lines**: As previously, we measured the inter-arrival time by requesting data with the aggregation mechanism. It can be noticed that the variation of this inter-arrival time is important for both experiments (between 3 and 105 ms). This shows that the server takes a long time to process

the request (almost 100 ms) leading to generate the next request shortly after the previous response (since we use the same code in the application than lines 8-11 in the algorithm 1). In addition, in this experiment, since there was no industrial process/communications, one may wonder whether the PLC would be able to manage the both tasks simultaneously (or if one task would have higher priority than the other). This investigation will be achieved in the future work. But, this already shows the weaknesses of such device to manage the Industry 4.0 requirements.

3.5 Discussion and Conclusion

Industrial companies are engaging their global transition toward the Industry 4.0, with the objective to increase the OEE. This can be done by collecting, storing and analysing production data. However, companies need to deal with legacy, heterogeneous and proprietary systems. Therefore, they are facing difficulties regarding technological/implementation choices in their digital shift. That's the reason why they need to rely on a suitable and application-oriented strategy to digitise an industrial system, as proposed in this paper. The underlying components and standards have to meet the applications' requirements, particularly in terms of performance. Beyond the choice of the standard, one important result of this study is the need to well select the hardware that will be implemented in the next generation of assembly lines. Indeed, the IT/OT convergence leads the manufacturers to integrate new functionalities in their devices (e.g. the OPC-UA server embedded in PLC OMRON NX1). Unfortunately, the first results obtained on the PLC under test show that the hardware resources are not sufficient to manage efficiently the application traffic load. Next step would be to define a real process, to check whether the PLC can still manage the both tasks (process and data management) or not.

In this study, the strategy has been applied to a specific real-world assembly line from our partner CEBI Luxembourg S.A., but suits all the industrial applications (results may differ according to the architecture and traffic). Only a part of the whole factory information model has been defined by focusing on a specific application (local monitoring). Note that the paper already shows that OPC-UA is a promising solution for implementing it. Future work will focus on the extension of this information model for taking into account new applications (in particular, data analytics) and assembly lines. This will lead to design a complete network infrastructure by integrating edge, fog and cloud computing while considering solutions ensuring a full interoperability and real-time requirements such as TSN.

Part II

Robustness evaluation of Machine Learning models in industrial settings

4

A Systematic Approach for Evaluating Artificial Intelligence Models in Industrial Settings

Foreword

The first part of the dissertation addressed the challenge of designing and evaluating a unified data-driven strategy in presence of heterogeneous data producers by relying on open communication standards, thus providing interoperability between vertically closed silos. Chapter II and III demonstrated how to build a unified data-driven approach for industrial production plant. To reach a full Industry 4.0 readiness (as defined in the RDI project between SnT and Cebi Luxembourg S.A.) it is also necessary to introduce ML methods for data analysis tasks. Industrial actors tend to be interested in ML approaches to help them analyse their production data and make decisions on top of it. Indeed, the digitisation of the industry using unified data-collection architecture allows the acquisition of such volume of data that becomes challenging to be analysed by human beings. In this kind of configuration, most of the data collected from the production floors are produced by the different sensors present on the machines and are collected/stored as time-series, and more specifically Univariate Time Series (UTS).

Previous works concerning the use of ML algorithms for UTS TSC tasks have delivered encouraging results in terms of performance and accuracy over time, *thanks to publicly available data sets (e.g., University of California Riverside (UCR)) that helps ML community improve their models.* However, industrial actors still need to get empirical evidence that the outputs (classification of UTS in our case) of these algorithms are accurate and can be used in such an industrial environment with its own expectations and constraints. Indeed, ML algorithms can have many perks in terms of performance for data analysis tasks, but require a good data-quality to produce accurate classification. Earlier studies have already highlighted that data-quality can affect the results of such stochastic approaches, and have tackled this problem in many domains (e.g., for image recognition), sometimes by relying on complex methods and solutions, that require solid ML knowledge to apply to domain specifics: many companies do not have such a solid knowledge of ML techniques. In the specific case of UTS classification in industrial settings, the problem has not been tackled and has still to be studied, since the **robustness** of ML is critical when it is used in industry, especially in industries for which consequences of misclassification can be tragic. Indeed, we demonstrated in Part I of the dissertation that even with a digitisation strategy that provides interoperability and open communication interfaces, performance can be tempered by different factors and then alter the data-quality. In chapter II ([BRLTK18]) we highlighted that depending on the data format/hierarchy and the data aggregation mecha-

nism used in the implementation of the protocols, the traffic load generated on the network can be highly heterogeneous, and this heterogeneity on the traffic load can grow in case of data losses. In chapter III ([BRLT19]), we evaluated different implementations of our generic strategy to digitise an industrial production plant and the results showed that time performance are sparse, depending on the protocol that is used, its communication mode (i.e., publish/subscribe or request/response), its programming language (C, Java) and even the hardware hosting the different agents (gateway or new generation of PLC). Also, industrial actors will continue to upgrade and to integrate new machines in their production plants to cover the upcoming customers needs. In addition, they will be continuously willing to enhance their production tools and data-analysis tools by enlarging the use of ML methods for analysing an increasing volume of data. This data-driven evolution of the production tool will increase the traffic-load on the network and can generate perturbations during the data-collection phase. Overall, Overall, even without considering any other factor but only network traffic and its impact, the question of the robustness of ML models in an industrial environment is critical and has not yet been studied.

Since there is no existing systematic approach to evaluate the robustness of ML classifiers for UTS TSC tasks, that would help industrial actors trust such techniques, we decide to design a reference use case and to evaluate it. It intends at being aware of the potential misclassification/weaknesses of the models, which can be the consequence of a degraded data-quality, likely to occur in such a data-driven production system prone to perturbations. More importantly, such an evaluation also helps know the limits of the models and their domain of validity. One of the most important aspect of this approach is that it is a systematic and reproducible method. It is not feasible to evaluate directly the system in production, since 1) the real data may not be degraded at the time of the evaluation, 2) the goal of the validation is to anticipate the risk for the system to fail in case of unforeseen but realistic cases. As a consequence, we propose to generate artificially such realistic perturbations and injecting them into the data sets. This approach is similar to adversarial testing of ML models, except that we target perturbations that mimic possible situations. This allows to cover a large amount of potential perturbations scenarios in a systematic and automatic manner, without stopping the production lines, which would result in production and money losses, while being detrimental for the production-line operations. As a matter of fact, a massive production system cannot be stopped for a long period in order to intentionally generate errors and dysfunctions for the sake of a

study, whereas it is critical for stepping into the next industrial revolution. Then, the method we further propose aims at providing the different steps that are required for evaluating ML models under perturbations and can be used before a future implementation of ML models in a production system, or to prevent the impact of the evolution of the production plant (e.g., when increasing the number of connected devices, models, protocols, ...). As we do not have an operational data set of a production-line data collection process at the moment of proposing this systematic approach, since our partner had not deployed a production-ready unified data driven architecture yet. As a consequence, we evaluate our methodology on state-of-the-art UTS datasets, thus allowing the research community to reproduce our findings, while allowing our partner to understand the benefits of such a robustness systematic study ML models.

This chapter presents our study and has been published as a peer-reviewed journal paper titled *"A Systematic Approach for Evaluating Artificial Intelligence Models in Industrial Settings"* [BRLT21] in MDPI Sensors journal, Volume 21, Issue 18, 2021.

Contents

4.1 Abstract

AI is one of the hottest topics in our society, especially when it comes to solving data-analysis problems. Industry are conducting their digital shifts, and AI is becoming a cornerstone technology for making decisions out of the huge amount of (sensors-based) data available in the production floor. However, such technology may be disappointing when deployed in real conditions. Despite good theoretical performances and high accuracy when trained and tested in isolation, a ML model may provide degraded performances in real conditions. One reason may be fragility in treating properly unexpected or perturbed data. The objective of the paper is therefore to study the robustness of seven ML and DL algorithms, when classifying UTS under perturbations. A systematic approach is proposed for artificially injecting perturbations in the data and for evaluating the robustness of the models. This approach focuses on two perturbations that are likely to happen during data collection. Our experimental study, conducted on twenty sensors' datasets from the public UCR repository, shows a great disparity of the models' robustness under data quality degradation. Those results are used to analyse whether the impact of such robustness can be predictable—*thanks to decision trees*—which would prevent us from testing all perturbations scenarios. Our study shows that building such a predictor is not straightforward and suggests that such a systematic approach needs to be used for evaluating AI models' robustness.

4.2 Introduction

Nowadays, with the advent of the IoT and IIoT, public and industrial actors are leveraging these technologies to enhance their systems while satisfying new requirements entailed by such a social revolution [GBMP13, GCL18, AIM10, BS15, VT14, KWLJ17]. Focusing on the industrial context, with the industry 4.0 (r)evolution, companies want to meet new business goals by increasing the OEE. In the meantime, their customers are increasingly demanding better quality, flexibility or even security. In order to tackle these new challenges, industrial actors are looking at exploiting unused data coming from their production's systems. The amount of data is unprecedentedly huge, making it difficult for humans to analyse and make decisions quickly and efficiently. That is the reason why AI is being increasingly used for solving a large range of problems and applications, e.g., the TSC problem, which is one of the most common in the industry and also recognised as one of the ten listed problems in data-mining researches [YW06].

Using AI requires good data quality whatever the applications. Although data-quality consideration strongly depends on the end-users or use-cases needs, it

should be specifically considered for each data-driven system [WS96, FF19, BBET+15] to avoid regrettable experiences, e.g., a pedestrian killed by a self-driving car in Arizona [ped18]. Indeed, by learning from inaccurate or inadequate datasets, the downstream results can be flawed and lead to an inaccurate analysis of the data, resulting in inappropriate actions from the different actors [LDSP18, GAD17, SLS+18]. In industry, there is a lot of situations where data quality can be degraded throughout the production system's entire lifetime. Beyond the ageing of the sensors, the whole data collection infrastructure may introduce some perturbations. This is all the more true for companies with decades of existence that rely on legacy industrial architectures where data producers (i.e., sensors, PLCs, etc.) are heterogeneous, requiring middleware to access and standardise data, while providing an interface between the business and industrial worlds [SL11]. Such data collection infrastructure may have sparse performance, in particular in terms of network metrics such as losses, delays or traffic-load [BRLT19, BRLTK18], resulting in data quality degradation.

Unfortunately, most of the studies – tackling the TSC problem using AI – do not consider such data quality degradation scenarios when evaluating their algorithms. Researchers are indeed focusing on improving the algorithms performance especially in terms of accuracy or even response time [SL17]. Performances are evaluated on public benchmark datasets such as the UCR datasets ([DBK+19] (`https://www.cs.ucr.edu/%7Eeamonn/time_series_data_2018/`, (accessed on July 2021))). Such datasets are then considered as clean and without any biases. Although sthese evaluations are needed, this is not sufficient to be confident in the robustness of these algorithms/models in case of perturbations (that may happen temporarily or gradually over time).

The objective of the chapter is therefore threefold: (i) to propose a systematic approach, inspired by mutation testing techniques, for artificially injecting two types of perturbations in benchmarking datasets, (ii) to evaluate the impact of such perturbations on 7 state-of-the-art algorithms and 20 sensor-based datasets and (iii) to analyse whether such impact can be predictable or not without testing all perturbations scenarios.

The chapter is organised as follows: Section 4.3 presents the related work regarding AI that consider perturbations. In Section 4.4, a systematic approach for evaluating AI models under perturbations is developed. Then, in Section 5.4, the systematic approach is therefore assessed by experiments on two realistic perturbations, called hereafter swapping and dropping perturbations. Section 4.6 aims at trying to predict the robustness of models by using decision trees. Finally, Section 5.5 presents the conclusion of the chapter.

4.3 Background and Related Work

AI is a domain that includes a lot of techniques to tackle a large range of problems and applications. Focusing on the *TSC* problem, ML techniques are beginning to be the new standard in recent industrial systems, and DL also tends to be adopted in certain cases ([WMZ+18, LCA+16, WLS+19, LDSP18]). Looking at the definition of a Time Series (TS) in the literature, authors use different ones depending on the context [SL17, MRB+18, FFW+19] while being quite similar. It is worth mentioning that it is, nonetheless, important to define it, as pointed out in [Gam17]. In this study, we define it as follows:

A time series TS is an ensemble E representing a sequence of N data-points e_n, assumed as equally distributed: $E = [e_1, \cdots, e_N]$.

Our literature review is intended to analyse to what extent research work (in ML and DL) are evaluating "the degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions", defined as robustness in IEEE standard glossary of software engineering terminology [15990]. In the AI context, "robustness (therefore) measures the resilience of a ML system's correctness in the presence of perturbations" [ZHML20]. Based on these definitions and the aforementioned literature review objective, we applied the following three-step methodology for selecting papers to analyse: (i) keep only papers dealing with the TSC problem. Our corpus consists of 1417 papers collected in seven main library databases: IEEE Xplore, ACM Digital Library, Springer, ScienceDirect, MDPI, Taylor and Francis and Wiley; (ii) filter papers mentioning perturbations (or related terms, e.g., robustness, adversarial, data inconsistency). Only 35 papers were remaining, and (iii) they were filtered through a careful reading. Finally, 14 papers are presented and listed in Table 4.1, while summing up (as in Table 4.1 footer) with:

- column "Approach": the approach (ML or DL) used in the research work;
- column "Method": the methods/algorithms used or analysed;
- column "TS Type": the type of TS, i.e., either UTS or Multivariate Time Series (MTS);
- column "Perturb. Model": the type of perturbations model;
- column "Reproducible?": if such analysis is reproducible (can we recreate ourselves datasets with perturbations according to predefined parameters);
- column "Public repo?": if such datasets before/after perturbations are publicly available.

Table 4.1: Related work.

Paper	**Approach**		**Method**	**TS type**		**Perturb. Model**	**Reproducible?**	**Public repo?**	
	M-L	D-L		UTS	MTS			initial	modified
[HL18]	✗	✓	CNN	✓	✗	Random Noise	✗	✗	✗
[TMH17]	✓	✗	SVM,1-NN, DT, RF	✗	✓	Data Loss	✓	✓	✗
[CJM+19]	✓	✗	XG-Boost	✓	✗	Missing Data	✗	✗	✗
[MSJR19]	✓	✓	SVM, DT, RF, NN, CNN	✓	✗	Random Missing Data	✗	✗	✗
[IFW+19]	✗	✓	ResNet, FCN	✓	✗	Noise	✓	✓	✗
[KMM18]	✗	✓	CNN	✓	✓	Missing Data	✓	✓	✗
[YPT19]	✓	✗	ARM-SONS	✓	✗	Missing Data	✗	✓	✗
[ZH13]	✓	✗	BPSO, IBPSO, INSIGHT	✓	✗	Random Noise	✗	✓	✗
[YWS+19]	✓	✗	DTW	✓	✗	Missing data and Noise	✓	✓	✗
[PAK+13]	✓	✗	ANN, SVM, SSL	✗	✓	Noise	✓	✓	✗
[ODPL20]	✓	✓	OSVM, DNN	✗	✓	Color Perturbations	✗	✓	✗
[NKK17]	✓	✓	STRiD, NN, SVM, ID3	✗	✓	Missing Data	✓	✓	✗
[MBSRJ18]	✓	✗	1-NN	✓	✓	Missing Data	✓	✓	✗
[dRSd+17]	✓	✗	BoW+SVM	✓	✗	Noise & Artifacts	✓	✓	✗

✓: original data set contains the perturbation

"Approach": the approach (M-L or D-L) used in the research work, **"Method"**: the methods/algorithms used or analysed, **"TS Type"**: the type of time-series (TS), i.e. either Univariate Time Series (UTS) or Multivariate Time Series (MTS), **"Perturb. Model"**: the type of perturbations model, **"Reproducible?"**: if such analysis is reproducible (can we recreate ourselves datasets with perturbations according to predefined parameters), **"Public repo?"**: if such datatets before/after perturbations are publicly available

***CNN**: Convolutional Neural Network, **SVM**: Support Vector Machine, **1-NN**: 1- Nearest Neighbors, **DT**: Decision-Tree, **RF**: Random Forest, **XG-Boost**: eXtreme Gradient Boosting , **NN**: Neural Network, **ARM-SONS**: Sparse Online Newton Step for AR with Missing Data, **BPSO**: Binary Particle Swarm Optimization, **IBPSO**: Immune Binary Particle Swarm Optimization, **DTW**: Dynamic Time Warping, **ANN**: Artificial Neural Network, **OSVM**: One-class Support Vector Machine, **SSL**: Semi-Supervised Learning, **DNN**: Deep Neural Network, **STRiD**: Statistical Tolerance Rough Set induced Decision tree, **ID3**: Iterative Dichotomiser 3, **BoW**: Bag of Words*

This highlights that there are only a few works that evaluate the robustness of models by providing a reproducible model of their (natural) perturbations or a (artificially) perturbed dataset that is publicly available. Only 3 studies (i.e., [IFW+19, TMH17, MBSRJ18]) out of 14 (artificially) generated perturbations to modify the datasets and then perform experiments on the models. Other listed studies do not provide a fault model that is reproducible, or they use datasets that are known as containing some perturbations (noise or missing data) but without identifying the characteristics of these perturbations (making it difficult to reproduce on other datasets for comparison purposes). Concerning the studies in which perturbations' models aim at modifying the data, the experiments are focused on the perturbations models and evaluating few algorithms, but they do not analyse to what extent the perturbations impact the performance of the model itself. Moreover, even if research solving some adversarial robustness problems exists, adversarial robustness can lead to a decrease in accuracy when no perturbations are present ([TSE+18]).

Based on those facts, we decided to study the robustness of ML/DL models under perturbations, which we defined in Section 4.4 and allows anybody to reproduce them for benchmarking purposes. To do so, we selected two different works as the baseline:

- The algorithm presented in [SL17], called The Word ExtrAction for time SEries cLassification (WEASEL)—as an ML solution, which obtains the best accuracy of the former algorithm on most of the public UCR datasets,
- The framework/algorithms presented in [FFW+19], which can be used as a black-box in the DL category and showed great performance on UCR datasets.

In addition, as we focused on industrial scenarios, only sensor-based datasets (UTS) were used for our experiments (20 in total, presented in the following section).

4.4 A Systematic Approach for Evaluating AI Models Under Perturbations

In this section, we intended to define a systematic approach for evaluating AI models under perturbations that could appear over the AI models' lifetime. In that sense, we assume that the AI models are trained in "normal conditions" it does not mean that the data are clean and without any biases but reflects the normal behaviour of the data collection infrastructure at a time of the model training. Usually, test datasets are also collected in the same conditions—even if it is not clearly mentioned, the characteristics of the training and testing datasets are simi-

lar. There is an important literature in ML that demonstrates that trained models may not be robust to corner cases (e.g., adversarial cases) despite a high accuracy. We thus believe that trained models shall be tested against perturbation to assess and improve their robustness in situations that may occur in a realistic setting. It also important to include potential derivations in the data that could appear over time or in "degraded conditions", so as to evaluate their robustness. Our objective is therefore to generate perturbations (on test datasets) that are not too far from the reality and more importantly reproducible—either on the same datasets or in a similar way on other datasets—so as to be able to benchmark/compare the robustness of AI models under such perturbations. In this approach, we define two kinds of perturbations:

- *The swapping perturbation*: the sequence of the N data points e_n is altered/not respected. It is a realistic situation in several settings, e.g., when using User Datagram Protocol (UDP) as transport protocol for data exchanging between sensors and controller, since UDP does not enable re-ordering of the packets in the network and/or if the timestamping of the data can only be performed on the controller side (sensors usually do not have the capacity of timestamping).
- *Dropping perturbation*: some data points in the time-series are missing. As for the swapping perturbations, this is realistic as a network protocol such as UDP does not enable packet retransmission in the case of loss, or when software processing data reception has memory overflow (especially when processing a huge amount of sensors data in constrained devices, such as raspberry-like devices), which can also lead to such losses.

Since every (industrial) environment is different, it is usually difficult to identify the perturbations existing in such environment. That is the reason why our perturbations require to be parameterised using a suitable mathematical and systematic form. It enables, in particular, varying the parameters so as to identify the limits of the robustness of the AI models in the experimental analysis. As a matter of fact, let us formally define the perturbations:

- *The swapping perturbation*: Let us first reiterate that a time series TS is an ensemble E representing a sequence of N data-points e_n, assumed as equally distributed: $E = [e_1, \cdots, e_i, \cdots, e_j, \cdots, e_N]$. A swapping perturbation is therefore a pair of data points that are interchanged/swapped. In the case where only one pair has been swapped, the time-series becomes $E' = [e_1, \cdots, e_j, \cdots, e_i, \cdots, e_N]$, where the events e_i and e_j have been swapped. However, swapping only one pair would probably not have an impact. We therefore define two parameters:
  - Pe as the percentage ($0\% < Pe < 100\%$) of swapped events/values in each time-series of a dataset. It means that $S = N * \frac{Pe}{100}$ values will be

randomly interchanged in a TS of length N.

– R as the range in which the value is swapped (i.e., at which position the data are moved in a certain range of possibilities). For instance, let $R = [1, 2]$, which means that if we randomly pick 1. as the position to be changed, an event e_4 is interchanged with the event e_3 (and vice-versa).

To apply such swapping perturbations on all the time-series in a dataset D, we define the function presented in Algorithm 2, which gives a new dataset D' as the output. Note that a dataset D consists of a set of times-series E_k with the same length—i.e., the number of data points N (as it is usually the case in public benchmarking repositories)—, such as $D = [E_1, \cdots, E_M]$ with M the number of TS in D. The newly created dataset D' consisting of a set of times-series E'_k has the same features (in particular, the number of TS and of data points per TS) than D. Finally, to keep it as generic and open (for experimentation) as possible, no probability distribution is imposed in the random processes used in this algorithm.

Algorithm 2: Swapping perturbations function.

input : D, Pe, R
output : D'

Function `Swap`$(E, pos1, pos2)$**:**
  $e_{pos1} = e_{pos2}$
  $e_{pos2} = e_{pos1}$
  return E;
begin
  $S \leftarrow \lfloor N * Pe \rfloor$ `// No. of values to swap`
  for $i \leftarrow 1$ **to** M **do**
    $indexS \leftarrow randomSelection(0, N, S)$ `//` S `indexes in the TS`
    for $j \leftarrow 1$ **to** S **do**
      $pos \leftarrow$ `randomSelection(`R, *1*`)`
      $E'_i \leftarrow$ `Swap(`$E_i, indexS_j, indexS_j - pos$`)`
    end
  end
end

- *Dropping perturbation*: A dropping perturbation is the consequence of a data loss (e.g., between a sensor that has sent the data to be stored and the controller that has to store the data). Formally speaking, it means that a time-series $E = [e_1, \cdots, e_N]$ is becoming $E' = [e_1, \cdots, e_{N-Q}]$ where the length of the time-series is decreased in terms of the number of lost/deleted events Q ($Q < N$). However, in that case and particularly in practice, E' will

not strictly follow the definition of a TS where the data points e_n are assumed as equally distributed over time. This means that all the indexes will be only shifted. To be consistent, mathematically speaking, we propose to have a reconstruction mechanism to fill out a missing value (e.g., in practice, the controller knows that it should receive a value periodically, so it can compute a value when detecting a missing value). Even if such mitigation mechanism can limit the impact on the models' robustness a priori, it is important for us to consider it in a formal way, since in a real environment, it would be probably and easily implemented. We nonetheless keep the term "dropping" since it is the origin of the expected perturbations. Based on these choices, we define a dropping function presented in Algorithm 3. It also takes the percentage Pe of removed events/values in each time-series of a dataset D as a parameter. Similarly to the swapping function, the indexes of the dropped (and reconstructed) elements are randomly selected. Note that, after reconstruction, the length of E' is equal to the one of E. Finally, to keep it as generic and open (for experimentation) as possible, the method for recomputing a value at the dropped values positions is not imposed by the algorithm as such.

Algorithm 3: Dropping perturbations function.

input : D, Pe
output : D'

Function `Drop`(E, pos)**:**
  $e_{pos} \leftarrow deleteAndReconstructElement(e_{pos})$
  return E;
begin
  $Q \leftarrow \lfloor N * Pe \rfloor$ `// No. of values to drop`
  for $i \leftarrow 1$ **to** M **do**
    $indexQ \leftarrow randomSelection(0, N, Q)$ `//` Q `indexes in the TS`
    for $j \leftarrow 1$ **to** Q*)* **do**
      $pos \leftarrow indexQ_j$
      $E'_i \leftarrow$ `Drop(`E_i, pos`) // drop value in` i^{th} `time series`
    end
  end
end

4.5 Robustness Evaluation based on Perturbations Generation

The approach presented in the previous section is applied for evaluating the robustness of AI models trained with seven algorithms: six DL algorithms (referred to as Fully Convolutional Neural Network (FCN), Residual Network(ResNet), Convolutional Neural Network (CNN), Multi Channel Deep Convolutional Neural Network (MCDCNN), Multi Layer Perceptron (MLP) and Time Le-Net (Tlenet)) proposed as a framework by [FFW+19] and one ML algorithm (WEASEL) developed and evaluated by [SL17]. This evaluation is achieved on 20 sensor-based datasets available in the public UCR (https://www.cs.ucr.edu, accessed on July the 5th, 2021) repository. These were selected since the data looks similar to the ones we can find in industrial scenarios, when a sensor delivers an UTS. The algorithms are selected for two reasons: (i) they are freely and publicly available to be used as a blackbox, and (ii) they give good performance on the selected datasets to be served as a baseline of our work.

4.5.1 Methodology of the Evaluation

Let us first present the methodology used for this evaluation:

1. The *preparation phase* consists of training models with the available datasets and then generating datasets with perturbations that will be used for evaluating the robustness of the models in phase 2.
   (a) Retrieve the 20 selected UTS datasets of the sensors' type (from the UCR repository).
   (b) Train models using the different selected classifiers on the previously collected datasets. This phase is needed since the models (of existing researches) for the benchmark datasets are not available publicly (and the hardware as well as the software can impact the models' accuracy, especially when DL is used). Since training a DL model several times can lead to different accuracy results (even with the same parameters and dataset), we trained each pair 5 times <classifier; dataset> (it was enough to reach the same—or even better—accuracy as the existing benchmark). In total, 700 models (5 iterations for 7 classifiers on 20 datasets) were trained. In this training phase, the objective is to obtain the best classifier/model that could be deployed in real settings. In that sense, we kept only the best iteration/model for each pair <classifier; dataset>, resulting in 140 models.

(c) As a first filter of our evaluation, we keep only the models that have an accuracy higher than 90% (models with lower accuracy would not even be considered for deployment in practice, or even for trying to improve them before deployment) on test datasets without perturbations. Results of this step are given in Table 5.1, where results in red are related to the models we used in the following (53 models). In this table, two accuracy values are presented for each dataset. This represents (i) the accuracy results obtained in the literature (column "Ref")—especially in [FFW+19]—and (ii) our own accuracy results (column "Our"). As explained previously, the hardware (as well as the software) can impact such results, so comparing results enables only keeping models that have equal or greater accuracy than the ones in the literature. Note that, as the Tlenet classifier does not offer satisfying models for any datasets (i.e., with accuracy $> 90\%$), it will not be studied any deeper. Similarly, no classifier gave suitable results on the datasets "DodgerLoopDay", "DodgerLoopGame", "Earthquakes", "FordB", "InsectWingbeatSound", "Lightning2" and "Lightning7". These datasets will, therefore, be discarded from further analysis.

(d) Generate new datasets containing perturbations as defined in the previous section. In this evaluation, we generated a total of 13,250 datasets (5 different values for Pe—from 1% to 20%, in steps of 5%—using swapping and dropping perturbations and 9 different values for R using swapping perturbations from 1 to 10 positions in steps of 1, all over 5 iterations to take into account the randomness of the perturbations functions). Note that only uniform distribution has been used in the random processes, and a linear regression between the previous and next values (i.e., the average value: $e_j = \frac{e_{j+1}-e_{j-1}}{2}$, where j is the index of the dropped/reconstructed value) is used for filling out the dropped value. Note also that this linear regression used to reconstruct a dropped value is convenient for the implementation of the DL algorithms since it requires the same length for all the TS (in addition to following the mathematical properties of a TS, i.e., to be equally distributed). We therefore believe that practitioners need to be aware of such constraints when implementing AI models.

2. The *empirical study* consists of:

(a) Evaluating the robustness of each model (on each dataset generated);

(b) Concluding about the impact of such perturbations on the algorithms/models.

To run these experiments, we implemented the DL models by using Keras 2 (`https://keras.io`, accessed on July 2021) framework with TensorFlow backend (Python 3.6) and the WEASEL algorithm developed in Java. All the models were trained on the University of Luxembourg High-Performance Computer (HPC) with 1 Graphics Processing Unit (GPU) (NVIDIA TESLA V100) on Compute Unified Device Architecture (CUDA).

Table 4.2: Accuracy of the best models for each dataset (Accuracy in %)

Dataset	**mlp**		**resnet**		**tlenet**		**mcdcnn**		**cnn**		**fcn**		**WEASEL**
	Our	Ref	Our	Ref	Our	Ref	Our	Ref	Our	Ref	Our	Ref	
Car	77	80	**93**	**93**	32	32	75	80	78	80	**93**	**93**	82
DodgerLoopDay	54	16	54	15	16	16	54	53	59	58	40	15	53
DodgerLoopGame	86	88	86	80	52	48	88	90	83	83	78	78	80
DodgerLoopWeekend	**99**	**98**	**96**	**96**	74	74	**99**	**99**	**98**	**98**	**91**	**93**	**97**
Earthquakes	76	73	75	73	75	75	75	75	72	72	74	73	74
FordA	85	82	**94**	**95**	52	52	89	89	90	90	**92**	**92**	**97**
FordB	72	71	82	82	50	50	70	73	77	77	78	78	83
FreezerRegularTrain	82	**91**	**100**	**100**	50	50	**98**	**98**	**99**	**99**	**100**	**100**	**98**
FreezerSmallTrain	69	69	**96**	**93**	50	50	70	74	74	75	71	71	**91**
InsectWingbeatSound	66	61	51	50	9	9	61	58	59	59	40	40	63
ItalyPowerDemand	**96**	**96**	**96**	**96**	50	50	**97**	**97**	**96**	**96**	**96**	**96**	**96**
Lightning2	77	70	80	80	54	54	72	69	67	66	77	75	61
Lightning7	67	64	85	85	26	26	62	64	70	66	82	84	70
MoteStrain	87	86	**94**	**93**	54	54	85	86	89	**90**	**94**	**94**	**95**
Plane	**96**	**98**	**100**	**100**	14	14	**98**	**98**	**98**	**97**	**100**	**100**	**100**
SonyAIBORobotSurface1	73	70	**97**	**97**	43	43	79	**90**	71	72	**97**	**97**	85
SonyAIBORobotSurface2	83	83	**98**	**98**	62	62	84	86	84	84	**98**	**99**	**95**
StarLightCurves	85	**95**	**98**	**98**	58	58	**95**	**95**	**93**	**93**	**97**	**97**	**98**
Trace	61	81	**100**	**100**	24	24	86	**95**	**96**	**96**	**100**	**100**	**100**
Wafer	**100**	**100**	**100**	**100**	89	89	**99**	**100**	**96**	**96**	**100**	**100**	**100**
Our case-study (Sec. 4.6)	*95*		*100*		*na*		*99*		*99*		*100*		*100*

4.5.2 Results of the Evaluation

Let us now look at the results of the *empirical study* with regard to the perturbations. Tables 4.3 and 4.4 present an overview of our results. For the sake of our analysis, we consider here that a model is impacted if its accuracy decreases more than 1% over the experiments (with regards to its accuracy without any perturbations). Results show that:

- Very few models are not impacted at all by the swapping perturbations. To understand to what extent the models are impacted by such perturbations over the considered sensor-based datasets, we compute an average robustness, as shown in Figure 4.1. It appears that:
  - *MLP, MCDCNN, CNN*: Even if the results of the MLP, MCDCNN and CNN models are limited to, respectively 4, 5 and 7 (out of 13 possible models/ datasets), the impact of these perturbations on the models tends to be quite limited since in the worst case (i.e., with a percentage of 20% and a swap range of [1–10]), the loss of accuracy is, respectively, around 3%, 3.5% and 5% on average. Of course, in practice, the tolerance of such degradation would require analysis for each given use case.
  - *ResNet, FCN*: Contrary to MLP, MCDCNN and CNN, ResNet and FCN have been evaluated on all possible models of our study. This analysis shows clearly that such algorithms/models are more rapidly impacted by the swapping perturbations since the loss of accuracy is already about 10% on average for a low percentage of swap values (5%) and a small range ([1–3/4]) to reach 30% in the worst case scenario of our study.
  - *WEASEL*: Although WEASEL is clearly impacted by the swapping perturbations, the impact is more limited (compared to ResNet and FCN) for low percentage and range since the loss of accuracy tends to be less than 6% before being really degrading when the percentage is important (more than 15%) and/or range is high (more than [1–4]).
- Very few models are impacted by the dropping perturbations. This shows how the mitigation mechanism (as simple as it is) plays an important role in the models' robustness. It is therefore really important that practitioners understand this point and integrate it from the design phase of such AI usage. Note that, even if we consider ResNet impacted on the Car dataset, the accuracy decreases only by 5% in the worst iteration (over the 5) when the perturbations are at the maximum (i.e., $P_e = 20\%$). Similar behaviour was found for FCN on this dataset (but from $P_e \geq 15\%$). FCN accuracy on FordA decreases by 6% and less than 2% on SonyAIBORobotSurface1 in

Table 4.3: Robustness under swapping perturbations

Dataset	mlp	resnet	mcdcnn	cnn	fcn	WEASEL
Car	na	✗	na	na	✗	na
DodgerLoopWeekend	✓	✗	✓	✓	✗	✓
FordA	na	✗	na	na	✗	✗
FreezerRegularTrain	na	✗	✗	✗	✗	✗
FreezerSmallTrain	na	✗	na	na	na	✗
ItalyPowerDemand	✗	✗	✗	✗	✗	✗
MoteStrain	na	✗	na	na	✗	✗
Plane	✗	✗	✗	✗	✗	✗
SonyAIBORobotSurface1	na	✗	na	na	✗	na
SonyAIBORobotSurface2	na	✗	na	na	✗	✗
StarLightCurves	na	✗	na	✓	✗	✗
Trace	na	✗	na	✗	✓	✗
Wafer	✓	✗	✓	✓	✗	✓

na: not considered in our study cf. XX ✓: robust (not impacted) ✗: not robust (impacted)

the worst iteration. Weasel on FordA appears to be an exception where the accuracy is decreased by 21% to 41%.

Overall, this study demonstrates that some of the models/algorithms can be impacted by the data quality, especially when it is decreasing over time. Indeed, the conditions in which the data are collected can be different from the time the models were trained, leading to an accuracy decrease. Based on those results, one may wonder whether such degradation can be predicted before deploying a model in real-life scenarios, i.e., without the need to generate as many datasets as possible for testing it under perturbations (as proposed by our systematic approach). The next section tries to answer this question.

4.6 Is Robustness Predictable?

The objective is to answer the question, *is robustness predictable?* If so, we aim to provide humans/engineers with a method—as simple to understand and to interpret as decision trees—to determine whether a model will be impacted by some of the perturbations. To do so, we assume that the characteristics of the datasets (i.e., the shape) impact the robustness of the models. Based on this assumption and our previous results, we created a dataset (consisting of 2700 rows) with the following information:

- The characteristics of the TS of each dataset presented in Table4.5, i.e.,

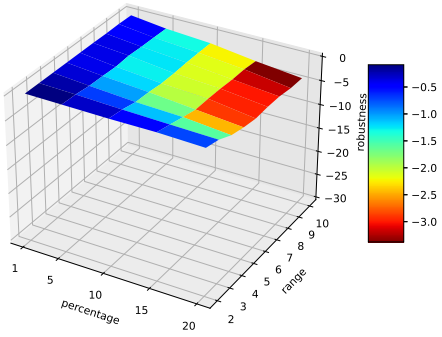
Table 4.4: Robustness under dropping perturbations

Dataset	mlp	resnet	mcdcnn	cnn	fcn	WEASEL
Car	na	✗	na	na	✗	na
DodgerLoopWeekend	✓	✓	✓	✓	✓	✓
FordA	na	✓	na	na	✗	✗
FreezerRegularTrain	na	✓	✓	✓	✓	✓
FreezerSmallTrain	na	✓	na	na	na	✓
ItalyPowerDemand	✓	✓	✓	✓	✓	✓
MoteStrain	na	✓	na	na	✓	✓
Plane	✓	✓	✓	✓	✓	✓
SonyAIBORobotSurface1	na	✓	na	na	✗	na
SonyAIBORobotSurface2	na	✓	na	na	✓	✓
StarLightCurves	na	✓	na	✓	✓	✓
Trace	na	✓	na	✓	✓	✓
Wafer	✓	✓	✓	✓	✓	✓

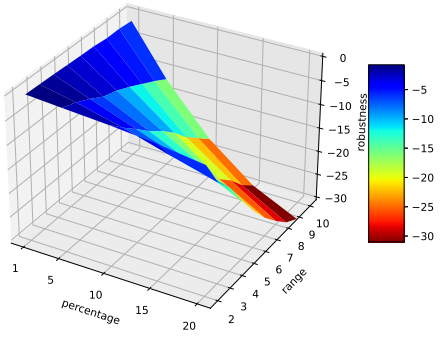
na: not considered in our study cf. XX ✓: robust (not impacted) ✗: not robust (impacted)

  - The TS length (denoted $len(TS)$),
  - The number of classes (No_{cl}),
  - The Pearson correlation coefficient [Pea96], which is defined as the covariance of two variables divided by the product of their standard deviations. It is an intuitive and easy to understand a way of measuring the linear correlation between two signals (here, two time-series), which has been used in many studies from different fields to characterise the correlation between TS [AP10, BCH08]. In this study, we used the average (i) of the Pearson correlation coefficients computed between all time series of the same class (P_{in-cl}) and (ii) of the Pearson correlation coefficients computed between the different classes for all the datasets (P_{bet-cl}).
  - The average derivative ($Deriv$), enabling to reflect the changes/variations in a time series.
- The parameters of the considered perturbations as defined in Section 5.4, i.e., P_e, the percentage of swapped/dropped values, and R, the range in which the value is swapped,
- Finally, a label (per classifier) representing if the model is impacted (or not) by such perturbations/characteristics settings.

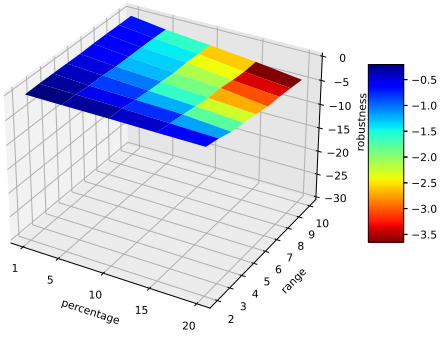
This dataset is used to create as many decision trees as classifiers. To do so, we used the sklearn decision trees library (`https://scikit-learn.org/stable/modules/tree.html`, accessed on June 2021). Although one of the major features



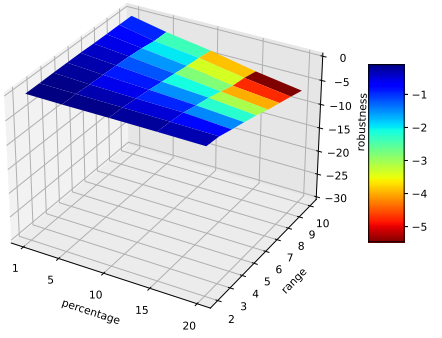
(**a**) MLP



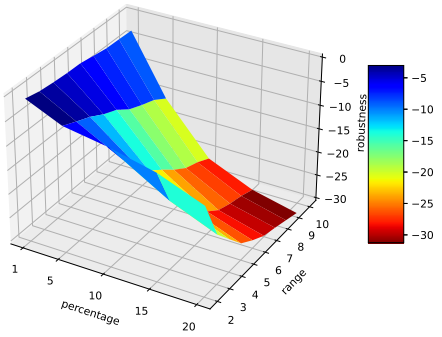
(**b**) ResNet



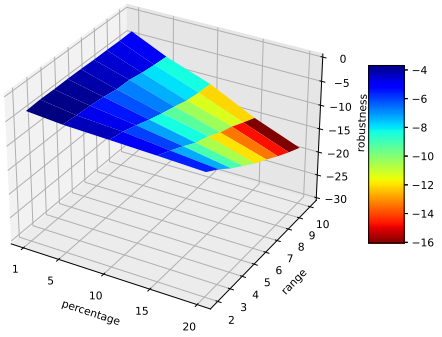
(**c**) MCDCNN



(**d**) CNN



(**e**) FCN



(**f**)WEASEL

Figure 4.1: Average robustness over all the considered datasets under swapping perturbations.

Table 4.5: Datasets' characteristics for decision trees.

Dataset	$len(TS)$	No_{cl}	P_{in-cl}	P_{bet-cl}	$Deriv$
Car	577	4	0.862	0.826	0.99
DodgerLoopWeekend	288	2	0.72	0.585	0.569
FordA	500	2	0.002	−0.002	0.115
FreezerRegularTrain	301	2	0.799	0.715	0.629
FreezerSmallTrain	301	2	0.799	0.715	0.629
ItalyPowerDemand	24	2	0.829	0.734	0.236
MoteStrain	84	2	0.547	0.421	0.238
Plane	144	7	0.95	0.648	0.131
SonyAIBORobotSurface1	70	2	0.743	0.672	0.153
SonyAIBORobotSurface2	65	2	0.402	0.231	0.206
Trace	275	4	0.743	0.064	0.575
Wafer	152	2	0.123	0.044	0.386
Our case-study (Section 4.6)	*305*	*3*	*0.7180*	*0.5470*	*0.0459*

$len(TS)$: the time-series length, No_{cl}: the number of classes, P_{in-cl}: the Pearson correlation coefficients between all time-series of the same class, P_{bet-cl}: the Pearson correlation coefficients between the different classes, $Deriv$: the average derivative.

of the decision trees visualisation, the size and number of our decision trees are too significant to be presented here. Note that the StarLightCurve dataset does not appear due to the computational resources that are needed to compute the different parameters since it contains too many long time series. Table 4.6 gives an overview of the number of leaves and depth. This shows an important disparity between the decision tree's features, and more particularly, the number of leaves, e.g., MLP has "only" 21 leaves while ResNet has 125 for the swapping perturbations. The depth is relatively steady, even if it is quite important with 7 to 12 levels. Contrary to the decision trees for the swapping perturbations, their number of leaves and the depth for dropping perturbations are not so important (even very low). Indeed, several classifiers count only 1 leaf and a depth of 0, showing the ability of models to correctly classify the time series after perturbations as it has also been raised in the previous section (partially due to the linear regression mitigation mechanism). Overall, such decision trees (in particular for swapping perturbations) do not generate easy-to-understand rules as we expected and do not provide clear indications of which parameter(s) impact the robustness the most (or the classification in a class "impacted"/"not impacted").

To test if such decision trees are nonetheless applicable to predict whether the model accuracy will be impacted by our perturbations, we applied a proof by contradiction (reductio ad absurdum), assuming that the characteristics of the

Table 4.6: Characteristics of decision trees for both perturbations.

Classifier	Swapping Perturb.		Dropping Perturb.	
	Leaves	Depth	Leaves	Depth
MLP	21	7	1	0
ResNet	125	12	6	5
MCDCNN	42	10	1	0
CNN	68	10	1	0
FCN	76	12	10	5
WEASEL	64	10	2	1

datasets and therefore the decision trees enable predicting the impact of the perturbations on a dataset. To put it in another way, if an example does not satisfy this assumption, then the answer to the aforementioned question will be considered as "No". To do so, we developed a case-study for collecting our own data. Thanks to our Fischertechnik factory simulation (`https://www.fischertechnik.de/en/service/elearning/simulating/fabrik-simulation-24v`, accessed on July 2021), we collected data from a light sensor that is used for classifying the parts according to their colours (blue, white or red), i.e., 3 classes for our time-series classification problem. The datasets' characteristics are described in the Table 4.5. Note that the training and testing sets consist of, respectively, 100 and 50 TS of each colour (i.e., resp., a total of 300 and 150 TS). New datasets with perturbations have, therefore, been generated as achieved with the public datasets (cf. previous section) and analysed similarly, as shown in Figure 4.2. Then, decision trees are applied to the characteristics of our original datasets to predict if the model will be impacted with a given level of perturbations. Table 4.7 gives an overview of the results. Overall, this shows that:

Table 4.7: Accuracy of decision trees for the swapping effect.

Classifier	MLP	ResNet	MCDCNN	CNN	FCN	WEASEL
Swapping perturb.	12%	30%	43%	37%	36%	89%
Dropping perturb.	100%	80%	100%	100%	100%	100%

- *Swapping perturbation*: we notice a disparity in the results between DL and ML methods.
  - *Deep-learning*: The accuracy of the decision trees on DL models is really low. This means that the decision trees here are not able to predict whether the dataset will be impacted (or not). Indeed, by training decision trees, we try to create a model (the tree) that represents the

behaviour of the DL classifier. However, deep-learning models are very complex. There are so many parameters to take into account —even the hardware resources—, which make it almost impossible to predict its behaviour beforehand. To illustrate this complexity, let us look at Figure 4.1 where *ResNet, FCN and WEASEL* were the least robust classifiers under perturbations over the 12 datasets, while *MLP, MCDCNN and CNN* seemed to be robust. This observation could have led to a first —quick/natural—conclusion, where the three latter classifiers should be the best to deploy (especially in a possibly noisy environment). However, regarding the results of our case-study, the results are the opposite: *MLP, MCDCNN and CNN* are the classifiers for which the accuracy decreases quicker under perturbations when *ResNet, FCN and WEASEL* are more robust to them. One may note that this has further opened up a research topic on explainable ML.

  - *WEASEL*: Concerning Weasel, which is more a *"traditional ML classifier"*, a decision tree is more able to predict the impact of a future perturbation. Actually, the decision tree has an accuracy of 89%, which can be satisfying for helping humans to make a decision out of it (especially when associated to his/her expertise of the environment).

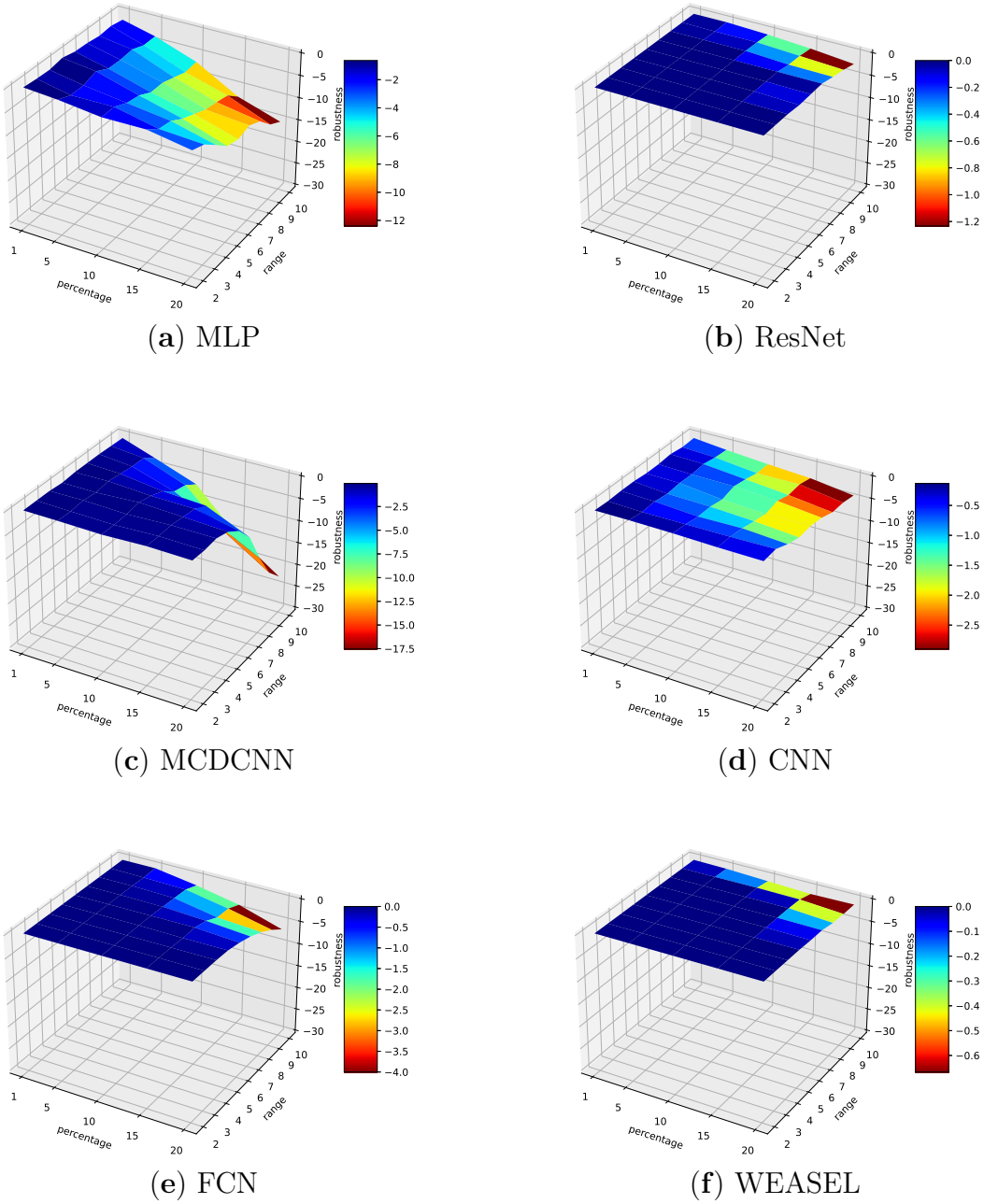
- *Dropping perturbation*: decision trees have a better accuracy in such perturbations. This is due to the small impact they have on the robustness of the models (again, thanks to linear regression mitigation mechanism), resulting in few scenarios where models are impacted, leading to an easier behaviour prediction.

In conclusion, this study shows that it is not easy to predict (based on the characteristics of a dataset) that some perturbations will impact the accuracy of a model trained on a dataset assumed to be ideal (i.e., without perturbations). As a consequence, the systematic approach, presented in Section 4.4, is really important to perform for evaluating AI models under perturbations before the deployment in an industrial environment prone to data quality degradation.

4.7 Conclusions, Implications, Limitations and Future Research

4.7.1 Conclusions

The world is engaging its digital transformation by providing industry with new tools for controlling their production and business systems. It aims at improving the efficiency of the production while covering the needs of sustainability,



(**a**) MLP (**b**) ResNet

(**c**) MCDCNN (**d**) CNN

(**e**) FCN (**f**) WEASEL

Figure 4.2: Average robustness over FischerTecknik dataset under swapping perturbations.

transparency, traceability and customisation requested by the customers. Thanks to the huge amount of sensor data available, AI is suitable for decision-making. However, in many companies, there is a lack of skilled engineers who master both AI technologies and the business specificities of the company. Thus, a condition for broad adoption is that engineering a performant and robust AI-based system must remain simple while leading to performances at least equal to existing solutions. In that sense, industries are not ready to implement such technology without being convinced that it will work smoothly and properly. That is why it is important to also evaluate the performance of AI models under perturbations (that could happen in the industrial environment). This paper shows that it is costly and hardly predictable, since predicting whether an AI model will be impacted is not straightforward (or not accurate enough). This shows the necessity to generate different perturbations (as presented in this paper) to evaluate the robustness of the AI models.

4.7.2 Implications

This research presents two main implications. First, it points out that ML/DL researchers should not stop their model evaluation once they have the accuracy computed on clean datasets and without any biases. The robustness of their model should always be assessed by presenting the perturbations formally. Second, it shows practitioners that simple measures such as the linear regression used for handling potential losses (dropping perturbations) can prevent AI algorithms from degrading in noisy environments.

4.7.3 Limitations and Future Research

Some limitations of our research can be pointed out. First, the approach and the empirical study are only based on two kinds of perturbations (swapping and dropping perturbations). Future researches will go further in that direction by proposing analyses of other perturbations (which we encourage when implementing/evaluating such an AI system in/for such industrial environment). In addition, choices have been made concerning the range of perturbations' parameters. For instance, ranges and percentages of perturbations could be widened, or a finer-grained analysis could be conducted. Furthermore, our perturbations follow a uniform distribution, which can be adapted for other use-cases. Finally, the proposed methodology for predicting the robustness of the AI models, relying on decision trees, is based on 'only' the experiments of thirteen datasets and seven algorithms for each kind of perturbation to train the trees. This might be too few, and input

training data could give different results with more inputs.

5

Towards Models Robustification in Industrial Settings

Foreword

This chapter presents an extension of the previous systematic approach to the Machine Learning models' robustness in industrial settings.
This work is not yet published, but we aim to submit it in the Elsevier Journal of Manufacturing Systems in the future.

Contents

5.1 Abstract

Machine Learning tends to be widely used for solving data analysis problems in our society. It can reach great performance for various data mining problems, such as time series classification. While industries are shifting from the third to the fourth industrial revolution, the volume of data becomes impossible to be analysed by humans. To overcome this issue, industrial actors aims at relying on Machine Learning solutions to monitor and make decisions based on this large volume of sensors-based data collected on their production tools. However, companies have to be ensured that AI-based solutions are able to perform over time, with the evolution of the production systems, the applications and the perturbations that are likely to happen during their lifetime. The objective of this study is to provide a fine-grained systematic approach, based on Genetic Algorithm, to inject artificially perturbations in the data sets and to evaluate the robustness of Machine Learning and Deep Learning models. In this evaluation, we extend our previous coarse-grained empirical study, and compare the results of both approaches on 53 models under swapping effect perturbation, regarding state-of-the-art classifiers and data sets. The results of this fine-grained evaluation aim at being used for further robustification process of the models. The results of this fine-grained robustness evaluation show that 49 out of the 53 models are impacted by the perturbations, and in most of the cases, they are more impacted than using the previous random perturbation generation approach from chapter IV.

5.2 Introduction

The recent progress in the fields of ML, automation and process management have motivated industrial actors upgrading their productions systems, shifting from the third to the fourth industrial revolution, the so-called Industry 4.0. Indeed, in the past decades, a large volume of data is becoming available from the production tools of manufacturers, due to the wide use of distributed control systems based on open communications technologies that allows to collect data in a unified manner as we already presented in the first part of the dissertation. While it becomes more and more difficult to build first-principle models in those increasingly complex processes, data-driven process modeling, monitoring, prediction and control have received much attention in recent years [GSDH17]. To overcome this growing complexity in data-analysis, researchers have paid attention to data-mining techniques for industrial purposes [MC12, GSG13, Qin12]. Data-driven models are based on the observations of the process (e.g., by the different sensors) and are very attractive modelling approaches that enhance the

decision support methods and the monitoring of the production [APBRPOM15]. In this category of models, we can find ML classifiers for which industrial actors tend to rely on for analysing their production data (observations). However, data coming from the production tools (e.g. sensors on mechanical machines) are prone to data-drift during the life-cycle of the production plant [KGS09]. We have demonstrated in Part I that data collection infrastructures may have sparse performance, in particular in terms of network metrics as losses, delays or traffic-load [BRLT19, BRLTK18] resulting in the data quality degradation. In order to fully trust intelligent data-analysis tools (e.g., Machine-Learning models), we (SnT and our partner Cebi Luxembourg S.A.) are willing to explore the limits of a given model, in order to ensure that this model can still perform in degraded conditions. We have therefore conducted a first coarse-grained study, presented in chapter IV ([BRLT21]), in order to evaluate the robustness of several ML models under perturbations, concerning TSC task in industrial settings. We have demonstrated that the robustness of the models under perturbations is contrasted, depending on the classifier and the data set under evaluation. Some of the models are really robust to perturbations, needing to make consequent changes in the data set in order to notice a decrease of the accuracy. On the contrary, for a subset of the models under test, the accuracy drops quickly with fewer changes in the data sets. In addition, we also noticed that for some of the models, even with a large percentage of perturbations, the accuracy does not drop, illustrating a great robustness of such models to a kind of perturbation. However, this study has been conducted using random perturbation generations using a straightforward and easy-to-apply algorithm to modify the time series composing the data sets. Even if it helps be aware of the potential misclassification of such models under perturbations, we want to refine this study by exploring limits and critical cases of each of the models. Those limit cases could be subject to be used for further robustification of the different models (independently). To do so, in this chapter, we want to extend the previously developed systematic robustness evaluation approach in a optimised manner. The goal of this optimisation of perturbations generation is to find the weaknesses of a model, meaning that the objectives are to decrease the accuracy while minimising the perturbations. This is done by shifting from a random method to a optimised one that aims at minimise multiple objectives. It intents at being aware if such an optimised method can perturb the models that were previously robust to (random) perturbations, but also to know whether for previously not robust models, we could decrease their accuracy with less changes in the time series, by optimising the perturbations on the weaknesses of the models. To do so, this chapter proposes a fine-grained study by artificially injecting perturbations in data sets. The method used to craft the perturbations is based on Genetic Algorithm (GA), that have shown great performance for optimi-

sation problem [EAAV20]. This optimised approach evaluated on the previously selected models and data sets (from chapter IV) in a matter of fair comparison. The UTS produced by the optimisation methods aims at being further used for the robustification of the models. This robustification process is not studied in this dissertation. The objective of the chapter is therefore twofold: i) to evaluate the impact of swapping perturbation generated using genetic algorithm method on 6 state-of-the-art algorithms and 13 sensor-based benchmark data sets and ii) to compare the results of the **optimised** approach using genetic algorithm with the **random** perturbation generation used in the previous chapter. The chapter is organised as follows: Section 5.3 presents a optimised systematic approach for evaluating AI models under perturbations. Then, in section 5.4, the systematic approach is therefore assessed by experiments on swapping perturbation, generated using Genetic Algorithm optimisation method. Finally, section 5.5 presents the conclusion of the chapter.

5.3 An optimised systematic approach to evaluate models' robustness

In this section, we present the systematic approach we propose to evaluate the robustness of the models. This approach is based on our previous study presented in Chapter IV, yet some of the stages have to be reshaped, as depicted in Figure 5.1.

Let us present the different steps of the methodology hereafter:

- **Training phase:** The first step of the approach consists in training models on the desired data sets (Tr_x on Figure 5.1, with $x \in X$, X a set of data sets). This training phase has to be done with the assumption that the data sets do not contain perturbations. The outputs of this first phase is the different trained models, namely m_x in Figure 5.1. In this study, this phase has been skipped, since we want to compare the optimised generation of perturbations with the random one. To do so, we reused the previously trained models from Chapter IV. However, in case of applying this methodology from scratch, user should care about the behaviour of the classifiers under test. Indeed, using DL methods can lead to different accuracy results through different iterations. In our previous study [BRLT21], presented in Chapter IV, we performed 5 training iterations for each couple "classifier/data set" and retained only the best out of the 5 iterations. This training phase had been done on 20 UCR UTS data sets of the Sensors category and 7 classifiers, namely MLP, Resnet, Tlenet, MCDCNN, CNN, FCN and Weasel.
- **Testing phase:** The second step of the approach is the testing phase of the models. This phases consists in testing the models on the data sets that

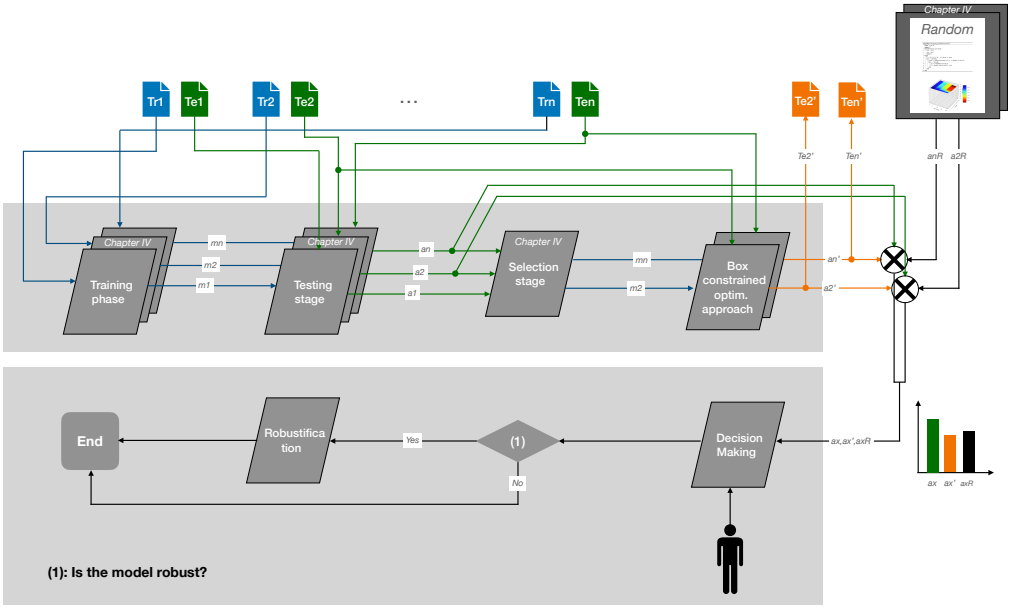


Figure 5.1: System approach for optimised models' robustness evaluation

do not contain perturbation (Te_x in Figure 5.1). This accuracy without perturbation will be used for two purposes: 1) It will be the input for the Selection Stage of the methodology, and 2) It will be an input to compare the accuracy of the models after testing them with perturbed data sets, in order to evaluate their robustness to perturbations. In this study, the testing phase (without perturbations) has also been skipped since we use the models from our precedent study presented in Chapter IV. In case of using UCR UTS data sets, the testing sets are provided independently from the training sets. However, in these cases, the testing phase is done on a sub-part of the training set.

- **Selection Stage:** The third step of our approach consists in selecting the models for which the robustness to perturbations will be evaluated. Actually, as mentioned in Chapter IV, from a practical viewpoint, a model that does not reach a certain threshold in terms of accuracy (before any perturbations injection) will not be retained for a future deployment, since it would require many engineering phases and human resources to improve its performance in normal conditions. Outputs of this stage are the previously trained models m_x from the Training phase that reached the predefined accuracy threshold. In our case, we decided to discard the models that do not reach an accuracy of 90% in nominal conditions, resulting in 53 models distributed between 6 classifiers and 13 data sets. The output of this phase is presented in Table 5.1.

- **Box Constrained Optimisation Approach:** This phase is the core of the systematic approach. It consists in selecting, designing and tuning the optimisation method that will be used to alter the data , and then to evaluate the robustness of the models in these degraded data quality conditions. In this chapter, we want to evaluate the robustness of the models under *swapping effect* perturbations. We already did it in the previous chapter using random algorithm to generate perturbations. In this study, we want to optimise the perturbations generation in order to perturb the models with less changes in the time series, and to compare the results with the random ones. As a reminder, the *swapping effect* consists in swapping a pair of data points in a TS[1]. We therefore define two parameters that define the swapping effect perturbation:

  1. Pe as the percentage ($0\% < Pe < 100\%$) of swapped events/values in each TS of a data set. It means that $S = N * \frac{Pe}{100}$ values will be swapped in a TS of length N.
  2. R as the range in which the value is swapped (i.e., at which position the data are moved in a certain range of possibilities).

  The optimisation problem is then multi-objectives, with three objectives to minimise, as formalized hereafter:

  1. The first objective $g_1(E')$ is related to the accuracy of the model under study. A classifier H outputs $h(E')$, the probability of the TS E' to belong to the right class. We want to minimise this objective so that the classifier predicts a higher probability for E' to belong to another (false) class. That is equivalent to minimise $h(E')$, denoted as follows:

  $$minimise(g_1(E')) \equiv h(E')$$

  2. The second objective is related to the perturbations generation. As mentioned above, we want to swap values of the TS in order to minimise the accuracy of the models. However, we want to minimise the percentage Pe of swapped values into the TS. It can be then expressed as the minimisation of a distance between an original time series E and a altered one E', with regards to the number of values that have been swapped:

  $$minimise(g_2(E')) \equiv DistSwap_1(E, E')$$

  $DistSwap_1$ represents the number of values that have been swapped

[1] A time series TS is an ensemble E representing a sequence of N data-points e_n, assumed as equally distributed: $E = [e_1, \cdots, e_i, \cdots, e_j, \cdots, e_N]$.

between a TS E and E' and is formalised as follows:

$$DistSwap_1(E, E') = \frac{|\{i : e_i \neq e'_i\}|}{|E|} \quad \text{s.t. } (e_i, e'_i) \in (E \times E') \quad (5.1)$$

With i representing the indexes of the values in the TS.

3. The third objective is also related to the perturbations generation. As mentioned above, we want to swap the values of the TS between a certain range R. In a matter of optimisation, we want to minimise this range in-which values are interchanged. It can be thus expressed as the minimisation of a distance between a TS E and E':

$$minimise(g_3(E')) \equiv DistSwap_2(E, E')$$

$DistSwap_2$ represents the average (avg) range R of the values that have been swapped between a TS E and E', and is formalised as follows:

$$DistSwap_2(E, E') = avg(\{|i - j|;\ \forall (i, j) \in [\![0, N]\!];\ e_i = e'_j\}) \quad \text{s.t. } (e_i, e'_i) \in (E \times E') \quad (5.2)$$

The three-objective function to minimize can then be expressed as follows:

$$\begin{aligned} minimise(g_1(E')) &\equiv h(E') \\ minimise(g_2(E')) &\equiv DistSwap_1(E, E') \\ minimise(g_3(E')) &\equiv DistSwap_2(E, E') \end{aligned}$$

To solve this multi-objective problem, we have got inspired by the MoEvA2 framework [SDG+21, GCG+20], which operates in a black-box way. This framework is based on a genetic algorithm and aims to craft adversarial examples to decrease the accuracy of ML models, by minimising a multi-objective fitness function. In their framework, they also implemented a domain-specific constraints aspect that, at this stage of this study, is not implemented for our case-study. Algorithm 4 formalizes our optimised perturbations generation approach. Note that the resulting data sets (named Te'_x in Figure 5.1), containing swapping effect perturbations, are stored for two purposes 1) To test the models for evaluating their robustness under perturbation, and 2) To use them in the robustification phase, in accordance with the decision-making process outcome.

- **Evaluation:** The evaluation phase consists in the comparison of the models accuracy before injection of perturbations (which is in our case the nominal accuracy, denoted a_x in Figure 5.1), and after injection of perturbations (denoted as a'_x). In addition, as we want to also compare the impact of

an optimised approach to generate perturbations with a random approach (evaluated in Chapter IV), the evaluation is done regarding three different accuracies for each model: the nominal accuracy a_x, the randomly-perturbed accuracy a_{xR} and the optimised-perturbed accuracy a'_x. These three accuracies (for each model) serve as inputs for the decision-making stage. Note that, in a matter of comparison with the random perturbations generation, we evaluate this optimised approach for a percentage Pe of swapped values between 0% and 20%, 0% representing the accuracy of the model without perturbation, and for a range R between 1 and 10 positions. Thus, during testing phase of the models under perturbations, we test the models only on the solutions (created with our genetic algorithm) respecting these two parameters boundaries.

- **Decision-Making:** This phase is related to the decisions to make with regards to the results of the robustness evaluation of the models. Indeed, according the the accuracy of the models under a certain pertubation, different counter-measures can be applied in order to make the system robust to them and avoiding misclassification. The decision belongs to the end-user of this approach depending on the context of its study (wrt. the perturbations that have been studied). For instance, if a model is robust to perturbations (its accuracy does not decrease when testing it in degraded data-quality conditions), the evaluation process can terminate and the model can be deployed. However, if a model shows signs of non-robustness to perturbations (its accuracy drops of a certain percentage after perturbations injection; in our case more than 1% compared to the nominal accuracy), counter-measures should be taken in order to either avoid the perturbations or to make the model robust to them. These counter-measures are applied in the robustification phase.
- **Robustification:** The robustification phase consists in applying the decision(s) that have been made during the previous decision-making stage, in case of models non-robustness. It aims at either make the models more robust to perturbations, or make changes in the data-collection architecture, by applying the different counter-measures that have been decided during the decision-making stage. Indeed, based on the perturbations that have been studied, the end-user could either have chosen to re-design its data-collection architecture (e.g. protocols, gateways, hardware, or even production tools instruments), or to use models robustification's techniques such as data augmentation for the retraining of the models, with the perturbed data generated during the box-constrained-optimisation phase. At the end of this phase, two choices can be made:

  - one is to conduct an other robustness evaluation on the model, in or-

der to know whether the counter-measures that have been taken have reached the expectations or not in terms of model's accuracy;
  - second is to stop the robustness evaluation, resuming this systematic evaluation process.

At this stage of the dissertation, the robustification phase has not been developed and belongs to the perspectives of the thesis.

- **End:** The End of this systematic approach is reached is two cases: 1) When a model is robust, meaning that there is no more need of robustification process before deploying it on a real environment, or 2) When a model is not robust to perturbations, but techniques to avoid them and/or robustifying the model are too costly (in terms of resources such as time, human or money). In these cases, end-users decide to stop the strategy to evaluate and/or robustify the model, and then make the decision to deploy it (or not) in their production plant data analysis tool.

Table 5.1: Accuracy of the best models for each dataset (Accuracy in %)

Dataset	**mlp**	**resnet**	**mcdcnn**	**cnn**	**fcn**	**WEASEL**
Car	na	93	na	na	93	na
DodgerLoopWeekend	99	96	99	98	91	97
FordA	na	94	na	na	92	97
FreezerRegularTrain	na	100	98	99	100	98
FreezerSmallTrain	na	96	na	na	na	91
ItalyPowerDemand	96	96	97	96	96	96
MoteStrain	na	94	na	na	94	95
Plane	96	100	98	98	100	100
SonyAIBORobotSurface1	na	97	na	na	97	na
SonyAIBORobotSurface2	na	98	na	na	98	95
StarLightCurves	na	98	na	93	97	98
Trace	na	100	na	96	100	100
Wafer	100	100	99	96	100	100

na: not considered in our study

Algorithm 4 presents the perturbations generation process of our study. We detail the different stages hereafter:

- **Initial population:** Here, an *individual* represents a particular time series E and a *gene* represents a particular index of the time series. Actually, in order to create solutions from values that belongs to the original times E and not creating new values in the time series, we process our algorithm on the indexes the time series, and then we reconstruct the time series by mapping the new indexes with their corresponding values. In addition, if two values

Algorithm 4: Swapping Effect using GA

Input : E, an initial state;
$fitness = [g_1(E'), g_2(E'), g_3(E')]$, a list of objectives to minimise;
N_{gen}, a number of generations;
L, a population size;

Output : P, a population minimising the objectives

```
1  begin
2      P ← init(E, L);
3      for i ← 1 to N_gen do
4          P_parents ← random(P)
5          P_offsprings ← Crossover(P_parents)
6          P_offsprings ← Mutation(P_offsprings)
7          P ← Survive(P_parents ∪ P_offsprings, fitness)
8      end
9  end
10 Return P
```

e_i and e_j are equal and are swapped, when computing the distance the result would be 0, leading to miss the swap that has been done. By working on the indexes, we avoid this case. For instance, a time series E composed of N values e_i becomes a time series E composed of N indexes i. Then our algorithm processes on this list of indexes. At the end of the evolutionary process, we replace the list of (swapped) indexes i by their values e_i. The algorithm first initializes a population P of L solutions. from the original test data set (Te_x on Figure 5.1) . In this case the initial population P comprises L copies of E, with $L = 203$. This choice is made for technical reason described later with the survival process, we use 203 instead of 200 elements.

- **Population Evolution:** The algorithm proceeds iteratively and makes the population evolve into new "generations", until it reaches a predefined number N_{gen} of generations. At each generation, the algorithm evaluates each individual in the current population on the three-objectives to minimise (e.g. $g_1(E'), g_2(E'), g_3(E')$) defined above. The evolution of the population is composed of four steps detailed as follows:

  1. Random Selection: A sampling process defines the initial set of solutions which are the starting point of the optimization algorithm. Our algorithm produces, through successive random selection of P population, a population $P_{parents}$ comprising 50 pairs of parents, randomly

selected from P.

2. Crossover: An order crossover [Liu10] is performed on $P_{parents}$ to create a new population of 50 pairs of offsprings $P_{offsprings}$. The order crossover is a permutation operator in which a strip of consecutive genes (here an index of a TS) from one parent drops down to the child, and the remaining values are placed in the child in the order as they appear in the second parent [WLW+16].

3. Mutation: Then, our algorithm operates permutation inversion mutation on $P_{offsrpings}$. It randomly selects a segment of a chromosome and reverse its order [WLW+16]. The set of children that results from this mutation process is then added to the current population.

4. Survival Selection: At this stage, our algorithm unifies initial population $P_{parents}$ composed of 203 individuals and $P_{offsprings}$ composed of 100 individuals (50 pairs). It then determines which individuals should be kept in the next generation by evaluating them with regards to our three objectives functions, and results in a new population P with 203 individuals. Acutally, using R-NSGA-III[2] algorithm requires to use reference directions in the survival selection. Thus 203 is the first number following 200 (which is the size of $P_{offsprings} \times 2$, such that the population is at most renewed of 50% at each generation) that equally divides a 3D space using Riesz energy[3] (due to our three-objectives function).

After the specified number of generations N_{gen}, set to 100 in this study, the algorithm returns the last population P of size L. This population is then used in the evaluation phase to test the robustness of the algorithm. Note that this algorithm is ran for each time series E of the data set, then resulting in a data set Te'_x, of size $L \times D$, with D representing the number of TS in the original data set Te_x.

5.4 Results of the Evaluation

Let us now look at the results of the empirical study, with regards to the robustness of models under swapping effect perturbations, generated using the genetic algorithm 4. For the sake of the study, we consider that a model is not robust (or impacted) if its accuracy drops more than 1% over the experiments (with regards to its accuracy without perturbations). Note that, in chapter IV, 10

[2]https://www.pymoo.org/algorithms/moo/rnsga3.html
[3]https://www.pymoo.org/misc/reference_directions.html

models were robust to swapping effect using the random process of perturbations, as reminded in Table 5.2 - column *R*.

Table 5.2: Robustness under swapping perturbations

Dataset	**mlp**		**resnet**		**mcdcnn**		**cnn**		**fcn**		**W.**	
	R	G	R	G	R	G	R	G	R	G	R	G
Car	na	na	✗	✗	na	na	na	na	✗	✗	na	na
DodgerLoopWeekend	✓	✗	✗	✗	✓	✓	✓	✓	✗	✗	✓	✓
FordA	na	na	✗	✗	na	na	na	na	✗	✗	✗	✗
FreezerRegularTrain	na	na	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
FreezerSmallTrain	na	na	✗	✗	na	na	na	na	na	na	✗	✗
ItalyPowerDemand	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
MoteStrain	na	na	✗	✗	na	na	na	na	✗	✗	✗	✗
Plane	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
SonyAIBORobotSurface1	na	na	✗	✗	na	na	na	na	✗	✗	na	na
SonyAIBORobotSurface2	na	na	✗	✗	na	na	na	na	✗	✗	✗	✗
StarLightCurves	na	na	✗	✗	na	na	✓	✓	✗	✗	✗	✗
Trace	na	na	✗	✗	na	na	✗	✗	✓	✗	✗	✗
Wafer	✓	✗	✗	✗	✓	✗	✓	✗	✗	✗	✓	✗

na: not considered in our study; ✓: robust (not impacted) ✗: not robust (impacted); W.: Weasel; R: Random; G: Genetic Algorithm

As a first step, we want to verify that the optimised perturbations generation approach (based on GA) helps us find weaknesses of the models that have not been found with the random approach. In such scenarios, the minimal accuracy of a given model would be lower than the minimal accuracy using the random approach. Figure 5.2 gives an overview of this comparison. Each point of the graph represents the minimum accuracy of a given model under perturbation , out of the 53 retained from the selection phase (presented in Table 5.1). X-axis represents the average minimum accuracy of the models using random perturbations, Y-axis represents the minimum accuracy of the models using GA perturbations. The red dashed line ($x = y$) represents the boundary that separates area where a method outperforms the other. Overall, the optimised method (using GA) outperforms the random method for decreasing the accuracy of the models. Over the 53 models, only 5 know a higher accuracy decrease using the random method.

As a second step, let us analyse to what extent the optimised method impacts the robustness of a given model. To do so, we plot the accuracy evolution in function of our two perturbations parameters (percentage and range of swapped values). In total, there are 53 robustness comparisons (two figures per model). For the sake of readability, we only include in this chapter some particular comparisons

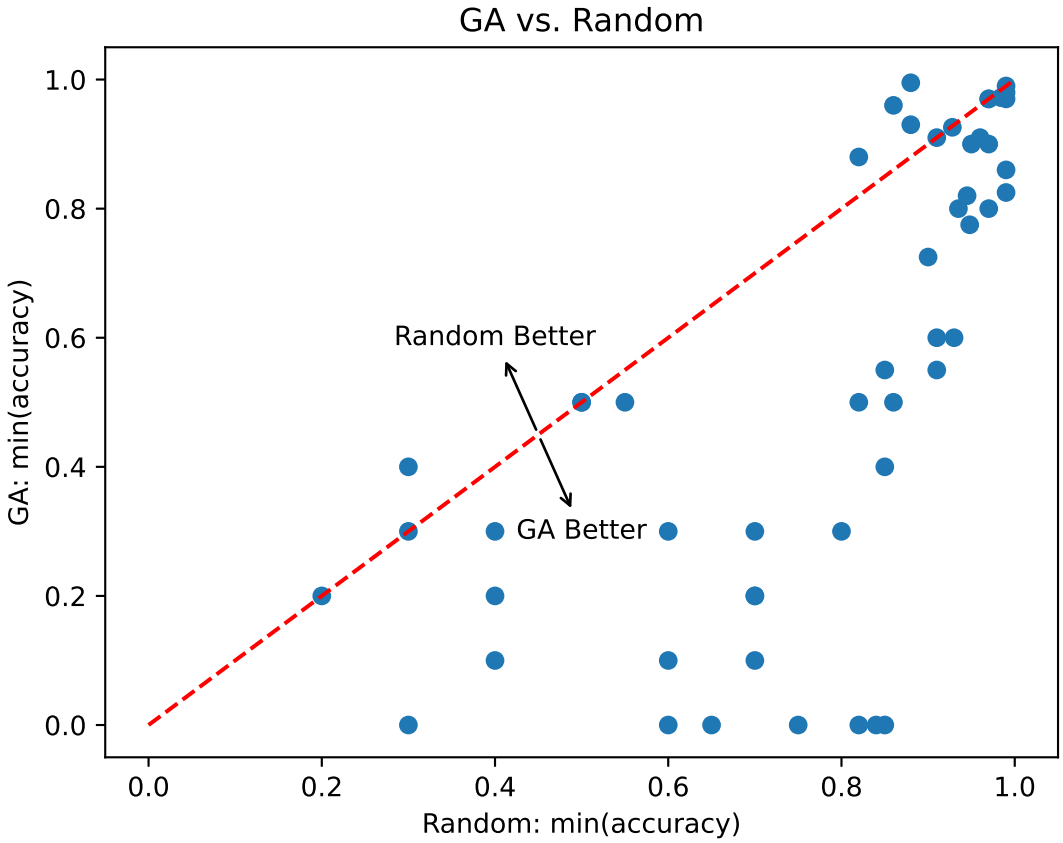


Figure 5.2: Comparison of perturbation methods results

that illustrate representative examples. For the interested reader, appendices of Chapter V presents the whole set of results.

Hereafter are presented the main observations of this analysis:

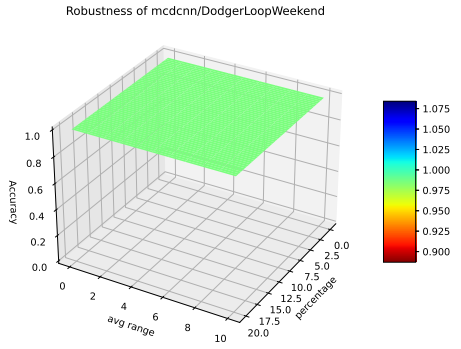
- **Robust models:** From the 10 robust models of the *random study*, there are still 4 models that are robust to perturbations generated with our GA, namely the models *{MDCNN/DodgerLoopWeekend}; {CNN/DodgerLoopWeekend}; {WEASEL/DodgerLoopWeekend}; {CNN/StarLightCurves}*, highlighted in blue in Table 5.2. For these 4 models, neither random method to generate perturbations nor optimised methods have been able to decrease the accuracy of the models for more than 1%. The models' accuracy is depicted in the following Figure 5.2. For these models, even with great perturbations (20% of values swapped in the time series, in a range of 10 indexes, the accuracy does not decrease more than 1%. It is especially the case for the data

set "DodgerLoopWeekend", for which 3 classifiers are still able to perform a correct classification after data quality degradation. In conclusion, for these 4 models that are robust to perturbations (to the extent of our study), during the decision-making phase of the approach, the end user could chose to deploy them without robustification phase. Indeed, both robustness evaluation studies (random and optimised) were not able to perturb the classification, leading to a confidence in their ability to be used in an environment prone to this swapping effect.

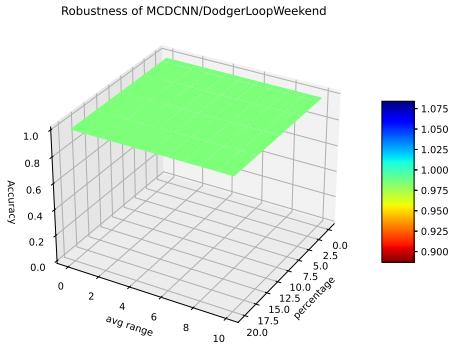
- **Robust to Not Robust models:** From the 10 robust models of the *random study*, 6 of them have been impacted by the perturbations generated with our GA approach, meaning that, for these particular models, our algorithm is better for finding the weaknesses than the random approach. This is the case for the following models: *{MLP/DodgerLoopWeekend}; {MLP/Wafer}; {WEASEL/DodgerLoopWeekend}; {Weasel/ Wafer}; {FCN/Trace}; {CNN/Wafer}*, highlighted in grey in Table 5.2. The robustness of these models is depicted in Figure 5.3. For these models, even if using an optimised approach that is able to perturb the models with swapping effect perturbations (with regards to the limits of our parameters), we can notice that the drop of accuracy is low. Actually, the models are still robust until the very high limit of percentage and range of swapped values. To conclude on these models, even if the optimised approach using GA is able to produce time series that are misclassified, in the specified limits of the perturbations parameters, the models remain very robust except in limit cases. For these models, the decision-making phase could propose two different options: 1) Consider that those limit cases may not happen, or at least so rarely such that a trade-off between cost of misclassifications and a robustification phase should be done, in order to make the decision to robustify the models or not, 2) Consider that we have to pay a particular attention to those limit cases and then process a robustification phase in order to avoid them or to be robust to them.
- **Not Robust models:** During this evaluation, we did not only focused on the robust models from chapter IV, but we also conducted the *optimised* robustness evaluation on models that were already not robust to swapping effect with the *random* approach. The results (depicted in appendices) show that all the models that were not robust to random perturbations are still not robust to optimised perturbations. It still exists 5 models for which the accuracy drops lower using the random approach, i.e. *{RESNET/Car}; {RESNET/StarLightCurves}; {WEASEL/FordA}; {WEASEL/FreezerRegularTrain}; {WEASEL/FreezerSmallTrain}; {WEASEL/StarLightCurves}*, depicted in in the upper part of the Figure 2.2. However, for these cases, the difference between optimised approach and the random approach is really

thin, meaning that the optimised approach almost reach the same performance in terms of accuracy perturbations. Even if our GA searches for the best individuals that lead to misclassification, it does not explore all the possible combinations and sometimes random approach can outperform optimisation approach. After all, for the remaining models, that have not been presented before, the accuracy's decrease is faster with the optimised perturbations generation (using GA) than with the random approach used in Chapter IV. Actually, many of the models know a faster accuracy decrease, meaning that with less perturbations the accuracy drops lower than with random perturbations generation. It can be explained by the fact that GA tries to keep most of the individuals that have the best fitness score according to our three-objectives functions, and then is more likely to find the limit cases that make the models misclassify the time series. Those limit cases, that can be interpreted as the weaknesses of the models, can be further used in the robustification phase. To conclude on these models, that are not robust to swapping effect in both evaluations, the end user should then decide (during the decision-making phase) to robustify them. This robustification could be done with a data augmentation, by re-training the models adding a subset of the data sets containing perturbations to the original training sets. Then, another robustness evaluation should be conducted on these re-trained models, in order to be aware whether their robustness have improved or not before a potential deployment, with regards to the results of the robustness evaluation.

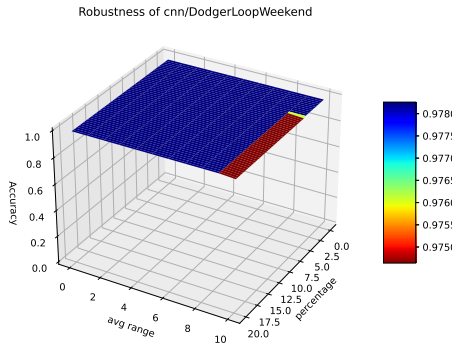
Very few models are robust to swapping effect perturbation generated using our optimised approach based on Genetic Algorithm. This fine-grained robustness evaluation shows that models can be really sensitive to perturbations that are likely to happen in industrial settings, and that even with few changes in the data, the output of classifier can be inaccurate. Indeed, using GA showed us that models can have weaknesses even if the data has not been hugely tempered. Finally, this evaluation shows that a particular attention have to be paid on the models robustness before a real-life deployment, in order to be aware of their limits, but also to make decisions to avoid or to be robust to these degraded data quality cases.



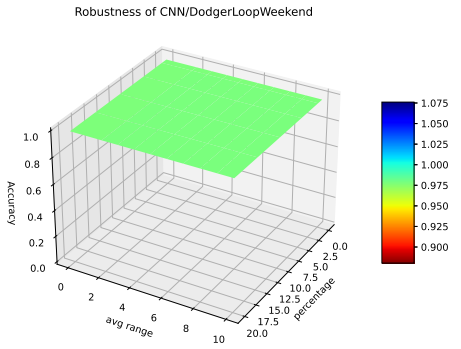
(**a**) GA



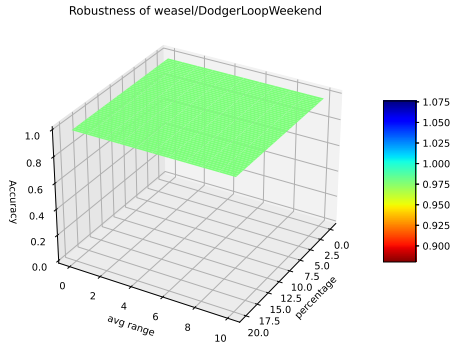
(**b**) Random



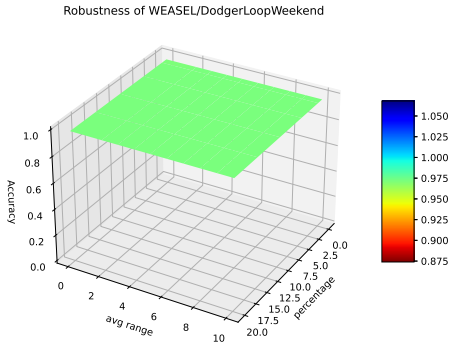
(**c**) GA



(**d**) Random

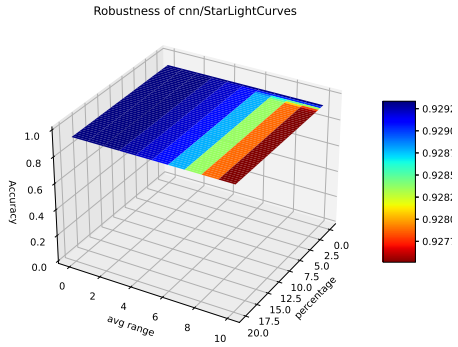


(**e**) GA

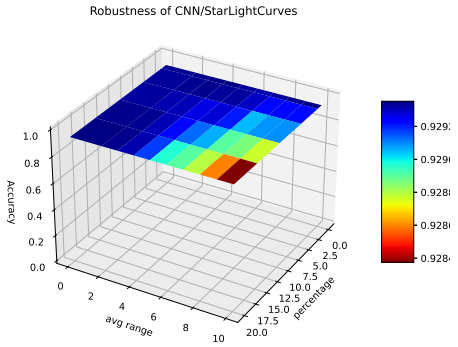


(**f**) Random

Figure 5.3: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for robust models



(**g**) GA



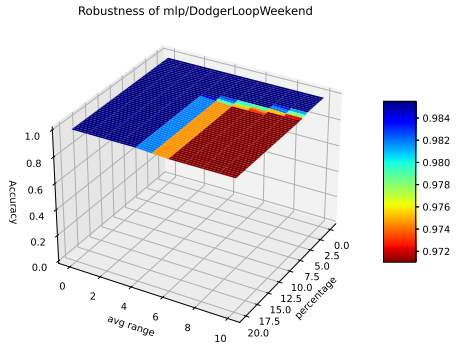
(**h**) Random

Figure 5.2: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for robust models

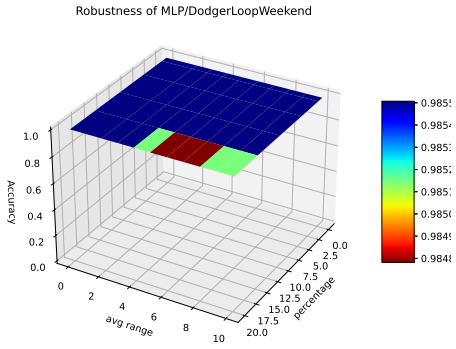
5.5 Conclusions, Limitations and Future Research

5.5.1 Conclusions

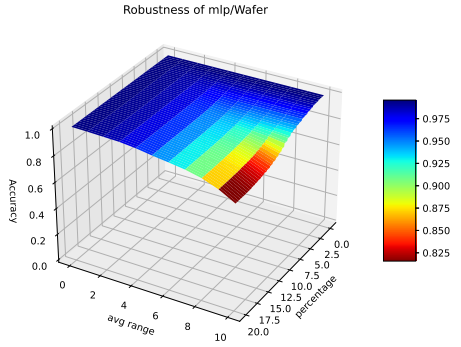
Thanks to the recent improvements that have been made in the field of Machine Learning, industrial actors are showing an interest in such intelligent approaches to analyse their production data, for monitoring and managing their productions plants. Indeed, due to the huge volume and diversity of data collected on the production tools, AI technologies tend to be at the forefront in the context on Industry 4.0. However, companies need to trust these technologies before relying on them for making decision. If studies showed that ML models can reach great performance for state-of-the-art applications, industrial actors need to be ensured that such AI-based systems can perform in case-specific degraded conditions. This chapter proposes a fine-grained robustness evaluation approach, based on injecting artificially perturbations in the data. This approach, extended from the Chapter IV, proposes to use optimisation method to generate perturbations, such that weaknesses and limits of the models can be found. The approach then proposes to use the robustness of the models for further decision-making process, aiming at deciding if a robustification of the models, before real-life deployment, as to be envisaged or not. This study aims at being systematic and reproducible for other perturbations, to the willing of the end-user, such as our partner Cebi Luxembourg S.A. We finally evaluated our approach on state-of-the-art UTS classifiers and data sets, with regards to swapping effect perturbation generated using a Genetic Algorithm. The results show that such perturbations can lead to inac-



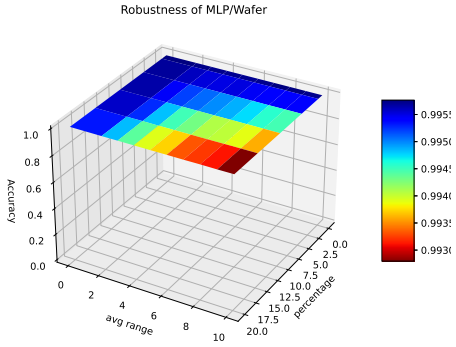
(**a**) GA



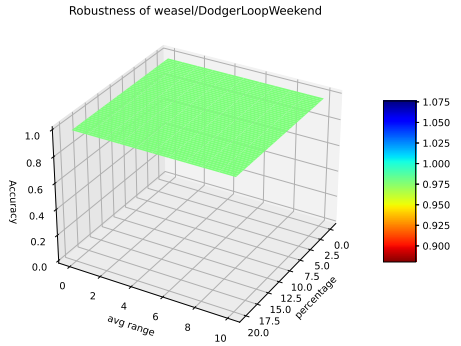
(**b**) Random



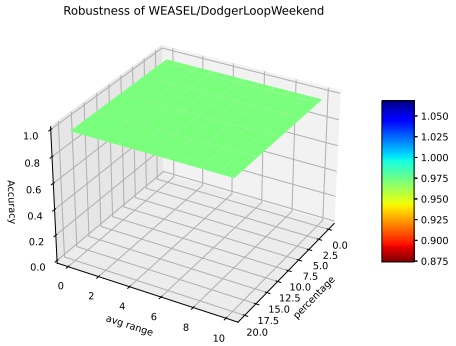
(**c**) GA



(**d**) Random

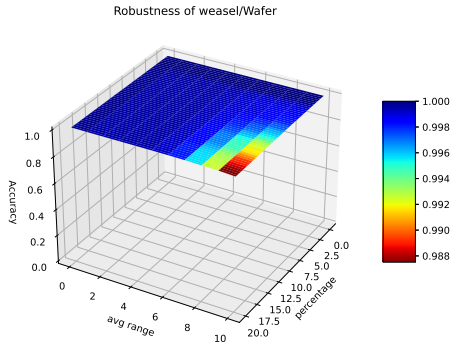


(**e**) GA

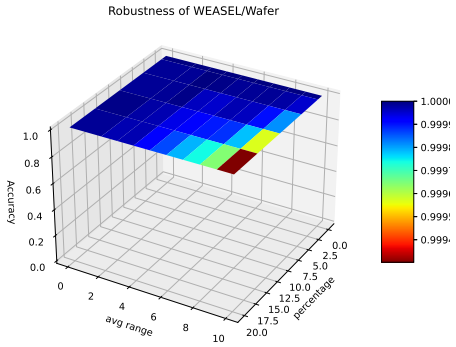


(**f**) Random

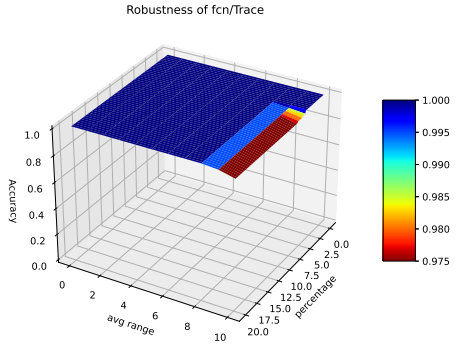
Figure 5.3: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for robust to not robust models



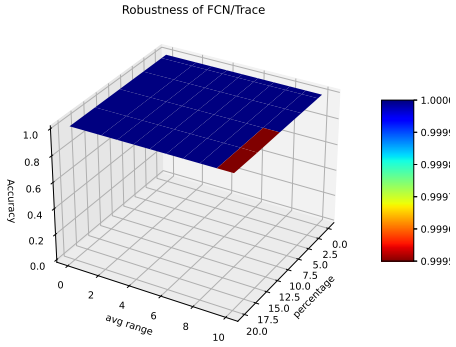
(**g**) GA



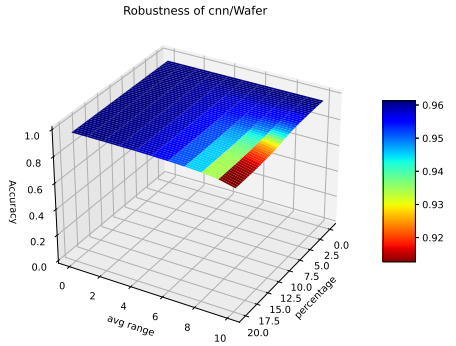
(**h**) Random



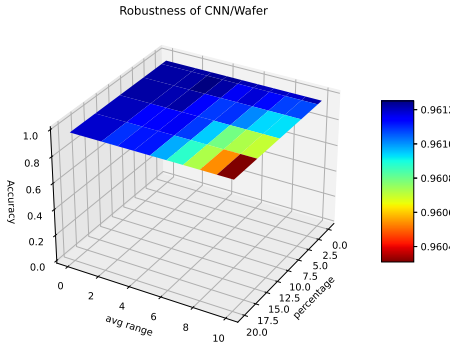
(**i**) GA



(**j**) Random



(**k**) GA



(**l**) Random

Figure 5.3: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for robust to not robust models

curate data analysis in most of the cases. We also compared the results of this fine-grained evaluation with our previous coarse-grained evaluation (based on random perturbations) generation). This comparison shows that is most of the cases, the optimised approach is more efficient to generate perturbations that will impact the classification. This shows the interest of conducting such fine-grained models robustness evaluation before real deployment, and we encourage to replicate it on the different perturbations that can happen in industrial settings.

5.5.2 Limitations and Future Research

Some limitations of our research can be mentioned. First, the optimisation method to generate perturbations is based on Genetic Algorithm with specific parameters. Indeed, choices have been made on the reference algorithm (i.e. R-NSGA-III), and on the initial population, the number of generations, the mutations and the survival selection. Also, the algorithm has been ran only one time for each case (perturbing a specific model), but the study could gain in consistency by iterating this process several times. However, running GA requires a lot of computational resources, especially in case of huge data sets. This could be a serious obstacle for companies that do not have the computational capacity to conduct such an empirical study. Even if GA show great performance for multi-objectives optimisation tasks, other otpimisation methods to generate perturbations in UTS, such as Particle Swarm Optimization (PSO) or Generative Adversarial Networks (GAN), could be investigated. In addition, we evaluated our approach for only one perturbation (swapping effect), and we have made choices concerning the scope of the parameters of the perturbations (i.e. the percentage of swapped values and their range). This systematic approach could be applied for other kind of perturbations. Finally, the results of the robustness evaluation aim to be used for studying the robustification possibilities, such as applying countermeasures on the data collection architecture to avoid the perturbations, or data augmentation process to make the models more robust to them. This should be studied in future research.

6

Conclusion and Perspectives

6.1 Conclusion

This dissertation is mainly composed of two parts. The first part of the dissertation proposes and evaluates a data-driven approach to collect data from industrial production plants in a unified manner. To do so, we have first proposed an analytical model of the traffic-load generated through the network for a remote industrial monitoring application, considering a real industrial use-case from our partner Cebi Luxembourg S.A. This analytical study has been conducted on two well-known protocols (i.e. O-MI/O-DF and MQTT) of the industrial community that provide interoperability. It showed that using such open and standardised protocols allows to monitor production tools in real-time. In the meantime, the theoretical performance evaluation showed that the traffic-load generated is still dependent of the implementation choices, necessitating a specific care before the deployment in the real production system. Then, we proposed a generic three-steps strategy that defines the different stages to make a step from vertically-closed production plants with heterogeneous protocols to a fully connected and interoperable architecture based and open standards, reaching the industry 4.0 expectations. We evaluated

this three-steps strategy on pilot productions line of our partner. Performance of three protocols (i.e. O-MI/O-DF,MQTT and OPC-UA), that are pushed by both academic and industrial communities, have been investigated. Results show that protocols implementations and configurations can have an impact in terms of temporal performance. We moreover highlighted, thanks to our partner making us available recent industrial equipment, that a particular attention has to be paid to the hardware aspects of the data collection architecture.

To conclude on this first part of the dissertation, we demonstrated that industrial actors can rely on open protocols to collect data from their production plants for feeding their monitoring and data analysis tools in a unified manner. Also, we demonstrated that the strategy we have developed can be used when designing a new production system or when upgrading a legacy production plant. Nonetheless, a particular attention have to be paid to the different protocols and hardware to be used, by conducting performance evaluation before a real-life deployment, since performance are impacted by these choices.

The second part of this dissertation presents a systematic approach to evaluate the robustness of Machine Learning models in industrial settings. These systematic approaches are based on the artificial injections of perturbations into the data sets. We conducted empirical study on Univariate Time Series (UTS) from state-of-the-art data sets of the well-known UCR repository, and on state-of-the-art Machine Learning models, that reach great performance for classification tasks on these data sets. The first approach proposes a coarse-grained evaluation of models' robustness, following a random algorithm method to craft the different perturbations (swapping effect and dropping effect) that are injected in the UCR data sets. Then, the models are tested on these degraded data in order to figure-out their robustness under such perturbations. The results show great disparity of models' robustness, and pointed-out the importance of such robustness evaluations before deploying AI-based models in industrial settings. The second approach is a finer-grained extension of the previous one. Actually, based on the diversity of the models' robustness from the coarse-grained evaluation, we extended the approach using an optimisation method, based on genetic algorithm, to generate perturbations. We focused on the swapping effect, and evaluated the robustness of each model individually, to find their limit cases for which they are prone to misclassification. The results show that this optimisation method is, most of the time, more efficient to impact the robustness of the models.

To sum up, let us discuss about those two parts that need to be merged for implementing a unified and robust data-driven approach. We explored the different steps and solutions to build unified and robust data-driven production plants, but this digital transformation of industrial productions plants brings-out many challenges to tackle. Actually, even if we demonstrated that unified data-driven

approaches can be used for industrial purposes, a specific attention as to be paid before totally relying on it in real-life environment. We have shown that open protocols providing a unified data collection architecture, still have some lacks in terms of performance, and have to be investigated with regards to scalability aspects. It is especially the case for companies willing to upgrade legacy production plants, since it needs some retro engineering stages, but also intermediate steps for accessing the data and providing interoperability between heterogeneous proprietary systems from different ages. In addition, we have pointed-out the fact the Machine Learning solutions, although showing outstanding results in the literature, are still impacted by perturbations likely to happen in industrial environments. However, since performance of open protocols and hardware shows a great diversity, relying on such AI-based solutions to analyse the data upon on these open protocols, seems to remain an obstacle for industrial companies. Thus, solutions exist to tackle the different challenges for accessing and analysing the production data, but there is still room for improvements to use these components in collaboration, as a unified and robust industrial ecosystem.

Thanks to our collaboration with Cebi Luxembourg S.A., we can see that there are not only scientific barriers to tackle, but also in terms of cultural change. Indeed, such changes in companies take a lot of time to be adopted and accepted by the different levels of the enterprise hierarchy, more precisely the management board as well as the operators. This paradigm shift implies a lot of strategic changes at the enterprise levels. Of course, such an industrial revolution induces expenses for Research and Development, but also for renewing/upgrading the production tools as well as the business tools (such as applications). It also needs a very deep expertise from the people designing and developing the different components to reach the new industrial expectations. This expertise is sometimes difficult to find in the within the companies that were, until today, focused on investing and developing new products. This gap of expertise is hard to fill internally and needs sometimes to be delegated to tiers companies, rising questions of cost, intellectual properties and confidentiality. Moreover, and not the least, human aspect has to be taken into account. This changes on the manner of managing the production, on collecting the data, and analysing it obviously implies changes for operators on the machines. These changes lead to a need of sensitisation and awareness to these new technologies in order to be accepted by the operators, as well as a need in formation/teaching.

Finally, we are going towards a unified and data-driven approach for the digital transformation of production plants, but it still remains some scientific and society challenges to overcome.

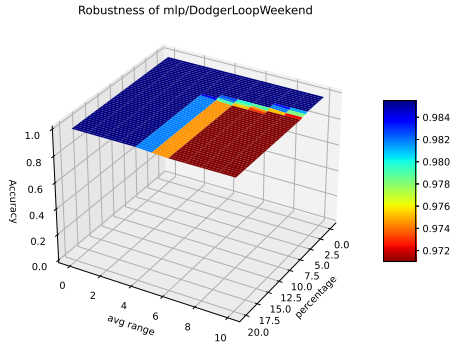
6.2 Perspectives

Looking at the studies that compose this dissertation, there are still researches to be conducted on the different topics, that we would like to explore in the future. Concerning the unified data collection aspect, some paradigms could be investigated in order to prevent some of the network perturbations to happen. Technologies such as SDN and TSN could be part of a solution for improving the network performance in industrial networks, that are prone to changes over time, leading to potential alterations of the data quality impacting the robustness of ML models. Evaluating the implications and the benefits of implementing/deploying such network configurations technologies in the case of the digital transformation of production plants, could have a real interest. Concerning the robustness of the Machine Learning models, there is still a lot of possibilities to explore. In our experimental study, we only focus on two types of network perturbations, that are swapping and dropping effect. However, future researches aiming at modelling and evaluating the models robustness for other kinds of perturbations (e.g. physical perturbations of the process) constitute an interesting extension of our researches. In addition, other methods to generate perturbations should be investigated and compared. Then, the robustification of the data-driven approach, by considering solutions such as TSN to avoid certain perturbations, or by robustifying the models using techniques like data augmentation, remains a challenge to be tackled. Overall, we are confident, considering the remaining challenges, that we can smoothly overcome them to propose a unified and robust data-driven approach to digitise the production plants in the era of Industry 4.0.

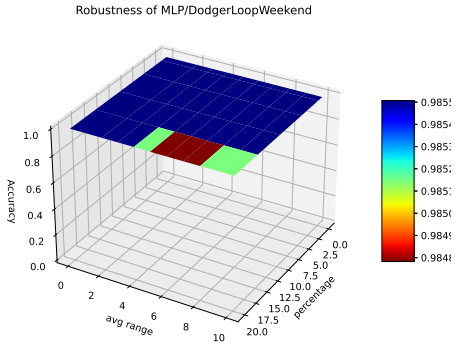
A

Appendix

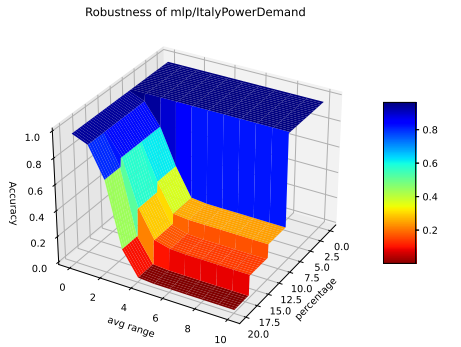
A.1 Appendices of Chapter V



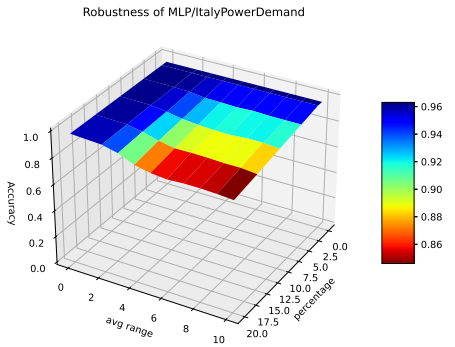
(**a**) GA



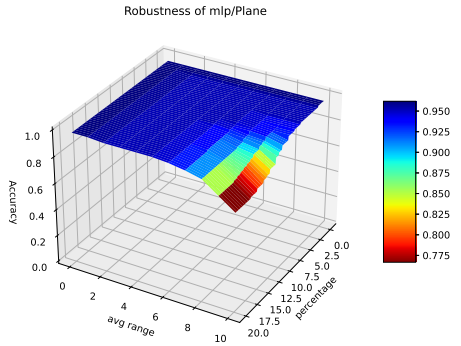
(**b**) Random



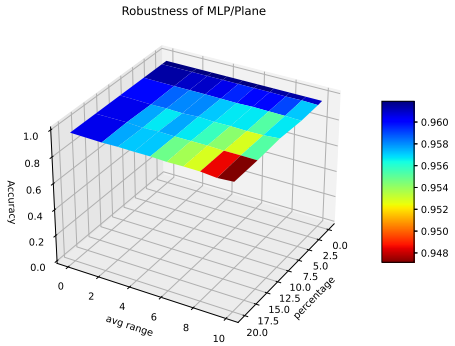
(**c**) GA



(**d**) Random

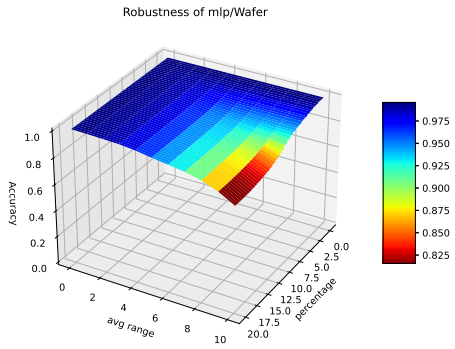


(**e**) GA

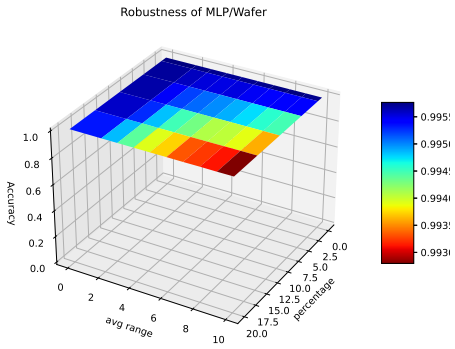


(**f**) Random

Figure A.1: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for MLP classifier

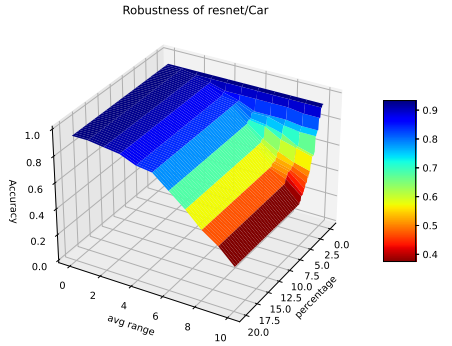


(**g**) GA

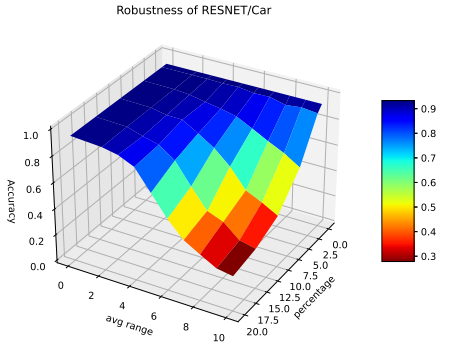


(**h**) Random

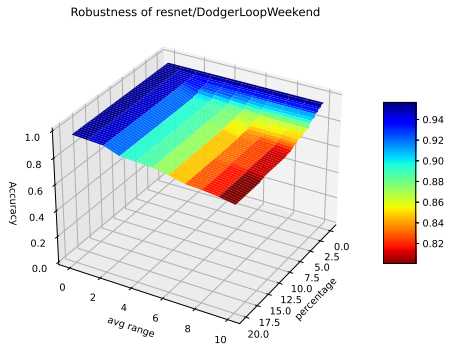
Figure A.2: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for MLP classifier



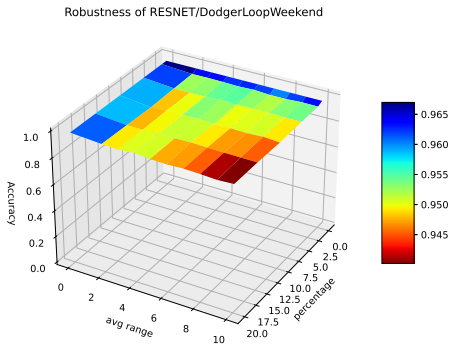
(**a**) GA



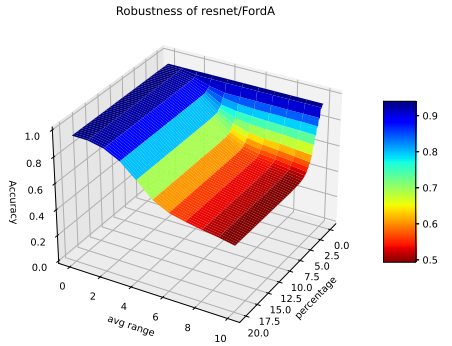
(**b**) Random



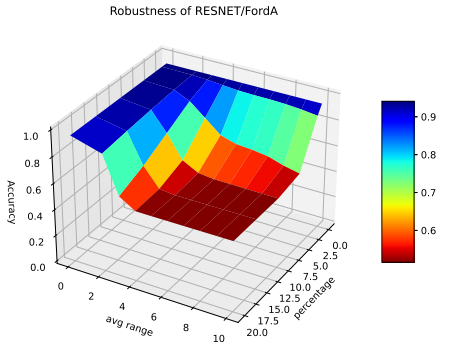
(**c**) GA



(**d**) Random

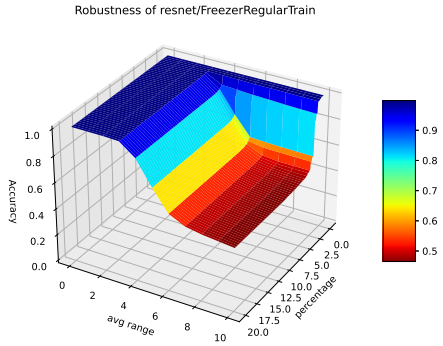


(**e**) GA

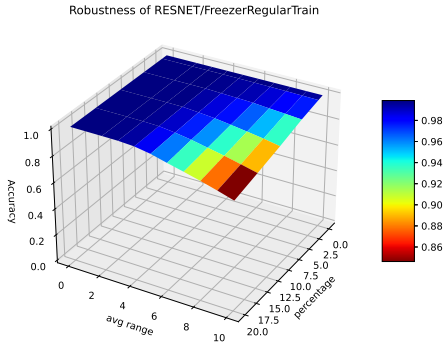


(**f**) Random

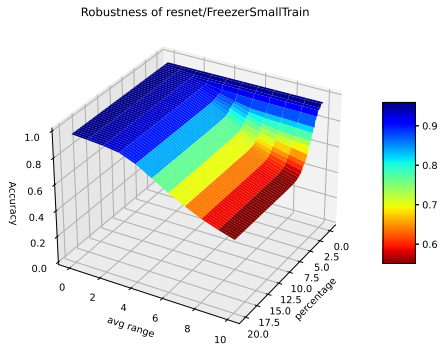
Figure A.3: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for ResNet classifier



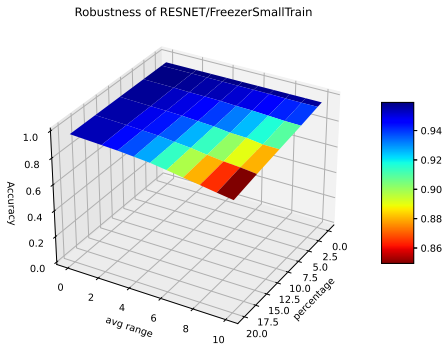
(**g**) GA



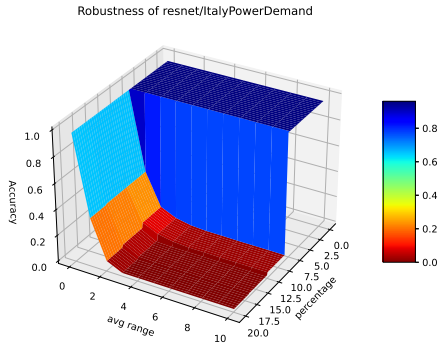
(**h**) Random



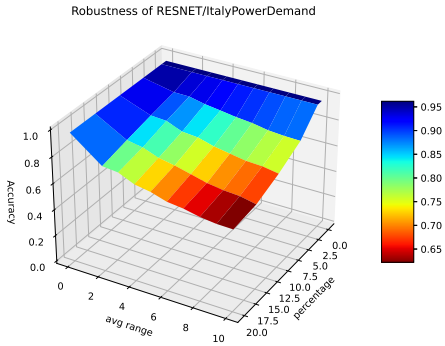
(**i**) GA



(**j**) Random

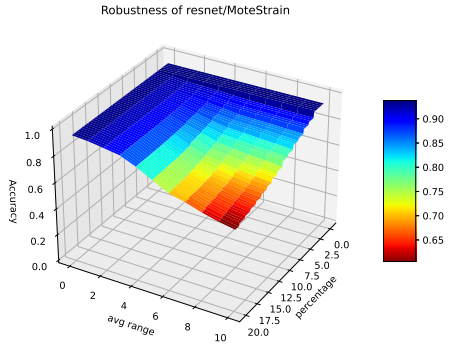


(**k**) GA

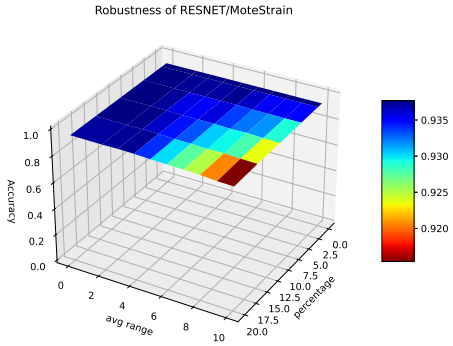


(**l**) Random

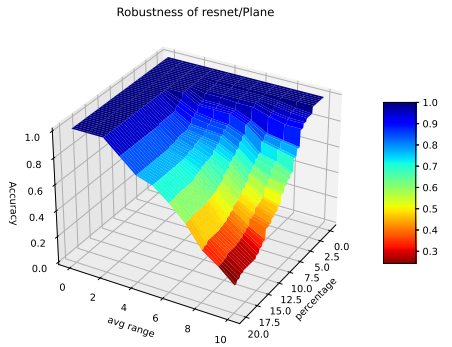
Figure A.3: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for ResNet classifier



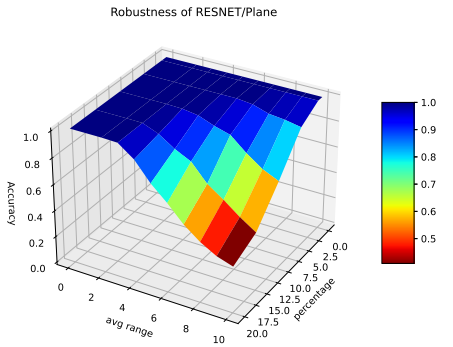
(**m**) GA



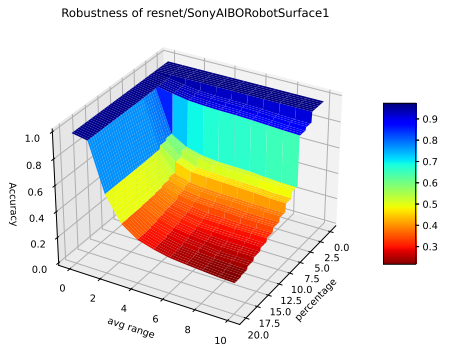
(**n**) Random



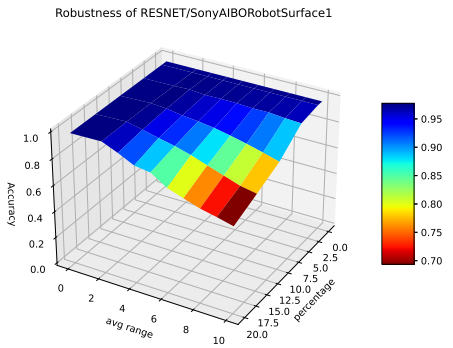
(**o**) GA



(**p**) Random

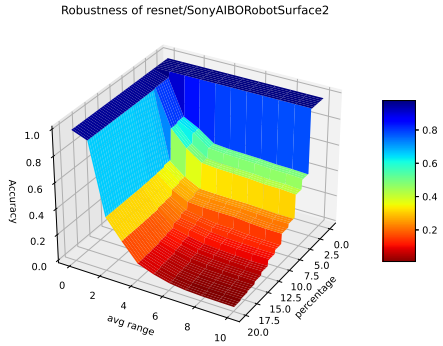


(**q**) GA

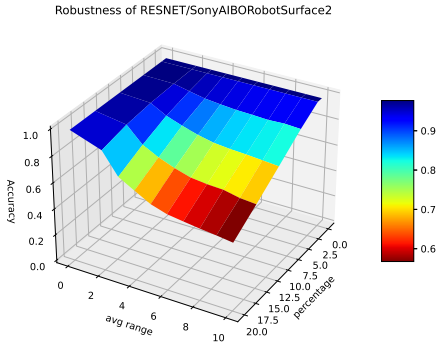


(**r**) Random

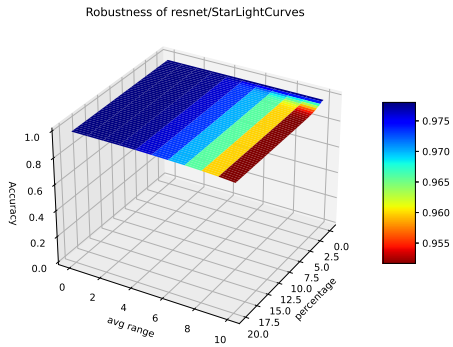
Figure A.3: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for ResNet classifier



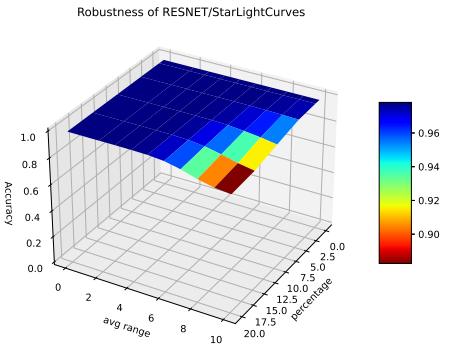
(**s**) GA



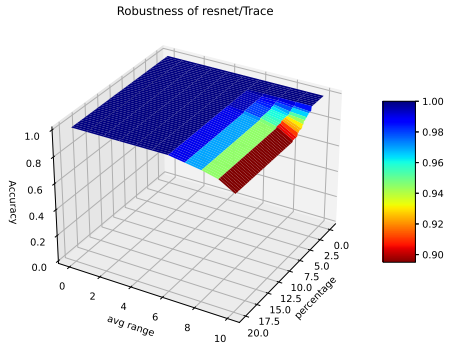
(**t**) Random



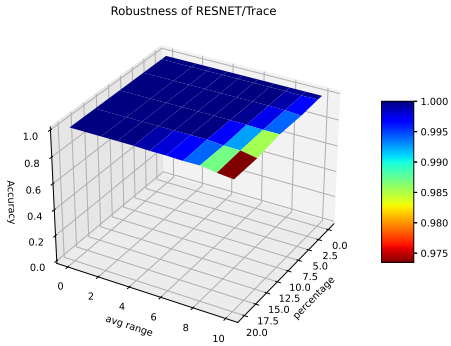
(**u**) GA



(**v**) Random

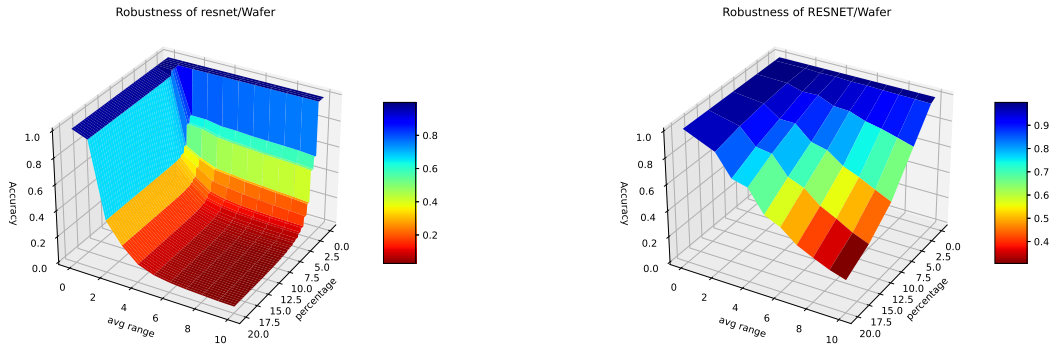


(**w**) GA



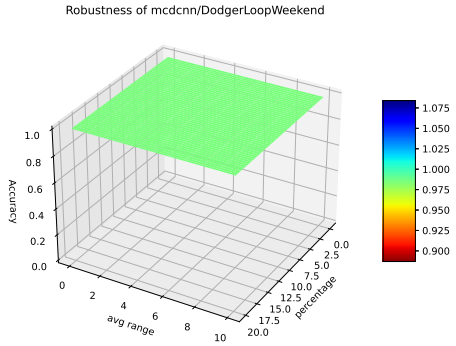
(**x**) Random

Figure A.3: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for ResNet classifier

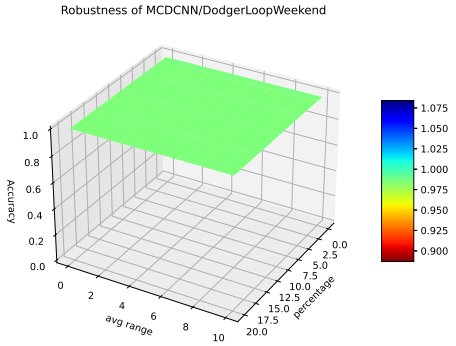


(**y**) GA (**z**) Random

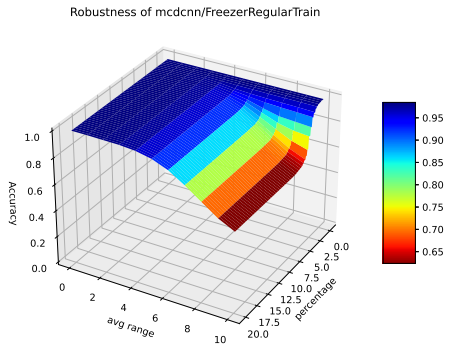
Figure A.3: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for ResNet classifier



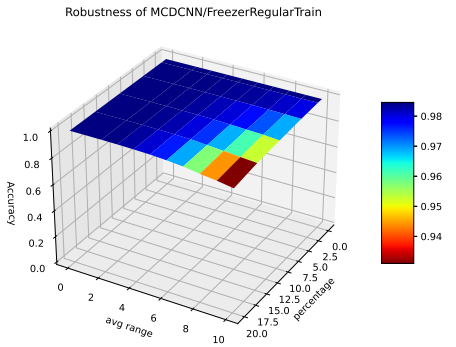
(**a**) GA



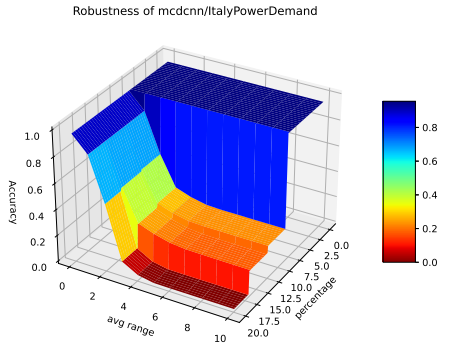
(**b**) Random



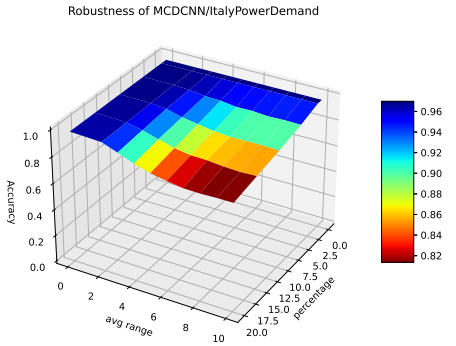
(**c**) GA



(**d**) Random

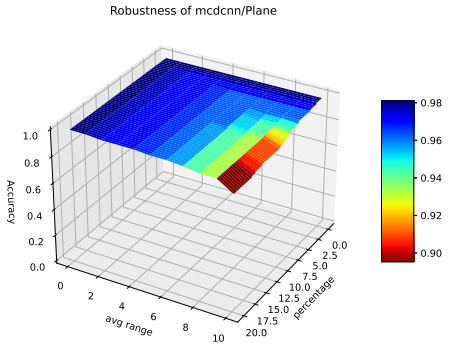


(**e**) GA

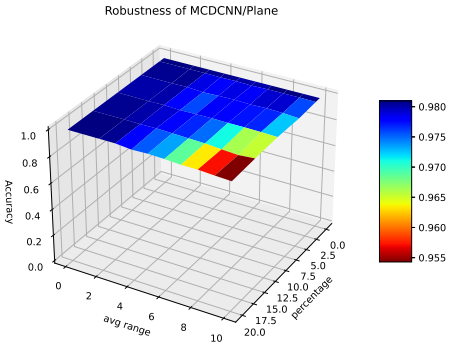


(**f**) Random

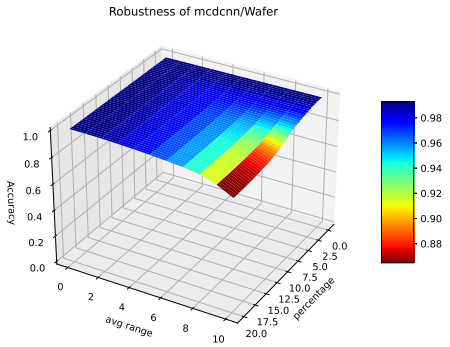
Figure A.4: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for MCDCNN classifier



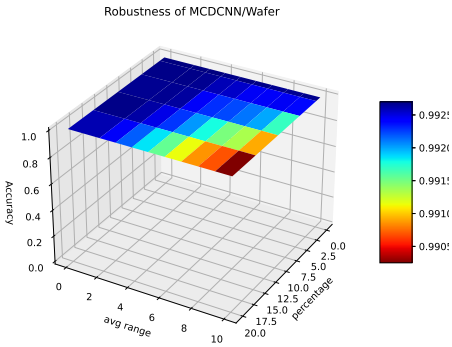
(**g**) GA



(**h**) Random

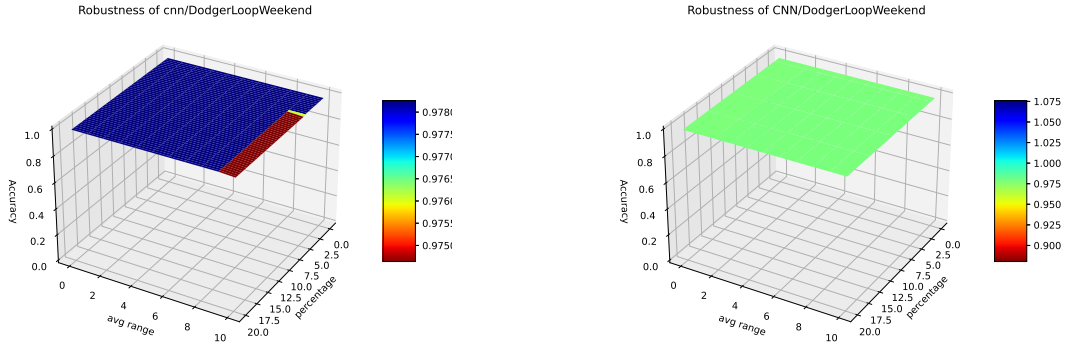


(**i**) GA



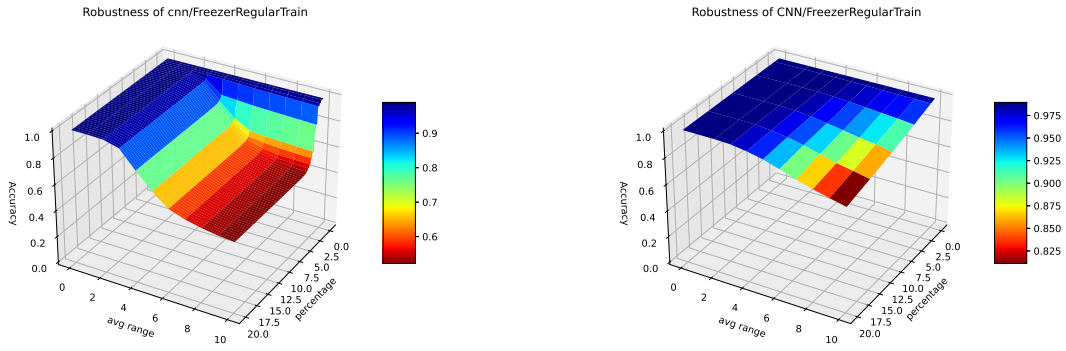
(**j**) Random

Figure A.4: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for MCDCNN classifier



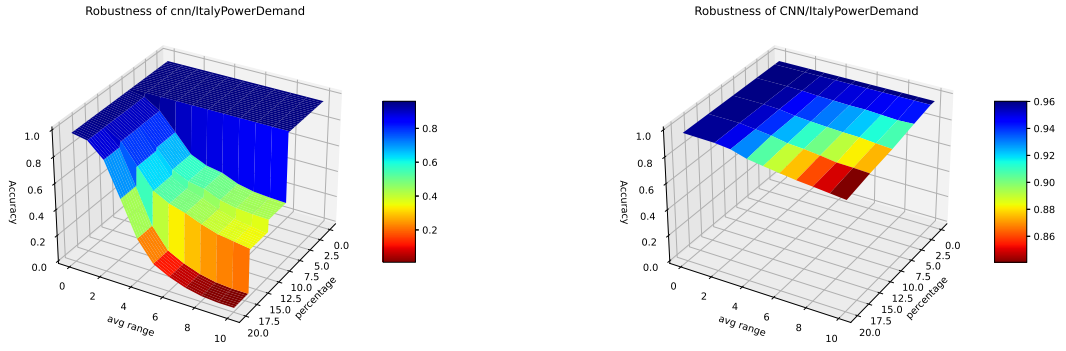
(**a**) GA

(**b**) Random



(**c**) GA

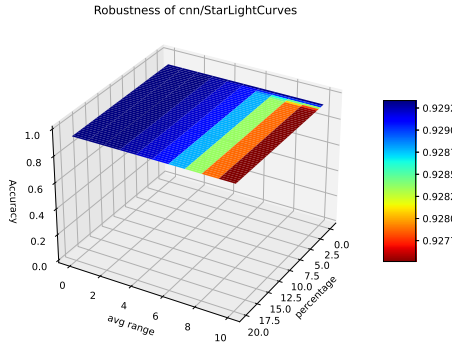
(**d**) Random



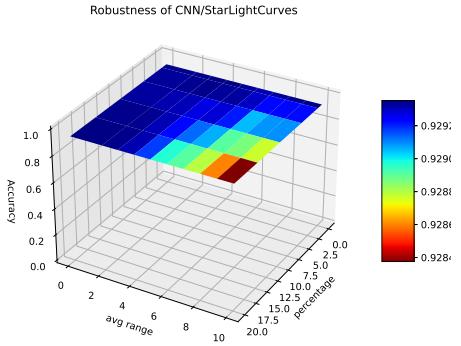
(**e**) GA

(**f**) Random

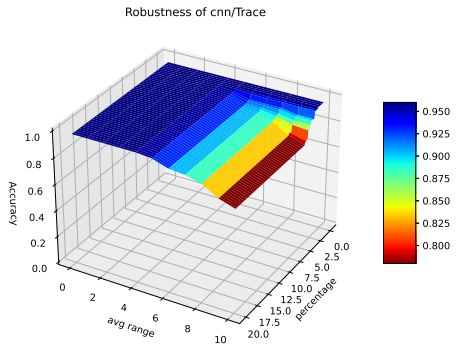
Figure A.5: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for CNN classifier



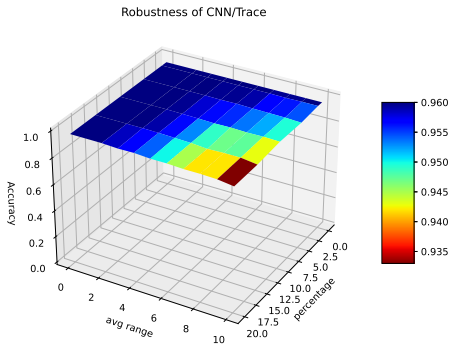
(**g**) GA



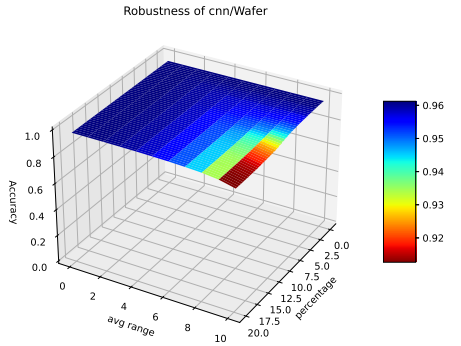
(**h**) Random



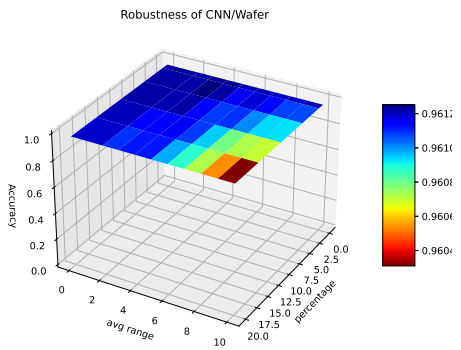
(**i**) GA



(**j**) Random

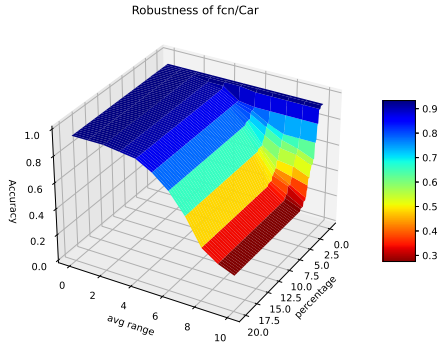


(**k**) GA

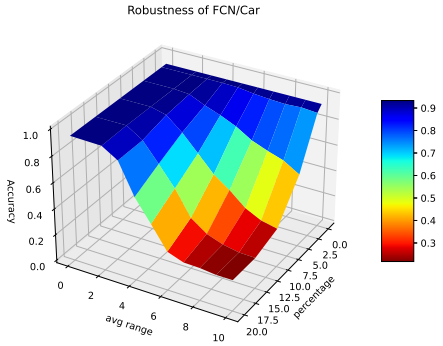


(**l**) Random

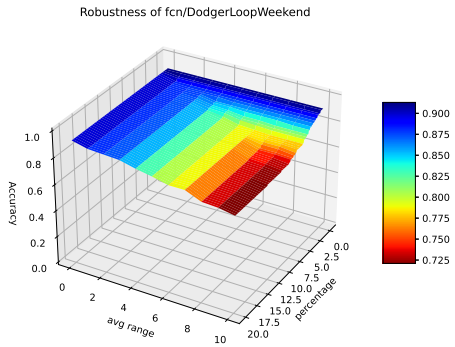
Figure A.5: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for CNN classifier



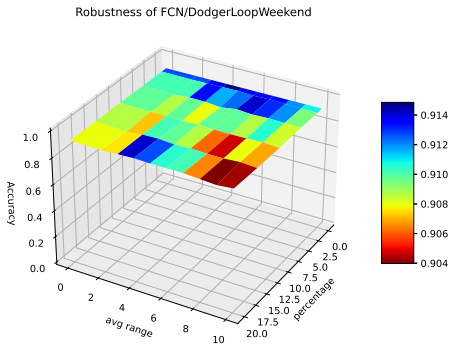
(**a**) GA



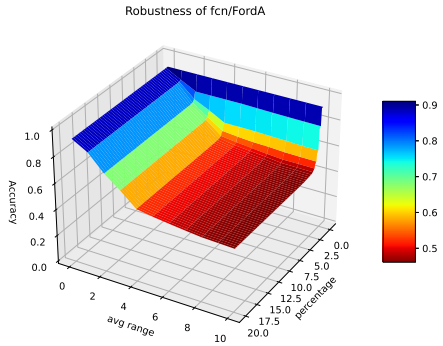
(**b**) Random



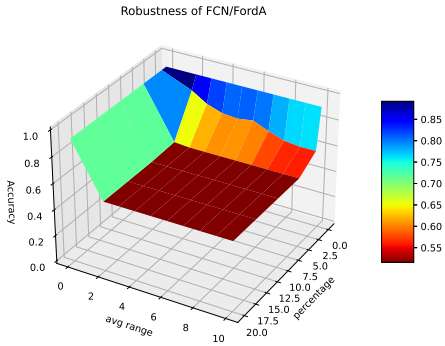
(**c**) GA



(**d**) Random

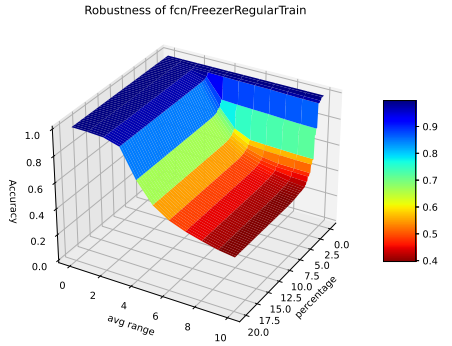


(**e**) GA

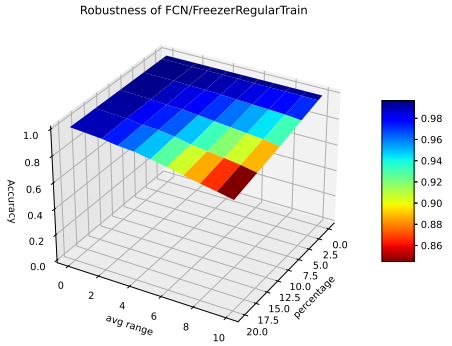


(**f**) Random

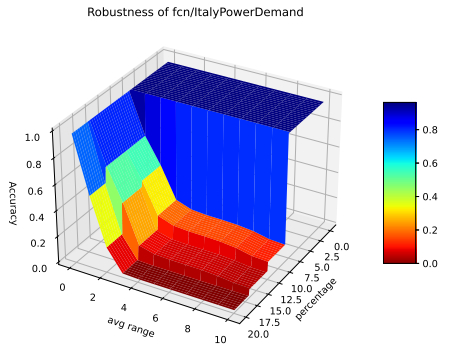
Figure A.6: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for FCN classifier



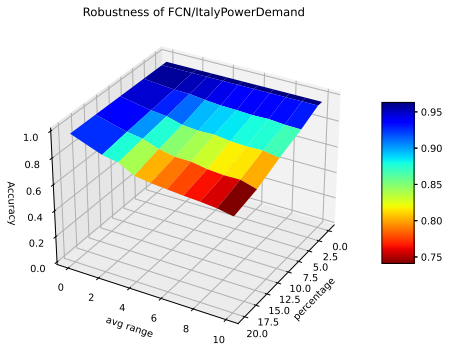
(**g**) GA



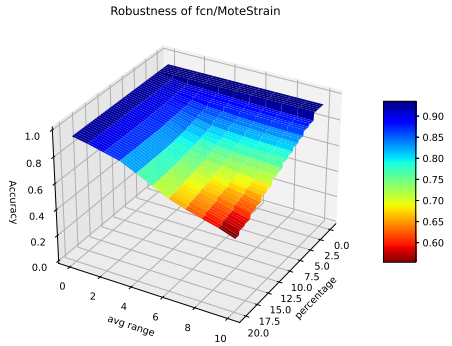
(**h**) Random



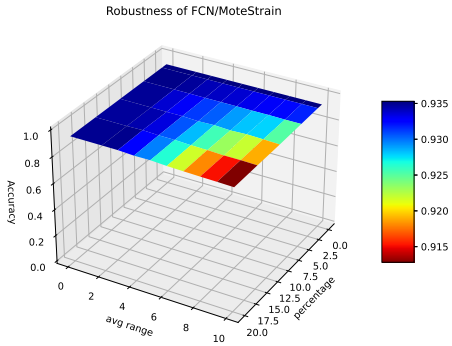
(**i**) GA



(**j**) Random

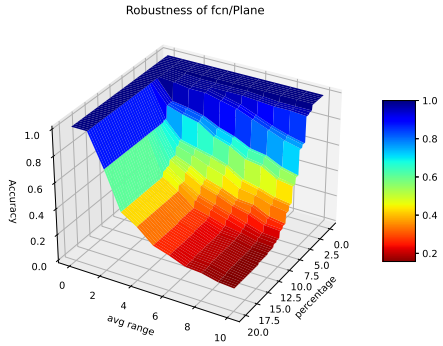


(**k**) GA

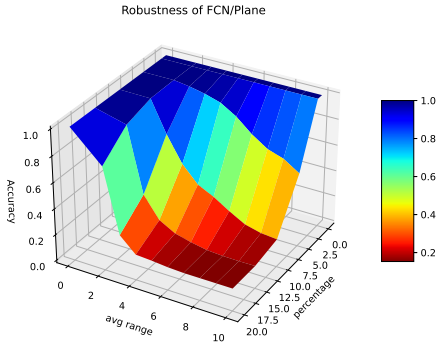


(**l**) Random

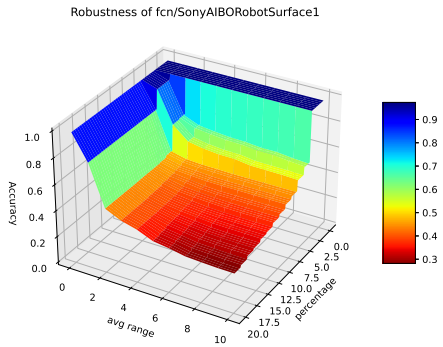
Figure A.6: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for FCN classifier



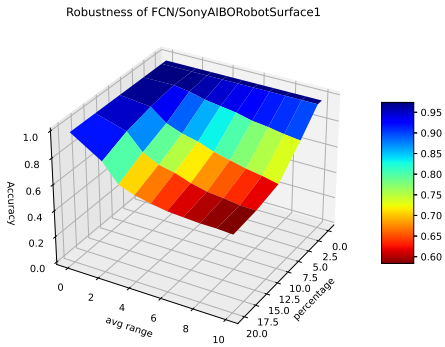
(**m**) GA



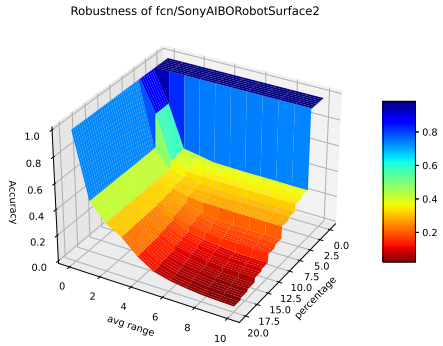
(**n**) Random



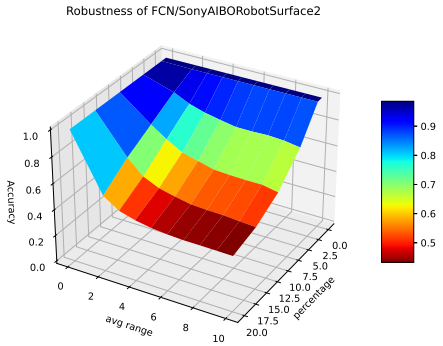
(**o**) GA



(**p**) Random

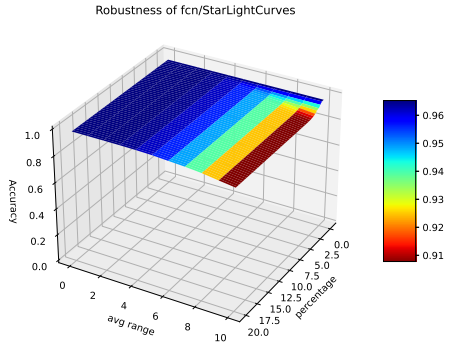


(**q**) GA

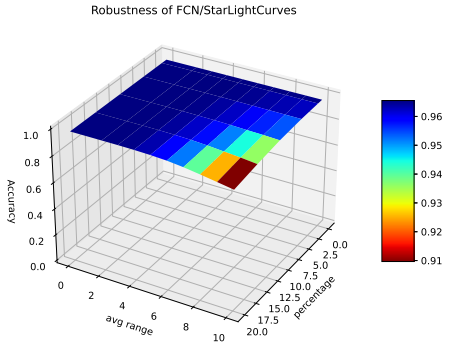


(**r**) Random

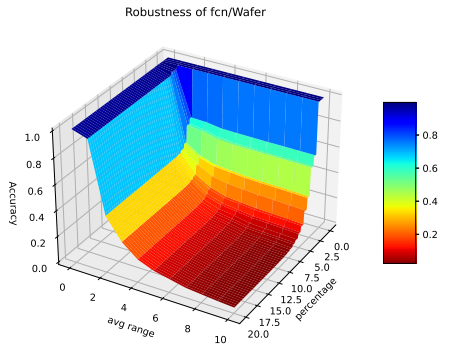
Figure A.6: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for FCN classifier



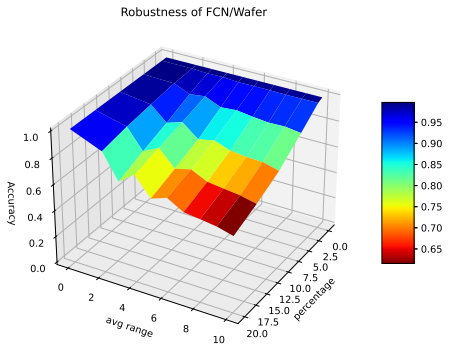
(**s**) GA



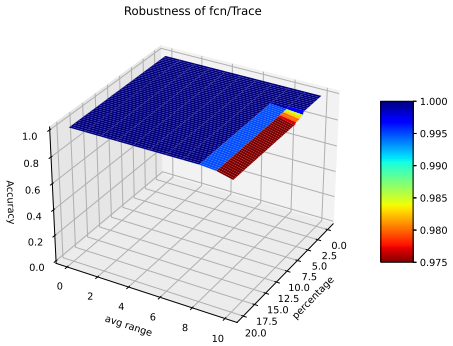
(**t**) Random



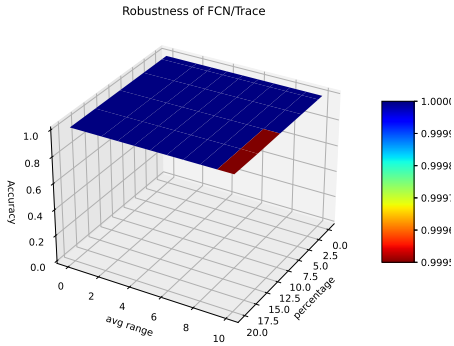
(**u**) GA



(**v**) Random

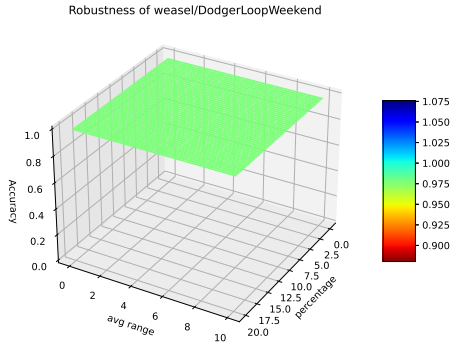


(**w**) GA

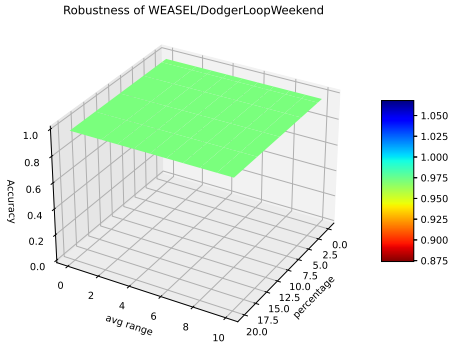


(**x**) Random

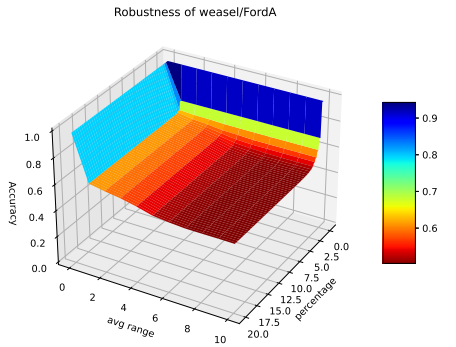
Figure A.6: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for FCN classifier



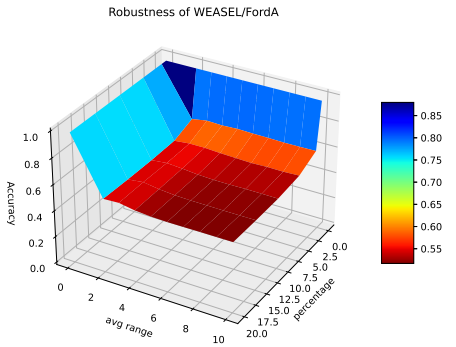
(**a**) GA



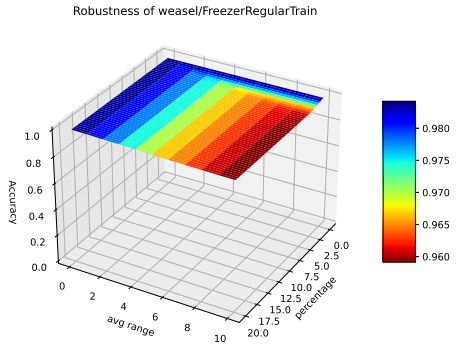
(**b**) Random



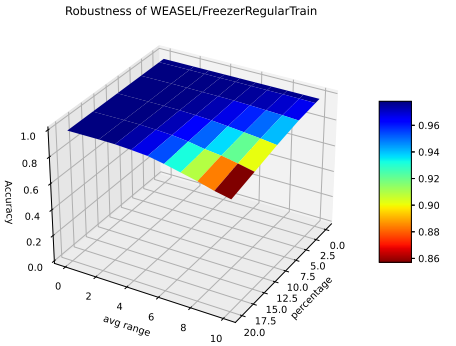
(**c**) GA



(**d**) Random

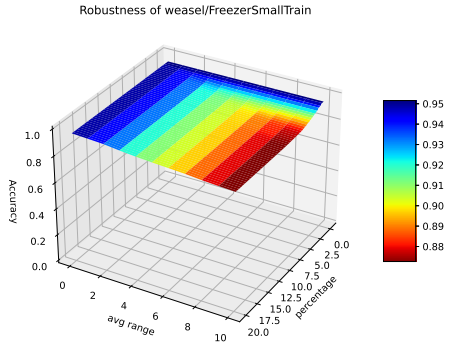


(**e**) GA

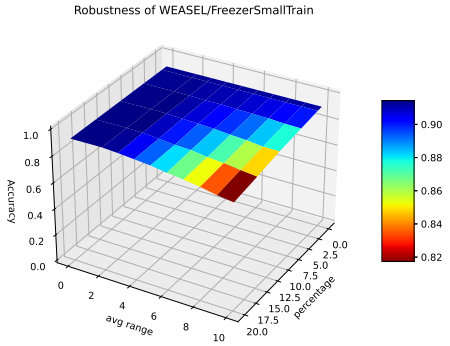


(**f**) Random

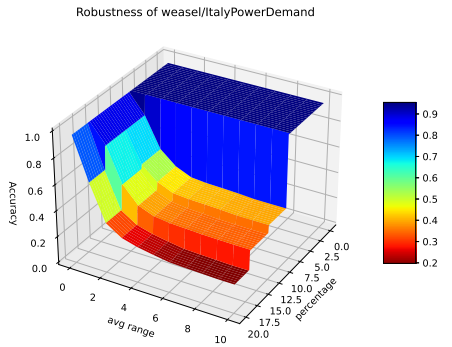
Figure A.7: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for WEASEL classifier



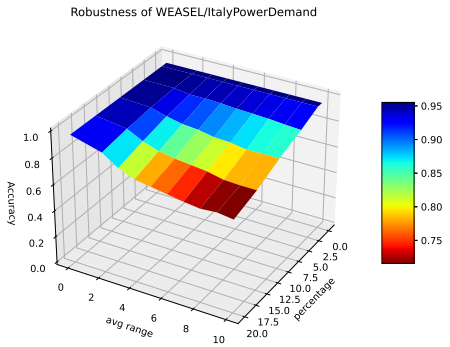
(**g**) GA



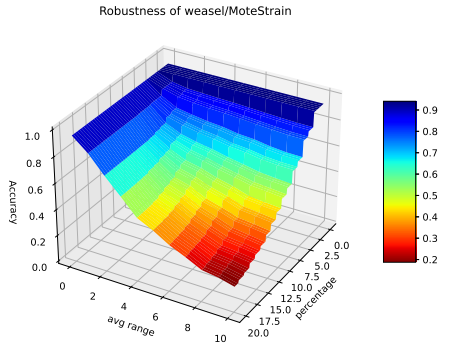
(**h**) Random



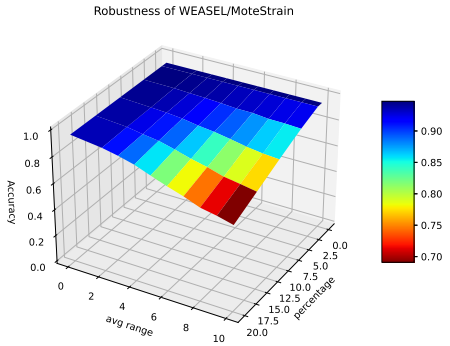
(**i**) GA



(**j**) Random

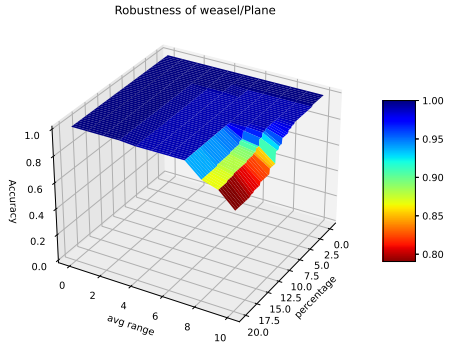


(**k**) GA

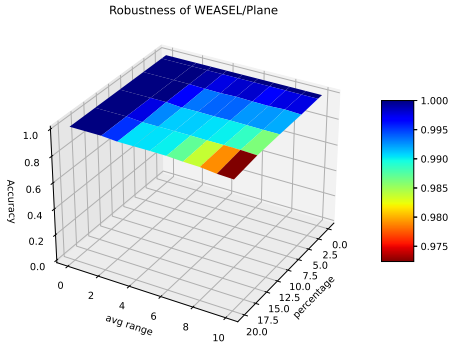


(**l**) Random

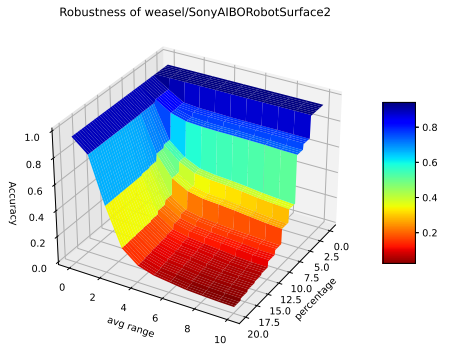
Figure A.7: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for WEASEL classifier



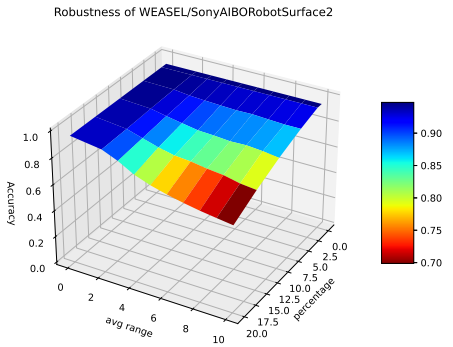
(**m**) GA



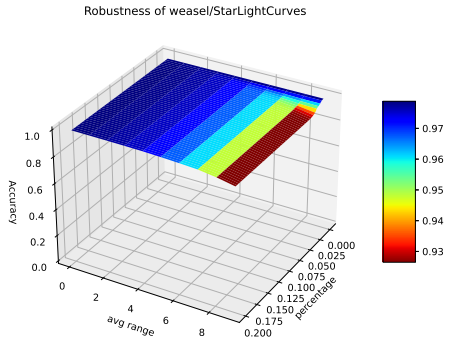
(**n**) Random



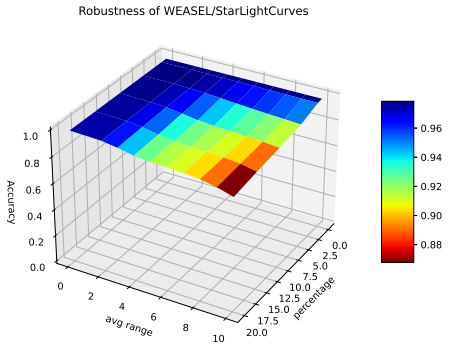
(**o**) GA



(**p**) Random

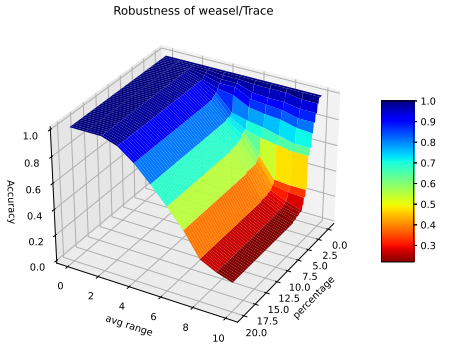


(**q**) GA

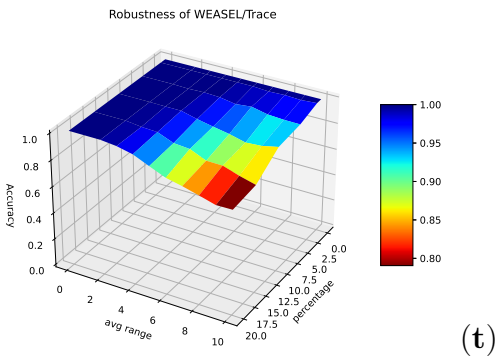


(**r**) Random

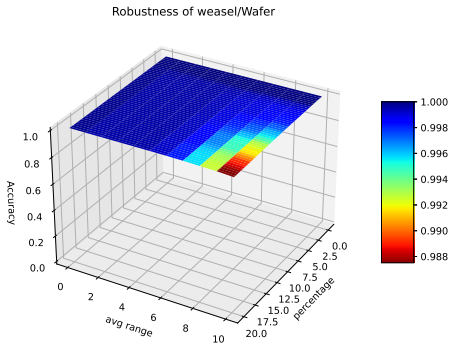
Figure A.7: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for WEASEL classifier



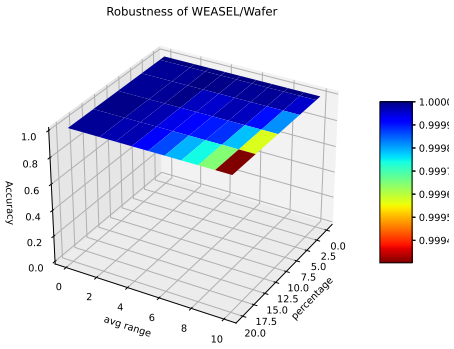
(**s**) GA



Random (**t**)



(**u**) GA



(**v**) Random

Figure A.7: Robustness comparison between perturbations generated by Genetic Algorithm (GA) and Random algorithm for WEASEL classifier

List of publications

Papers included in the dissertation

- [BRLT21]: *Paul-Lou Benedick, Jérémy Robert, and Yves Le Traon. A systematic approach for evaluating artificial intelligence models in industrial settings. Sensors, 21(18), 2021.*
- [BRLT19]: *Paul-Lou Benedick, Jérémy Robert, and Yves Le Traon. Trident: A three-steps strategy to digitise an industrial system for stepping into industry 4.0. In IECON 2019-45th Annual Conference of the IEEE Industrial Electronics Society, volume 1, pages 3037–3042. IEEE, 2019.*
- [BRLTK18]: *Paul-Lou Benedick, Jérémy Robert, Yves Le Traon, and Sylvain Kubler. O-mi/o-df vs. mqtt: A performance analysis. In 2018 IEEE Industrial Cyber-Physical Systems (ICPS), pages 153–158. IEEE, 2018.*

List of figures

List of tables

List of algorithms

Bibliography

[15990] Ieee standard glossary of software engineering terminology. *IEEE Std 610.12-1990*, pages 1–84, Dec 1990.

[AIM10] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.

[AP10] Jeremy Adler and Ingela Parmryd. Quantifying colocalization by correlation: the pearson correlation coefficient is superior to the mander's overlap coefficient. *Cytometry Part A*, 77(8):733–742, 2010.

[APBRPOM15] Juan C Alvarado-Pérez, Harold Bolaños-Ramírez, Diego H Peluffo-Ordóñez, and S Murillo. Knowledge discovery in databases from a perspective of intelligent information visualization. In *2015 20th Symposium on Signal Processing, Images and Computer Vision (STSIVA)*, pages 1–7. IEEE, 2015.

[BBET+15] Payam M Barnaghi, Maria Bermudez-Edo, Ralf Tönjes, et al. Challenges for quality of data in smart cities. *J. Data and Information Quality*, 6(2-3):6–1, 2015.

[BCH08] Jacob Benesty, Jingdong Chen, and Yiteng Huang. On the importance of the pearson correlation coefficient in noise reduction. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(4):757–765, 2008.

[BRLT19] Paul-Lou Benedick, Jérémy Robert, and Yves Le Traon. Trident: A three-steps strategy to digitise an industrial system for stepping into industry 4.0. In *IECON 2019-45th Annual Conference of the IEEE Industrial Electronics Society*, volume 1, pages 3037–3042. IEEE, 2019.

[BRLT21] Paul-Lou Benedick, Jérémy Robert, and Yves Le Traon. A systematic approach for evaluating artificial intelligence models in industrial settings. *Sensors*, 21(18), 2021.

[BRLTK18] Paul-Lou Benedick, Jeremy Robert, Yves Le Traon, and Sylvain Kubler. O-mi/o-df vs. mqtt: A performance analysis. In *2018 IEEE Industrial Cyber-Physical Systems (ICPS)*, pages 153–158. IEEE, 2018.

[BS15] Hongyu Pei Breivold and Kristian Sandström. Internet of things for industrial automation–challenges and technical solutions. In *Data Science and Data Intensive Systems (DSDIS), 2015 IEEE International Conference on*, pages 532–539. IEEE, 2015.

[BSB+19] Dietmar Bruckner, Marius-Petru Stănică, Richard Blair, Sebastian Schriegel, Stephan Kehrer, Maik Seewald, and Thilo Sauter. An introduction to opc ua tsn for industrial communication systems. *Proceedings of the IEEE*, 2019.

[CCM10] Salvatore Cavalieri, Giovanni Cutuli, and Salvatore Monteleone. Evaluating impact of security on opc ua performance. In *Human System Interactions (HSI), 2010 3rd Conference on*, pages 687–694. IEEE, 2010.

[CJM+19] Richard Chen, Filip Jankovic, Nikki Marinsek, Luca Foschini, Lampros Kourtis, Alessio Signorini, Melissa Pugh, Jie Shen, Roy Yaari, Vera Maljkovic, Marc Sunga, Han Hee Song, Hyun Joon Jung, Belle Tseng, and Andrew Trister. Developing measures of cognitive impairment in the real world from consumer-grade multimodal sensor streams. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '19, pages 2145–2155, New York, NY, USA, 2019. Association for Computing Machinery.

[DBK+19] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Annh Ratanamahatana, and Eamonn Keogh. The ucr time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2019.

[DCJ15] Lars Durkop, Bjorn Czybik, and Jurgen Jasperneite. Performance evaluation of m2m protocols over cellular networks in a lab environment. In *Intelligence in Next Generation Networks (ICIN), 2015 18th International Conference on*, pages 70–75. IEEE, 2015.

[DdCGH16] Roberto Alexandre Dias, Gregory das Chagas Gomes, and Marcelo Lobo Heldwin. Evaluation of opc-ua framework applied to microgrid energy dispatch management. *International Journal of Computer Applications*, 154(7), 2016.

[dRSd+17] Jesus Martinez del Rincon, Maria J. Santofimia, Xavier del Toro, Jesus Barba, Francisca Romero, Patricia Navas, and Juan C. Lopez. Non-linear classifiers applied to eeg analysis for epilepsy seizure detection. *Expert Systems with Applications*, 86:99 – 112, 2017.

[EAAV20] Absalom E Ezugwu, Olawale J Adeleke, Andronicus A Akinyelu, and Serestina Viriri. A conceptual comparison of several metaheuristic algorithms on continuous optimisation problems. *Neural Computing and Applications*, 32(10):6207–6251, 2020.

[FF19] Harald Foidl and Michael Felderer. Risk-based data validation in machine learning-based software systems. In *Proceedings of the 3rd ACM SIGSOFT International Workshop on Machine Learning Techniques for Software Quality Evaluation*, pages 13–18. ACM, 2019.

[FFR+18] Paolo Ferrari, Alessandra Flammini, Stefano Rinaldi, Emiliano Sisinni, Davide Maffei, and Matteo Malara. Impact of quality of service on cloud based industrial iot applications with opc ua. *Electronics*, 7(7):109, 2018.

[FFW+19] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, 2019.

[FKB14] Kary Främling, Sylvain Kubler, and Andrea Buda. Universal messaging standards for the iot from a lifecycle management perspective. *IEEE Internet of things journal*, 1(4):319–327, 2014.

[GAD17] Venkat Gudivada, Amy Apon, and Junhua Ding. Data quality considerations for big data and machine learning: Going beyond data cleaning and transformations. *International Journal on Advances in Software*, 10(1):1–20, 2017.

[Gam17] John Cristian Borges Gamboa. Deep learning for time-series analysis. *arXiv preprint arXiv:1701.01887*, 2017.

[GBMP13] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660, 2013.

[GCG+20] Salah Ghamizi, Maxime Cordy, Martin Gubri, Mike Papadakis, Andrey Boystov, Yves Le Traon, and Anne Goujon. Search-based adversarial testing and improvement of constrained credit scoring systems. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 1089–1100, 2020.

[GCL18] Ashish Ghosh, Debasrita Chakraborty, and Anwesha Law. Artificial intelligence in internet of things. *CAAI Transactions on Intelligence Technology*, 3(4):208–218, 2018.

[GPP16] Sten Grüner, Julius Pfrommer, and Florian Palm. Restful industrial communication with opc ua. *IEEE Transactions on Industrial Informatics*, 12(5):1832–1841, 2016.

[Gro17a] The Open Group. Open data-format (o-df) specification: https://publications.opengroup.org/c14a, 2017.

[Gro17b] The Open Group. Open messaging-interface (o-mi) specification: https://publications.opengroup.org/c14b, 2017.

[GSDH17] Zhiqiang Ge, Zhihuan Song, Steven X Ding, and Biao Huang. Data mining and analytics in the process industry: The role of machine learning. *Ieee Access*, 5:20590–20616, 2017.

[GSG13] Zhiqiang Ge, Zhihuan Song, and Furong Gao. Review of recent research on data-based process monitoring. *Industrial & Engineering Chemistry Research*, 52(10):3543–3562, 2013.

[HL18] Hua Huang and Shan Lin. Widet: Wi-fi based device-free passive person detection with deep convolutional neural networks. In *Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, MSWIM '18, pages 53–60, New York, NY, USA, 2018. Association for Computing Machinery.

[HSK08] Tom Hannelius, Mikko Salmenpera, and Seppo Kuikka. Roadmap to adopting opc ua. In *2008 6th IEEE International Conference on Industrial Informatics*, pages 756–761. IEEE, 2008.

[IFW+19] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. Muller. Adversarial attacks on deep neural networks for time series classification. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, July 2019.

[KGS09] Petr Kadlec, Bogdan Gabrys, and Sibylle Strandt. Data-driven soft sensors in the process industry. *Computers & chemical engineering*, 33(4):795–814, 2009.

[KMM18] S. Karimi-Bidhendi, F. Munshi, and A. Munshi. Scalable classification of univariate and multivariate time series. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 1598–1605, Dec 2018.

[KRH+16] Sylvain Kubler, Jérémy Robert, Ahmed Hefnawy, Chantal Cherifi, Abdelaziz Bouras, and Kary Främling. Iot-based smart parking system for sporting event management. In *Proceedings of the 13th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pages 104–114. ACM New York, 2016.

[KRKLT17] Niklas Kolbe, Jérémy Robert, Sylvain Kubler, and Yves Le Traon. Proficient: Productivity tool for semantic interoperability in an open iot ecosystem. In *Proceedings of the 14th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, 2017.

[KWLJ17] Waqas Ali Khan, Lukasz Wisniewski, Dorota Lang, and Jürgen Jasperneite. Analysis of the requirements for offering industrie 4.0 applications as a cloud service. In *Industrial Electronics (ISIE), 2017 IEEE 26th International Symposium on*, pages 1181–1188. IEEE, 2017.

[LCA+16] Andre Luckow, Matthew Cook, Nathan Ashcraft, Edwin Weill, Emil Djerekarov, and Bennie Vorster. Deep learning in the automotive industry: Applications and tools. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 3759–3768. IEEE, 2016.

[LDSP18] Jay Lee, Hossein Davari, Jaskaran Singh, and Vibhor Pandhare. Industrial artificial intelligence for industry 4.0-based manufacturing systems. *Manufacturing letters*, 18:20–23, 2018.

[LFK+14] Heiner Lasi, Peter Fettke, Hans-Georg Kemper, Thomas Feld, and Michael Hoffmann. Industry 4.0. *Business & information systems engineering*, 6(4):239–242, 2014.

[Liu10] Yu-Hsin Liu. Different initial solution generators in genetic algorithms for solving the probabilistic traveling salesman problem. *Applied mathematics and computation*, 216(1):125–137, 2010.

[MBSRJ18] Karl Øyvind Mikalsen, Filippo Maria Bianchi, Cristina Soguero-Ruiz, and Robert Jenssen. Time series cluster kernel for learning similarities between multivariate time series with missing data. *Pattern Recognition*, 76:569 – 581, 2018.

[MC12] John MacGregor and Ali Cinar. Monitoring, fault diagnosis, fault-tolerant control and optimization: Data driven methods. *Computers & Chemical Engineering*, 47:111–120, 2012.

[MRB+18] Mohammad Saeid Mahdavinejad, Mohammadreza Rezvan, Mohammadamin Barekatain, Peyman Adibi, Payam Barnaghi, and Amit P Sheth. Machine learning for internet of things data analysis: A survey. *Digital Communications and Networks*, 4(3):161–175, 2018.

[MSJR19] Anthony D. McDonald, Farzan Sasangohar, Ashish Jatav, and Arjun H. Rao. Continuous monitoring and detection of post-traumatic stress disorder (ptsd) triggers among veterans: A supervised machine learning approach. *IISE Transactions on Healthcare Systems Engineering*, 9(3):201–211, 2019.

[MSS18] Salome Maro, Jan-Philipp Steghöfer, and Miroslaw Staron. Software traceability in the automotive domain: Challenges and solutions. *Journal of Systems and Software*, 141:85–110, 2018.

[NDJ+16] Zilvinas Nakutis, Vytautas Deksnys, Ignas Jarusevicius, Vilius Dambrauskas, Gediminas Cincikas, and Alenas Kriauceliunas. Round-trip delay estimation in opc ua server-client communication channel. *Elektronika ir Elektrotechnika*, 22(6):80–84, 2016.

[NKK17] Jane Y. Nancy, Nehemiah H. Khanna, and Arputharaj Kannan. A bio-statistical mining approach for classifying multivariate clinical time series data observed at irregular intervals. *Expert Systems with Applications*, 78:283 – 300, 2017.

[NSP17] Lavinia Năstase, Ionu Eugen Sandu, and Nirvana Popescu. An experimental evaluation of application layer protocols for the internet of things. *Studies in Informatics and Control*, 26(4):403–412, 2017.

[ODPL20] Izaskun Oregi, Javier Del Ser, Aritz Pérez, and José A. Lozano. Robust image classification against adversarial attacks using elastic similarity measures between edge count sequences. *Neural Networks*, 128:61 – 72, 2020.

[PAK+13] Kanghee Park, Amna Ali, Dokyoon Kim, Yeolwoo An, Minkoo Kim, and Hyunjung Shin. Robust predictive model for evaluating breast cancer survivability. *Engineering Applications of Artificial Intelligence*, 26(9):2194 – 2205, 2013.

[Pea96] Karl Pearson. Vii. mathematical contributions to the theory of evolution.—iii. regression, heredity, and panmixia. *Philosophical Transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character*, (187):253–318, 1896.

[ped18] How a self-driving uber killed a pedestrian in arizona. *The New York Times*, Mar 2018.

[PSK09] Olli Post, Jari Seppälä, and Hannu Koivisto. The performance of opc-ua security model at field device level. In *ICINCO-RA*, pages 337–341, 2009.

[Qin12] S Joe Qin. Survey on data-driven industrial process monitoring and diagnosis. *Annual reviews in control*, 36(2):220–234, 2012.

[RJ17] Paul Didier Rajive Joshi, Stephen Mellor. The industrial internet of things volume g5: Connectivity framework. Technical report, Industrial Internet Consortium, 2017.

[RKK+17] Jérémy Robert, Sylvain Kubler, Niklas Kolbe, Alessandro Cerioni, Emmanuel Gastaud, and Kary Främling. Open iot ecosystem for enhanced interoperability in smart cities example of métropole de lyon. *Sensors*, 17(12):2849, 2017.

[RKLTF16] Jeremy Robert, Sylvain Kubler, Yves Le Traon, and Kary Främling. O-mi/o-df standards as interoperability enablers for industrial internet: A performance analysis. In *Industrial Electronics Society, IECON 2016-42nd Annual Conference of the IEEE*, pages 4908–4915. IEEE, 2016.

[Sau10] Thilo Sauter. The three generations of field-level networks - evolution and compatibility issues. *IEEE Transactions on Industrial Electronics*, 57(11), November 2010.

[SDG+21] Thibault Simonetto, Salijona Dyrmishi, Salah Ghamizi, Maxime Cordy, and Yves Le Traon. A unified framework for adversarial attack and defense in constrained feature space. *arXiv preprint arXiv:2112.01156*, 2021.

[SKK+15] Jan Schlechtendahl, Matthias Keinert, Felix Kretschmer, Armin Lechler, and Alexander Verl. Making existing production systems industry 4.0-ready. *Production Engineering*, 9(1):143–148, 2015.

[SL11] Thilo Sauter and Maksim Lobashov. How to access factory floor information using internet technologies and gateways. *IEEE Transactions on Industrial Informatics*, 7(4):699–712, 2011.

[SL17] Patrick Schäfer and Ulf Leser. Fast and accurate time series classification with weasel. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 637–646, 2017.

[SLS+18] Sebastian Schelter, Dustin Lange, Philipp Schmidt, Meltem Celikel, Felix Biessmann, and Andreas Grafberger. Automating large-scale data quality verification. *Proceedings of the VLDB Endowment*, 11(12):1781–1794, 2018.

[Sta14] OASIS Standard. Mqtt version 3.1. 1. http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.pdf, 2014.

[TMH17] Pattreeya Tanisaro, Florian Mahner, and Gunther Heidemann. Quasi view-independent human motion recognition in subspaces. In *Proceedings of the 9th International Conference on Machine Learning and Computing*, ICMLC 2017, pages 278–283, New York, NY, USA, 2017. Association for Computing Machinery.

[TSE+18] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*, 2018.

[Tur07] Charles D Turnitsa. Applying the levels of conceptual interoperability model in support of integratability, interoperability, and composability for system-of-systems engineering. *Journal of Systemics, Cybernetics and Informatics*, 2007.

[VHH16] Birgit Vogel-Heuser and Dieter Hess. Guest editorial industry 4.0–prerequisites and visions. *IEEE Transactions on Automation Science and Engineering*, 13(2):411–413, 2016.

[VT14] Anitha Varghese and Deepaknath Tandur. Wireless requirements and challenges in industry 4.0. In *Contemporary Computing and Informatics (IC3I), 2014 International Conference on*, pages 634–638. IEEE, 2014.

[WLS+19] Dorina Weichert, Patrick Link, Anke Stoll, Stefan Rüping, Steffen Ihlenfeldt, and Stefan Wrobel. A review of machine learning for the optimization of production processes. *The International Journal of Advanced Manufacturing Technology*, 104(5-8):1889–1902, 2019.

[WLW+16] Shuihua Wang, Zeyuan Lu, Ling Wei, Genlin Ji, and Jiquan Yang. Fitness-scaling adaptive genetic algorithm with local search for solving the multiple depot vehicle routing problem. *Simulation*, 92(7):601–616, 2016.

[WMZ+18] Jinjiang Wang, Yulin Ma, Laibin Zhang, Robert X Gao, and Dazhong Wu. Deep learning for smart manufacturing: Methods and applications. *Journal of Manufacturing Systems*, 48:144–156, 2018.

[WS96] Richard Y Wang and Diane M Strong. Beyond accuracy: What data quality means to data consumers. *Journal of management information systems*, 12(4):5–33, 1996.

[YPT19] Haimin Yang, Zhisong Pan, and Qing Tao. Online learning for time series prediction of ar model with missing data. *Neural Processing Letters*, 50(3):2247–2263, Feb 2019.

[YW06] Qiang Yang and Xindong Wu. 10 challenging problems in data mining research. *International Journal of Information Technology & Decision Making*, 5(04):597–604, 2006.

[YWS+19] Jining Yan, Lizhe Wang, Weijing Song, Yunliang Chen, Xiaodao Chen, and Ze Deng. A time-series classification approach based on change detection for rapid land cover mapping. *ISPRS Journal of Photogrammetry and Remote Sensing*, 158:249 – 262, 2019.

[ZH13] Tingting Zhai and Zhenfeng He. Instance selection for time series classification based on immune binary particle swarm optimization. *Knowledge-Based Systems*, 49:106 – 115, 2013.

[ZHML20] Jie M Zhang, Mark Harman, Lei Ma, and Yang Liu. Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering*, 2020.