UNISINOS

Programa de Pós-Graduação em

Computação Aplicada

Mestrado/Doutorado Acadêmico

Jovani Dalzochio

ELFpm:
An Ensemble-Based Learning Framework for Predictive Maintenance in Industry 4.0

São Leopoldo, 2020

Jovani Dalzochio

ELFPM: **An Ensemble-Based Learning Framework for Predictive Maintenance in Industry 4.0**

Dissertação apresentada como requisito parcial para a obtenção do título de Mestre pelo Programa de Pós-Graduação em Computação Aplicada da Universidade do Vale do Rio dos Sinos — UNISINOS

Advisor:
Prof. Dr. Rafel Kunst

Co-advisor:
Prof. Dr. Jorge Barbosa

São Leopoldo
2020

D153e Dalzochio, Jovani.

ELFpm : an ensemble-based learning framework for predictive maintenance in industry 4.0 / Jovani Dalzochio. – 2020.

91 f. : il. ; 30 cm.

Dissertação (mestrado) – Universidade do Vale do Rio dos Sinos, Programa de Pós-Graduação em Computação Aplicada, 2020.

"Advisor: Prof. Dr. Rafael Kunst".

1. Computação aplicada. 2. Indústria 4.0. 3. Ontologia. 4. Aprendizado do computador. 5. Manutenção preditiva. I. Título.

CDU 004

Dados Internacionais de Catalogação na Publicação (CIP)
(Bibliotecário: Flávio Nunes – CRB 10/1298)

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior Brasil (CAPES) -Código de Financiamento 001 /"This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior -Brasil (CAPES) -Finance Code 001

ACKNOWLEDGEMENTS

First of all, I have to thank my mother Cecília, who despite leaving us very early, taught me and my brothers the importance of studies and the value of knowledge. I have to quote here my sister Joelma and my brother Gerson, who through the certainty that they are with me for everything I need, give me the security to keep going forward.

Furthermore, this dissertation would not have happened without the great help of my advisor Rafael Kunst, who through long weekly talks gave me valuable help, advice, and words that in one way or another motivated me to continue despite all the difficulties. I thank Professor Jorge Luis Victória Barbosa and Professor Gabriel de Oliveira Ramos for their time and valuable suggestions. Finally, I would like to thank CAPES for the financial support through the Support Program for the Graduate Studies of Private Education Institutions (PROSUP)

ABSTRACT

The topic of predictive maintenance has great relevance in the search for the rationalization and efficiency of the industrial plants in the context of Industry 4.0. Monitoring equipment parameters and identifying behavior changes that identify a future failure allows for anticipation of maintenance while avoiding unnecessary preventive maintenance. There are numerous works in the literature that work towards the prediction of maintenance of various equipment. However, the same equipment has different behavior depending on the conditions of use or the operating environment, making a tool capable of being trained for new environments is necessary. This work describes the methodology of creating a framework that can be configured to work on predicting equipment failures, that is, regardless of location or condition of use. For this, starting from the initial configuration of the framework, the use of an ontology is applied in the choice of the best prediction technique for each established condition of the initial parameterization.

Keywords: Industry 4.0. Ontology. Machine learning. Predictive Maintenance.

LIST OF FIGURES

1 5C architecture for CPPS application (Adapted from (LEE; BAGHERI; KAO, 2014)). 19
2 Ontology design and evaluation procedure (Adapted from (GRUNINGER; FOX, 1995)) 24
3 Ontology build steps of Uschold e King (1995) method (Adapted from (GÓMEZ-PÉREZ, 1999)) 25
4 Ontology build processes of Bernaras, Laresgoiti e Corera (1996) approach (Adapted from (GÓMEZ-PÉREZ, 1999)) 26
5 The memory cell of a LSTM, adapted from (WANG; RAJ, 2017) 30
6 Structure of bagging. 31
7 System of interconnected neurons in an MLP (adapted from (GARDNER; DORLING, 1998)). 32
8 Three-step methodology of Box-Jenkins method to create an ARIMA model. 34
9 Filter process of selected articles by database 43
10 A Taxonomy to Classify Challenges and Open Issues in PdM 48
11 Taxonomy of Machine Learning Techniques 51
12 Taxonomy of Ontologies Applications in Predictive Maintenance 53
13 Framework overview. 55
14 Ontology Agent. 57
15 Sequence diagram of the ontology creation. 59
16 Sequence diagram of the data collection, analysis, and notification. 60
17 Classes of the ontology. 61
18 Vibration trend over the time 66
19 Performance of best RF model for training (red) and test (green). 67
20 Performance of best LSTM model for training (red) and test (green). 68
21 Performance of best MLP model for training (red) and test (green). 69
22 Autocorrelation of the dataset. 70
23 Performance of best ARIMA model for test (green). 70
24 Comparison of the standard derivation. 71
25 Model error selected by ε-Greedy Algorithm based on the smallest error . . 73
26 Model time selected by ε-Greedy Algorithm based on the smallest error . . . 74
27 Model error selected by ε-Greedy Algorithm based on the smallest time . . . 75
28 Model time selected by ε-Greedy Algorithm based on the smallest time . . . 75
29 Model error selected by TOPSIS 76
30 Model time selected by TOPSIS 77
31 Average and standard deviation of the best by error 78
32 Average and standard deviation of the best by times 79
33 Mean average of the last 400 points. 81
34 Threshold value in the dataset. 81
35 Threshold values in the last 400 points collected 82

LIST OF TABLES

1 Complexity analysis of attribute normalization and weighing (Adapted from (HAMDANI; WARDOYO, 2016)) 36
2 Complexity analysis of PIS and NIS (Adapted from (HAMDANI; WARDOYO, 2016)) 37
3 Calculation of the distance of values compared to NIS and PIS (Adapted from (HAMDANI; WARDOYO, 2016)) 37
4 Complexity analysis of ranking. (Adapted from (HAMDANI; WARDOYO, 2016)) 37
5 Complexity analysis of ranking (Adapted from (HAMDANI; WARDOYO, 2016)) 38
6 Answers and Grades 46
7 Quality Assessment Scores 46
8 Ontology query return 64
9 Parameters comparison using RF 67
10 Parameters comparison using LSTM 68
11 Parameters comparison using MLP 69
12 Best value for RMSE in each algorithm 71
13 Confidence value for the proposed indexes 77

CONTENTS

1 INTRODUCTION **15**
1.1 Motivation 15
1.2 Problem Definition 15
1.3 Objectives 16
1.4 Methodology 16

2 BACKGROUND **17**
2.1 Industry 4.0 17
2.1.1 Cyber Physical Systems 18
2.1.2 Predictive Maintenance 19
2.2 Ontology 21
2.2.1 Engineering for Building Ontologies 23
2.2.2 Grüninger and Fox methodology 23
2.2.3 Uschold and King method 24
2.2.4 The KACTUS approach 26
2.2.5 METHONTOLOGY 26
2.2.6 Ontology Development 101 27
2.3 Machine Learning 28
2.3.1 Long-Short Term Memory 28
2.3.2 Random Forest 31
2.3.3 Multilayer Perceptron 32
2.3.4 Autoregressive integrated moving average 33
2.4 Technique for order preference by similarity to ideal solution (TOPSIS) 33
2.4.1 Calculate Normalized Ratings 34
2.4.2 Calculate Weighted Normalized Ratings 35
2.4.3 Determine ideal and negative-ideal solutions 35
2.4.4 Calculate the separation measure 35
2.4.5 Calculate the relative closeness to the ideal solution 36
2.4.6 Rank the preference order 36
2.4.7 Computational Cost 36
2.5 ε-Greedy 37
2.5.1 Computational Cost 38

3 RELATED WORK **39**
3.1 Research Methodology 39
3.1.1 Research Questions 39
3.1.2 Search Process 40
3.1.3 Papers Selection Process 40
3.1.4 Quality Assessment 41
3.2 Search Results 42
3.2.1 Exclusion of papers from the initial corpora 42
3.2.2 Performing the Quality Assessment to Select Relevant Papers 45
3.3 Answer to the Research Questions and Discussion 47
3.3.1 What are the challenges and open questions regarding machine learning and reasoning for predictive maintenance in Industry 4.0? 48
3.3.2 What machine learning techniques are being used? 50
3.3.3 What are the contexts in the use of ontology in predictive maintenance? 52

3.4 Systematic literature review conclusions 53

4 ELFPM FRAMEWORK **55**
4.1 Architecture of ELFpm 55
4.1.1 Physical Objects Layer 56
4.1.2 Provisioning Layer 56
4.1.3 Application Layer of the framework 57
4.2 Ontology Modeling 59
4.2.1 Motivating scenarios 60
4.2.2 Elaborate informal competency questions 60
4.2.3 Specify the terminology using first order logic 61
4.2.4 Formal axioms 62
4.2.5 Formal competence question 63
4.2.6 Completeness theorem 64

5 RESULTS **65**
5.1 Methodology 65
5.2 Machine learning model training 66
5.2.1 Random Forest 66
5.2.2 LSTM 67
5.2.3 Multilayer Perceptron 68
5.2.4 ARIMA 69
5.2.5 Machine learning model training results 70
5.2.6 ε-Greedy Algorithm Results 72
5.2.7 ELFIndex Results 76
5.2.8 Computational Cost 78
5.2.9 Selection of the best model 79
5.2.10 Threshold definition 80

6 CONCLUSION AND FUTURE WORK **83**

REFERENCES **85**

1 INTRODUCTION

1.1 Motivation

Our society is generating more and more data. This data is being generated everywhere, at all times and by any device. With the use of data processing and analysis tools new opportunities are opened in both academia and industry (YI et al., 2014). There is a highly competitive scenario in the industry and many companies are failing to take advantage of the large amount of data that is constantly being generated by the industry.

Thinking about it, Germany is leading the so-called fourth industry revolution (Industry 4.0). In the context of industry 4.0 it is possible to integrate software with artificial intelligence to perform predictions of machine and component degradation (LEE; KAO; YANG, 2014).

By performing predictions of failures in equipment and components it is possible to increase the performance of the machines, minimizing downtime and reducing their inefficiency (YAN et al., 2017). In quantitative terms, a study done in the Swiss industry published in 2016 shows that there is an estimate that 12.7% of a machine's time is in a standstill, with only 37% of that time being planned. These stops cause a 22.3% impact of the manufacturing cost (Salonen e Tabikh (2016).

Although it is a topic of great interest and a high priority for many industries, growth occurs as a matter of course, sectors such as biochemical and biological have the opportunity to seek new paths, while other industries such as semiconductors have made better use of the new ones features offered by industry 4.0 like predictive maintenance (PdM)(MOYNE; ISKANDAR, 2017).

1.2 Problem Definition

Since several industries and sectors are working on the prediction of failure, it is very common for identical machines to be exposed to completely different environments to perform various types of tasks. However, the prediction of failure methods are limited according to the machine and its work process (LEE; KAO; YANG, 2014). For the most part, the proposed works in this area addresses situations and scenarios with machines performing specific tasks, which from this context seeks to find the best tool to enhance the prediction for maintenance (ZHANG; YANG; WANG, 2019).

Is it possible to develop a framework for predictive maintenance that is capable of defining the best possible prediction algorithm, regardless of how many are available, for making predictions?

1.3 Objectives

This work propose *ELFpm*, a framework that through the use of ontologies has the ability to select the machine learning algorithms with the the smallest error and best training and prediction times for the prediction of failure, regardless of which machine or process is being monitored.

To achieve this objective, this work will focus on the following specific objectives.

- Propose ELFpm, a framework for PdM;
- Build an ontology to represent the working context of ELFpm;
- Implement the machine learning algorithms that are used by ELFpm;
- Implement choice optimization techniques
- Implement a method for setting threshold values
- Elaborate scenarios and evaluate the framework.

1.4 Methodology

The first step to reach the proposed objectives is to make a bibliographic review of the themes related to PdM. This study will serve as a support to understand how works that have been already developed deal with prediction of failures, to identify which techniques are employed and understand their use.

The second step is to carry out a systematic review of the literature that aims to identify what is being studied within the area of predictive maintenance, which frameworks are being proposed in the context of industry 4.0, which machine learning algorithms are being used, how ontologies are being used and what the challenges are being faced.

The third step will be to describe the framework *ELFpm*, explaining which are the layers that make up the proposed framework, and within each layer what are the components and their functions. In addition, it will be presented how the framework will help to achieve the goal in a satisfactory way.

After the presentation of ELFpm, the fourth step is the presentation of the results obtained by the framework, both with regard to the use of ontology, as well as the results obtained by the machine learning algorithms until the choice of the best model.

At the end, the conclusions made from the development and testing of ELFpm are presented, ending with suggestions for future work.

2 BACKGROUND

In this section we will focus on the main concepts related to the work developed here, starting with an introduction to Industry 4.0, the related Cyber Physical Systems and a conceptualization on Predictive Maintenance. After the presentation of these initial concepts, an introduction to ontologies will be presented, starting from its conception, construction, and use of ontologies through the use of some methodologies.

Afterwards, the machine learning algorithms used in the implementation of this work will be demonstrated, with an explanation of the basic concepts about the Long short-term memory model of an Recurrent neural network, the Random Forest model, the Multi Layer Perceptron models and the ARIMA method. Finally, at the end of this section, the operation of the TOPSIS and ε-Greedy algorithms, applied to choose the best learning model, will be explained.

2.1 Industry 4.0

The term Industry 4.0 is used to represent the fourth industrial revolution that is occurring in recent years. The term was first proposed in 2011 at the Hannover fair in Germany and was later announced in 2013 as a German government strategy to play a leading role in the industrial sector (XU; XU; LI, 2018).

Lasi et al. (2014) described Industry 4.0 as a project that has two forces driving its development. The first is an application-pull, which is driven by social, economic and political changes. In this scenario, there is a strong demand for innovations, and the period between each innovation is shorter. The demand presented by buyers in the market requires individualization and greater flexibility in the development of products, which has the effect of decentralizing the chain of hierarchies in the company. All of these changes must be implemented always with a commitment to efficiency in mind, with the objective of achieving not only economic, but also ecological gains.

The second force is a technology-push, where some words like Web 2.0, Apps, 3D-printers end up having great prominence. With the use of innovative technologies, an increase in the automation of manufacturing processes is observed, which was made possible by the large-scale use of sensors for control and analysis of components through digitization and the use of communication networks. The miniaturization of sensors and computers allowing new fields of applications to develop.

The term Industry 4.0 refers to a list of concepts (LASI et al., 2014), the main ones being listed below:

- Intelligent factories are equipped with sensors, applying technologies in a ubiquitous way besides autonomously controlled;
- Junction of the physical with the virtual, being no longer possible distinguish. In Cyber-

physical Systems environments, the condition of a real-world object requires the processing of digital analysis;

- Manufacturing is increasingly decentralized and adapted to human needs;
- The development of products and services will be individualized. In this context, open approaches to innovation and product intelligence, as well as product memory, are of the utmost importance;
- Social and environmental responsibility in the design of manufactures are fundamental for the success of products.

Along with the advancement of industry 4.0 new technologies and concepts have emerged and gained relevance, such as the concept of Cyber-Physical Systems (CPS) and PdM.

2.1.1 Cyber Physical Systems

A CPS consists of a controller that controls sensors and actuators for data collection and interaction with the real world. Through a communication interface data exchanges are made with other local systems or in the cloud. This data exchange is the most important part of a CPS system, when this exchange takes place over the internet, the CPS can be called the "Internet of things" (JAZDI, 2014).

So, cyber-physical system models are usually composed of three parts, a physical subsystems with computing and networking. An integration of these three parts to monitor and control physical processes, in some occasions with feedbacks that affect the physical process and vice versa. CPS are an intersection between the physical world and the cyber (LEE; SESHIA, 2016). The concept of interaction between the physical world and the cyber is a foundation in the implementation of intelligent plants in the context of industry.

CPS applied to industry are called Cyber-Physical Production Systems (CPPS). CPPS offers a great advantage in that it is a flexible concept capable of adapting to new plant topology or new products, as well as supporting communication between products, machines and humans via interfaces (MONOSTORI, 2014).

In general, a CPPS has two main components, connectivity and data collection in real time and analysis and management of this data. To apply these two concepts, architecture and guides are created to assist companies, such as the architecture proposed by Lee, Bagheri e Kao (2014). In the architecture called 5C, a structure of 5 levels is presented, starting from the most basic and fundamental part that is the collection of the data until the aid in making descriptions. This architecture is presented in Figure 1 where each level is briefly presented.

Architectures with 5C assist in the implementation of CPS, and consequently enables a more resilient and intelligent manufacturing.

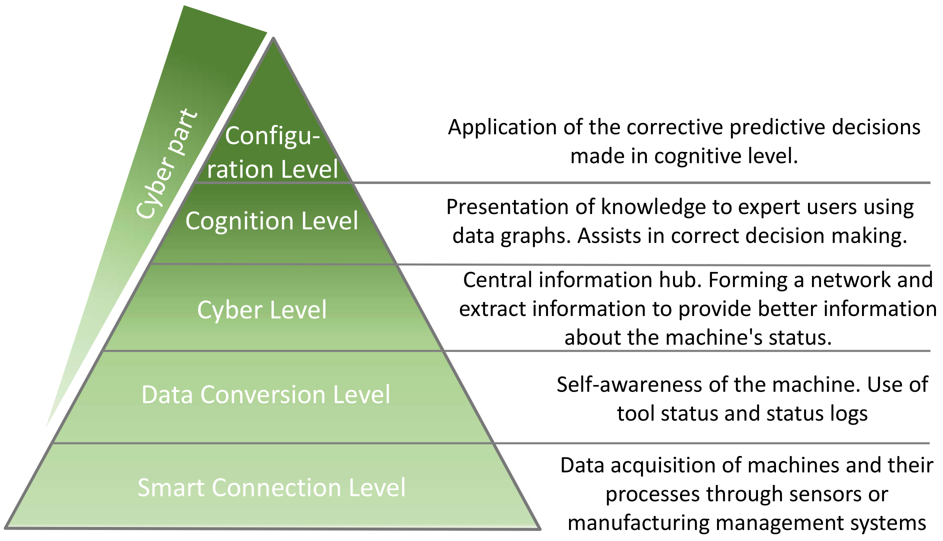

Cyber part
Configu-
ration Level
Cognition Level
Cyber Level
Data Conversion Level
Smart Connection Level
Application of the corrective predictive decisions made in cognitive level.
Presentation of knowledge to expert users using data graphs. Assists in correct decision making.
Central information hub. Forming a network and extract information to provide better information about the machine's status.
Self-awareness of the machine. Use of tool status and status logs
Data acquisition of machines and their processes through sensors or manufacturing management systems

Figure 1: 5C architecture for CPPS application (Adapted from (LEE; BAGHERI; KAO, 2014)).

2.1.2 Predictive Maintenance

Predictive Maintenance (PdM), also known as “online monitoring,” “condition-based maintenance,” or “risk-based maintenance", is not a new concept in Industry 4.0. In general, PdM consists of collecting signals from a specific equipment and detecting patterns that may indicate the presence of a possible failure and acting actively with the purpose of preventing that a significant failure really occur. Initially, the signs that a possible failure could occur were perceived through the observation of specialists, with visual inspection being the oldest and most common form of PdM. Although visual inspection remains a widely used method, this symptom detection process has evolved into automated methods, based on the use of sensors for data collection, which are then analyzed through the use of pattern recognition techniques, thus avoiding exchanging unnecessary parts and increasing process safety and efficiency (HASHEMIAN, 2010).

PdM is usually employed to achieve two main objectives: failure prevention and improvement of process efficiency. Fault prevention is the main objective for the use of PdM, since 99% of the failures in machines are preceded by some sign of discrepancy in the operation of the same. From the standpoint of improving the process efficiency, PdM helps increase process safety, increase product quality, improve process reliability, increase resource availability, reduce parts exchange costs and manual labor, reduce waste of raw materials and consumables and reduce the energy consumption of the machines Selcuk (2017).

In order to implement a PdM program, it is necessary initially to use monitoring techniques, and in addition, the existence of a more complete program also requires the use of diagnostic techniques. For (MOBLEY, 2002), these techniques include:

- Vibration Monitoring: It is the main tool for PdM in electromechanical equipment, being widely used. Despite this, some limitations are associated with its use, these limitations can be seen as disadvantages or advantages, such as simplified data acquisition and analysis, but the difficulty of dealing with machines that have low operating speed, producing low vibration.

- Thermography:An analysis is made of the infrared energy emitted by the equipment, detecting changes in temperature. It can be done by using infrared thermometers, line scanners, or infrared imaging. Its use implies several challenges, such as external heat emitters influencing the temperature registered in the monitored equipment, possibly paintings that alter the signature of the collected signal, implying a loss of accuracy in the trained prediction models.

- Tribology: It is the term used to refer to operating dynamics of the bearing-lubrication-rotor support structure of machinery where the lubricating oil is analyzed, assessing its condition in the lubrication of mechanical and electrical components and determining, among other things, the exchange time, in addition to particle wear analysis. There are some challenges in the application of tribology in a PdM program, with equipment costs, acquiring accurate oil samples, and interpretation of data being the three main.

- Visual Inspections: As already mentioned, it is the first method for PdM used in the industry. Despite being used since the beginning of the industrial revolution, a complete current PdM program must include visual inspections as PdM tools.

- Ultrasonic: It works similarly to the use of vibration, changing only the monitored frequency band. However, the use of ultrasonic should be restricted to the detection of abnormally high ambient noise levels and leaks.

- Other Techniques: Here come electrical tests such as resistance and impedance, which together with vibration techniques prevent premature failure of electric motors.

Hashemian (2010) created a classification for these techniques, dividing it into three distinct categories. The first category, called “Process Sensors” and characterized as passive, concerns the use of data captured by the machine’s sensors, data that Mobley (2002) previously commented, such as temperature, pressure, and level. The second category, also passive and called “Test Sensors, Including Wireless”, uses data from test and diagnostic sensors such as vibration, electrical data, in addition to the use of Wireless sensors that can provide information about the plant’s environment. Finally, the third category, called “Test signal”, is characterized as active and measures actively using methods such as Loop Current Step Response (LCSR). The LCSR technique can be applied to determine the water level in a pipe, and is even applied in the accident at the Three MileIsland nuclear plant to help determine the water level in the primary refrigerant pipes.

The application of these techniques in conjunction with software to assist in data capture, model training and failure prediction aims to prevent failures from occurring unexpectedly, reducing costs, increasing the safety of processes and increasing the level of competitiveness of the company in market.

2.2 Ontology

Ontology is a term that has philosophical origins, its purpose is to describe the entities and types of entities that exist through the study of the structures of the world. For example an entity of the living things of the world can have sub classes to distinguish animal and plants, which also have their own sub classes. In the field of computer science, ontologies are used to formalize knowledge of a specific domain, with its concepts, entities, vocabulary and relationships between them (HORROCKS, 2008).

The use of an ontology helps to shared information of a domain through the definition of a common vocabulary. (NOY; MCGUINNESS et al., 2001) lists some reasons to justify the development of an ontology such as the sharing between people and software of a common knowledge about the information structure; allows reusing the knowledge of the domain already produced in other ontologies, besides making integration's that allow to expand the knowledge about a domain of interest; makes explicit assumptions about a domain by allowing them to be easily changed when compared to assumptions made in programming language; separate knowledge from the domain of operational knowledge; formally analyze domain knowledge.

For the construction of a formal model of an ontology, it is necessary that some characteristics are defined. In a first level of distinction is defined by the model $O = (S, A)$, where the *signature* **S** comprises a set of conceptual entities used to represent knowledge and **A** the sets of *axioms* expressed in a specific ontology language or a formalism of knowledge representation. At a second level of distinction the signature S can comprise a set of entities divided into *classes* **C**, *instances* **I**, and *properties* **P**. This distinction allows class associations with their instances or the interrelationship of instances via properties. In a third level of distinction the signature *classes* **C** are divided into a set of *concepts* **C** and *datatypes* **D**, *instances* **I** in *individuals* **I**, and *data values* **V** and finally *properties* **P** in *relations* **R** and *attributes* **T**. At this point it is possible to distinction between domain abstract objects and concrete data values.

In guiding the construction of an ontology whose purpose is to share knowledge between systems, some criteria must be observed (GRUBER, 1995):

- Clarity: The terms defined should be objective and clear in the transmission of their meaning. Documentation should be done in natural language. These terms must be independent of social context and when possible stated in logical axes.

- Coherence: An ontology must be coherent, its inferences must be consistent with the definitions. Coherence should also apply to concepts defined in a formal way, such as those described in natural language documentation and examples. If an inference of an axioms contradicts the definition or example given informally, then the ontology is incoherent.

- Extendibility: When designing an ontology the use of shared vocabulary must be anticipated. You should also be able to define new terms for special uses based on your existing vocabulary in a way that does not require reviewing existing definitions.

- Minimal encoding bia: Conceptualization should not depend on a specific coding, because knowledge sharing agents can be implemented in different representation systems and representation styles.

- Minimal ontological commitment: An ontology should make as few assertions as possible about the entity being modeled, allowing the parties involved to have the freedom to specialize ontology instances as needed.

As seen before, an ontology provides a common vocabulary that can be shared across systems and assists in the formalization of knowledge. To represent this formalisation of knowledge the ontology provides five types of components (GÓMEZ-PÉREZ, 1999), being:

- Concepts: It can be concrete or abstract, real or fictional, an action, task or process. Anything that someone said about something.

- Relations: They represent interactions between concepts of the domain. As an example of a binary relationship, subclass-of.

- Functions: Are special relations where the n-th element of the relation is unique to the preceding n-1 elements

- Axioms: Model sentences that must always be true

- Instances: They are used to represent the elements of an ontology

For the construction of the ontologies is necessary the use of some type of language. There are several of them in the literature (GRIMM et al., 2011) like the Resource Description Framework (RDF), that arise as an effort to standardize metadata, RDF and RDF Schema (RDFS) emerged from a World Wide WebConsortium (W3C) initiative. RDF is a standard that has become popular and today is widely used to encode basic metadata and ontologies on the Web. RDF and RDFS are languages that allow the representation of concepts, taxonomies of concepts and binary relations.

After the development of RDF, OWL was standardized by W3C. OWL comes as a language that allows the development of more expressive ontologies for use on the Web. Several OWL variants have been created with OWL Full, OWL DL as well as a second version, OWL 2.

In addition to RDF's built-in functions OWL offers the construction of complex classes from simpler expressions through logical expressions, rich axiomatization, including the exclusion of classes. This was the language chose to be used in this work.

To assist in the development and construction of ontologies it was necessary to develop new tools to assist in this task. The tool used in the development of this work was the Protégé, currently the most used tool in building and maintaining ontologies. There exists a variety of Protégé structures. The desktop system (Protégé 5) was used to develop the ontology of this work. It supports many advanced features to enable the construction and management of OWL ontologies. There is also a simplified version on the Web (WebProtégé). Currently the Web version is more used than the full Desktop version (MUSEN et al., 2015).

2.2.1 Engineering for Building Ontologies

As seen earlier, modeling an ontology involves following several steps, taking into account aspects such as the domain of ontology or what questions it should be able to answer. To guide the construction of these ontologies, the ontology construction methodologies were created. The following is a brief overview of the most prominent methodologies, in addition to that used in this paper.

2.2.2 Grüninger and Fox methodology

The approach proposed by (GRUNINGER; FOX, 1995) to building ontologies has started from specific problems coming from industry partners and has as an initial effort to support reasoning in these industry environments. It begins by elaborating questions that ontology must be able to answer, moving on to definitions of ontology terminology (objects, attributes, and relationships) and specifying the constraints of terminology using first order logic. Finally, test your competency by providing completeness theorems for competency questions. The proposed mechanism for guiding the construction of ontology and subsequent evaluation (Figure. 2) is divided into 6 activities, as follows:

- Motivating Scenarios: To assist in understanding the proposal of ontology and its application, there are motivational scenarios. These scenarios allow the elaboration of ideas for possible solutions that will be transformed into objects and relations in a future ontology.

- Informal Competency Questions: These are the questions that the ontology should answer, serving as a form of initial assessment of the ontology. At this point it can be verified that the defined questions can be answered with existing ontologies.

- Specification in First-Order Logic - Terminology: Definition of ontology terminology, formalizing the terms of objects, property of objects, relationships between objects by

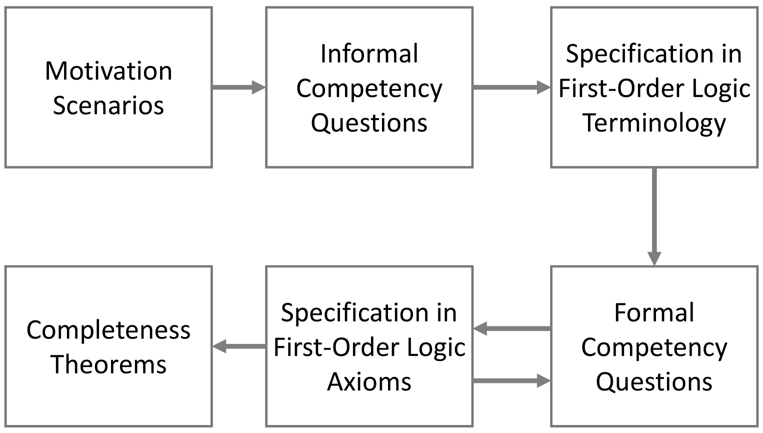

Motivation Scenarios
Informal Competency Questions
Specification in First-Order Logic Terminology
Formal Competency Questions
Specification in First-Order Logic Axioms
Completeness Theorems

Figure 2: Ontology design and evaluation procedure (Adapted from (GRUNINGER; FOX, 1995))

applying first-order logic. For new ontologies, informal competence questions will serve as the basis for proposing the terms needed to answer each question.

- Formal Competency Questions: After informal competence questions and ontology terminology are established, formal competence questions are proposed and written in a formal representation using first-order logic. Formal competence questions add constraints, and on those constraints will be included the axioms. It is important that any new ontology or even extensions of existing ontologies present such questions so that the ontology can be evaluated and stated if it is adequate.

- Specification in First-Order Logic - Axioms: As a way of specifying definitions of terms in ontology and constraints in their interpretation, providing a semantic definition or meaning for those terms. The process of defining the axioms is guided by the questions of formal competence, with the proposed axioms capable of characterizing solutions to the questions of formal competence.

- Completeness Theorems: After the formal competence questions have been established it is necessary to define the conditions under each of the questions are complete. Completeness theorems also provide a means of determining the extensibility of an ontology through the proof of theorem that each axiom plays. Every extension of ontology must preserve the completeness of theorem.

2.2.3 Uschold and King method

The first method proposed for the construction of ontologies (GÓMEZ-PÉREZ, 1999), the methodology proposed by (USCHOLD; KING, 1995) presents a small number of stages that he believed will be necessary in any future methodology for the construction of ontologies. As seem in Figure 3 the following steps must be followed to build an ontology:

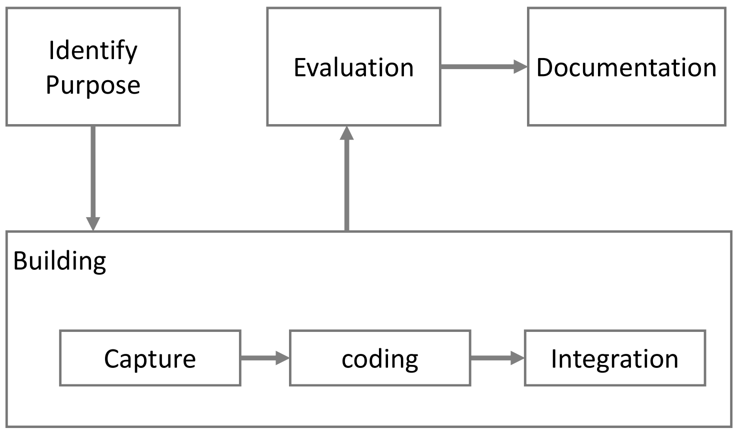


Figure 3: Ontology build steps of Uschold e King (1995) method (Adapted from (GÓMEZ-PÉREZ, 1999))

- Identify Purpose: In this stage we seek to understand why the ontology is being build and what is the purpose of its use. It is also at this moment that the potential users of ontology are identify and characterized.

- Building the Ontology: This step is divided into three activities. The first activity is *Capture*, where the key concepts and their relationships are identified within the domain of interest. These concepts and relationships must be defined textually. It is in the *Capture* activity that terms (concepts and relationships) are defined. The second activity is *Coding*, where the formal knowledge acquired in the previous activity is explained in a formal language. This involves committing to some meta-ontology, choosing a representation language and creating the code. Finally, the third activity is *Integrating existing ontologies*, which refers to how and when to use existing ontologies.

- Evaluation: In this step is used the definition of Gómez-Pérez that affirm: “to make a technical judgment of the ontologies, their associated software environment, and documentation with respect to a frame of reference... The frame of reference may be requirement specifications, competency questions, and/or the real world".

- Documentation: It recommends the use of guidelines for the documentation of the ontology of your type and purpose. As an example is to find similar definitions together or create naming conventions, such as: use upper or lower case letters to name the terms or write the terms of the uppercase representation ontology.

2.2.4 The KACTUS approach

The approach to the development of ontologies propose by (BERNARAS; LARESGOITI; CORERA, 1996) has the condition that the development of an application be considered complete, and whenever an application is construed, the ontology that represents his knowledge must be improved. Since the development of an ontology is linked to the development of an application, for the construction of the application the following processes must be followed as see in Figure 4:

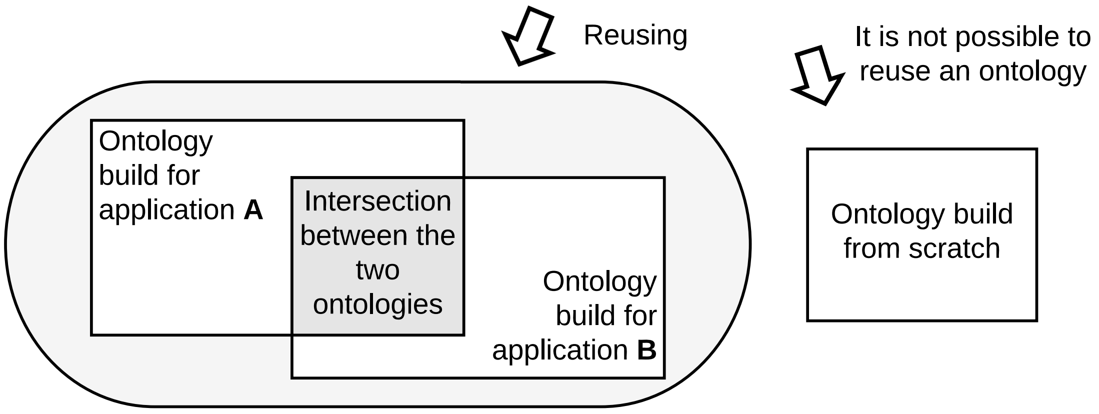

Reusing
It is not possible to reuse an ontology
Ontology build for application **A**
Intersection between the two ontologies
Ontology build for application **B**
Ontology build from scratch

Figure 4: Ontology build processes of Bernaras, Laresgoiti e Corera (1996) approach (Adapted from (GÓMEZ-PÉREZ, 1999))

- Specification of the application: At this stage a list of terms and tasks should be provided, helping to visualize the context and the components that the application will model.
- Preliminary design based on relevant top-level ontological categories: The terms and tasks obtained in the previous process will be used as input to visualize the model globally with its concepts, relationships, and attributes for example. This is done using top-level ontological categories. As can be seen in Figure X, to obtain this visualization can be used existing ontologies, redefined terms or extending the ontology to be used in a new application.
- Ontology refinement and structuring: In this process we seek to reach a definitive design following the principles of hierarchical organization modularization, ensuring that the modules are not very dependent on each other and are as coherent as possible, obtaining the highest possible level of homogeneity within each module.

2.2.5 METHONTOLOGY

The METHONTOLOGY (FERNáNDEZ-LóPEZ; GOMEZ-PEREZ; JURISTO, 1997) it is a methodology based on the knowledge acquired in the development of chemical ontology's.

The methodology consists of the following phases:

- Specification: Using informal, semi-informal or formal language, write a document with ontology specifications, such as ontology purpose, usage scenario, and end-users. The document must define the level of formality of implementation, the terms, characteristics and granularity of ontology. It is important to ensure that there are no irrelevant or duplicate terms and that they all make sense.

- Knowledge acquisition: There are several sources of knowledge that can be used, from book to handbooks or even other ontologies. In addition it is possible to interview experts in the ontology domain or brainstorming. This activities helped to formulate a glossary of potential terms.

- Conceptualization: It is the phase where knowledge will be structured. The glossary of terms (GT) should be built in this step, ensuring knowledge of the domain of ontology through concepts, terms, verbs and definite properties. All new term must be include in the GT.

- Integration: The reuse of definitions constructed in other ontologies should always be considered, this speeds up the construction of the new ontology. If there are ontologies that fit for reuse make sure the terms are the same, otherwise create new implementations of meta-ontology.

- Implementation: It is the coding of ontology in a formal language like Prolog or C ++. The development environments used must have some features that check for lexical or syntactic errors, detect redundancies or inconsistencies, allow additions or removals of definitions, among other characteristics.

- Evaluation: It consists of conducting a technical judgment of the ontology, software environments and the documentation created in each phase. The correctness of the ontology, software environments and documentation is verified during and between each phase and the correctness of the ontology, software environments and documentation is validated according to the system expected to be represented.

- Documentation: Documentation is an activity that must be performed throughout the development process of ontology. At the end of each phase, a resulting document should be generated.

2.2.6 Ontology Development 101

The Ontology Development 101 proposed by Noy, McGuinness et al. (2001) has 7 steps for developing an ontology which will be detailed below. These steps guide ontology modeling to reach something that works well, is intuitive, extendable, and maintainable is achieved. Importantly, there is no one "correct" way to build an ontology, the process of development is always interactive, and the concepts of ontology must be close to objects and their relationships in a domain of interest.

- Determine the domain and scope of the ontology: It is suggested as the first step in constructing inquiries as to what domain the ontology will cover, what its use will be, what questions the information of the ontology should answer and who will maintain it. These questions help in defining the domain of onotlogy and its scope.
- Consider reusing existing ontologies: It is possible to check if someone has done an ontology in the same context and analyze their reuse for a domain or task that suits us.
- Enumerate important terms in the ontology: Write a list of terms, their meanings, their property, their relationship, and what we mean by those terms.
- Define the classes and the class hierarchy: In this step it is possible to define the general concepts of the domain and then specialize these concepts in a top-down model or to perform a bottom-up approach, starting with the definition of more class-specific concepts and then grouping these concepts hierarchically until they reach more general levels. A combination of the two concepts is also possible, the choice depends on the personal view of the domain. Regardless of the choice of approach, in this step the list of previously defined terms will become ontology classes, which will be organized hierarchically.
- Define the properties of classes—slots: Previously defined classes need to gain properties, which can be classified as intrinsic, extrinsic, parts, and relationships with other individuals. Each property becomes a class slot and these can properties provide information that helped solve competency questions.
- Define the facets of the slots: A slot can describe features such as data type (string, number, etc.), domain and range, and what cardinality (how many values a slot can have)
- Create instances: In the last step, instances of individuals of the class with their informed property slots are created.

2.3 Machine Learning

2.3.1 Long-Short Term Memory

Recurrent neural networks are able to use information from nearby previous tasks as input to the neural network, but there are scenarios where relevant information is distant and will have its weight decreased. To solve this problem the Long-Short Term Memory (LSTM) was proposed, which is a type of RNN developed by (GERS; SCHMIDHUBER; CUMMINS, 2000). The LSTM has as its characteristic a gradient learning based. This characteristic allows to consider as input long-term information becoming able to deal satisfactorily with time series.

The LSTM is a cell composed of a neuron and gates of input, forgot and output. The memory of the network is the responsibility of the neuron, which can allow information to be added or not in the cell through these gates. A LSTM consists of seven components:

- Forgot gate: Decide what information will be kept or forgotten of the previous state.

$$f^t = \sigma(W_{fi} i^t) \tag{2.1}$$

- Input gate: decides which new information will enter the cell, has a layer with a tanh function that creates a vector of values for the new candidates.

$$g^t = \sigma(W_{gi} i^t) \tag{2.2}$$

- Output gate: decides if the internal state is passed out to the hidden state in the next step.

$$o^t = \sigma(W_{oi} i^t) \tag{2.3}$$

- Input data: LSTM input data
- Hidden state: used to determine what to forget, enter and exit in the next step

$$h^t = o^t \bigodot m^t \tag{2.4}$$

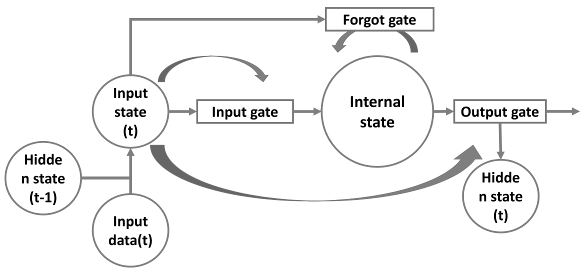
- Input state: combination of the hidden state and the current input

$$i^t = \sigma(W_{ix} x^t + W_{ih} h^{t-1}) \tag{2.5}$$

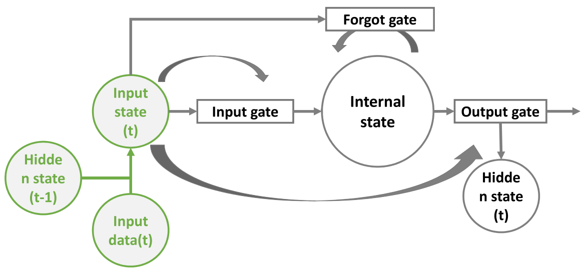
- Internal state: values with memory function

$$m^t = g^t \bigodot i^t + f^t m^{t-1} \tag{2.6}$$

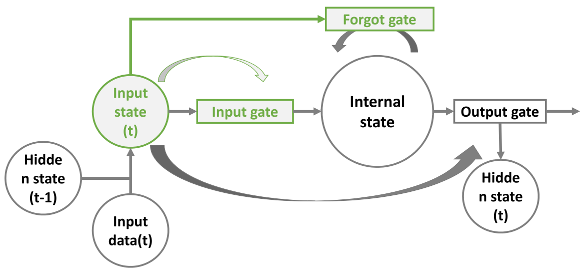
The functioning of the LSTM memory cell is described in Figure 5, where the construction of the input data using the previous hidden state is shown in 5 (b), as described in Equation 2.5. The calculation of the input gate (Equation 2.2) and the forget gate (Equation 2.1) in Figure 5 (c), and the output gate (Equation 2.3) in Figure 5 (d), the update of the internal state (Equation 2.6) in Figure 5 (e) and finally the update of the output of the hidde state are also presented (Equation 2.4) in Figure 5 (f).



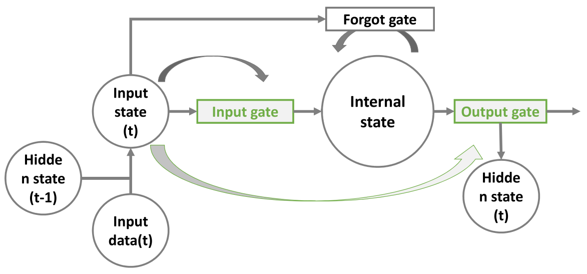
(a) (a) The memory cell



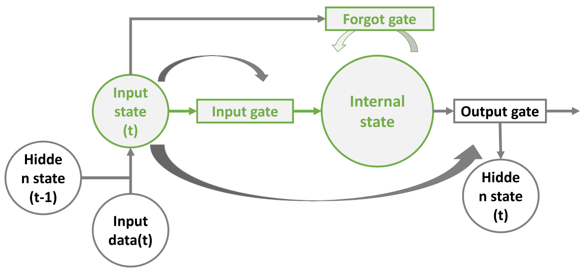
(b) (b) Input data and previous hidden state form into input state



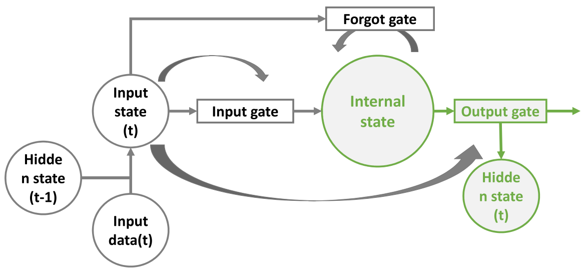
(c) (c) Calculating input gate and forget gate



(d) (d) Calculating output gate



(e) (e) Update internal state



(f) (f) Output and update hidden state

Figure 5: The memory cell of a LSTM, adapted from (WANG; RAJ, 2017)

2.3.2 Random Forest

Proposed by Breiman (2001) in 2001, the Random Forest works with *n* decision trees. In the RF algorithm, several subsets of data are created from the original set. Each subset will serve as input to a decision tree, which will generate a classifier. At the end of the process a voting of the results of each tree is carried out, which will select the most voted class for classification or average for regression (LIAW; WIENER et al., 2002).

Random Forest is able to maintain accuracy even when missing data, in addition, is able to handle a large amount of data and variables, including being able to identify the variables that have a greater weight, that is, it is even able to reduce the number of variables resulting in a decrease in the dimension of the data (BREIMAN, 2001).

In general, the Random Forest works by creating a set n of sampling through bootstraping for training, these data are normally taken from a population corresponding to $\frac{2}{3}$ of the original data. The remaining $\frac{1}{3}$ left out are called out-of-bag (OOB) and will be applied as a way of calculating errors. The bootstraping sampling data is then used to create the trees that will generate a final result. Each sample will have its value calculated in a process called Bootstrap Aggregating, also called bagging (6). In the case of a classification tree a vote will be used, and in a regression tree the average will be used.

The OOB is also used in Random Forest to measure the importance of the variable since the errors generated in the OOB represent the number of times that the majority vote, or the average in the case of regression, resulted in an incorrect value. The tree then replaces the values of the explanatory variable, and if the error increases the importance of the attribute changes.

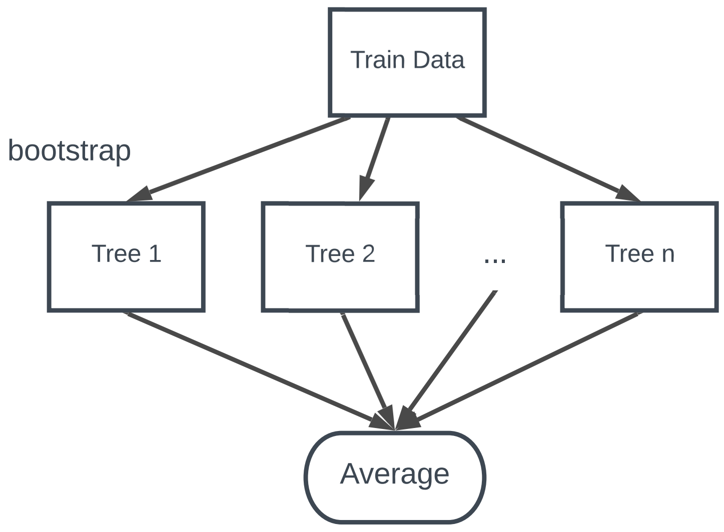


Figure 6: Structure of bagging.

2.3.3 Multilayer Perceptron

The multilayer perceptron (MLP) consists of a system of interconnected neurons, forming a model representing the nonlinear mapping between an input vector and an output vector as seen in figure 7, where $i = [i_1, i_2, i_3, i_4]$ represent the input vector and $o = [o_1, o_2]$ represent the output vector (GARDNER; DORLING, 1998).

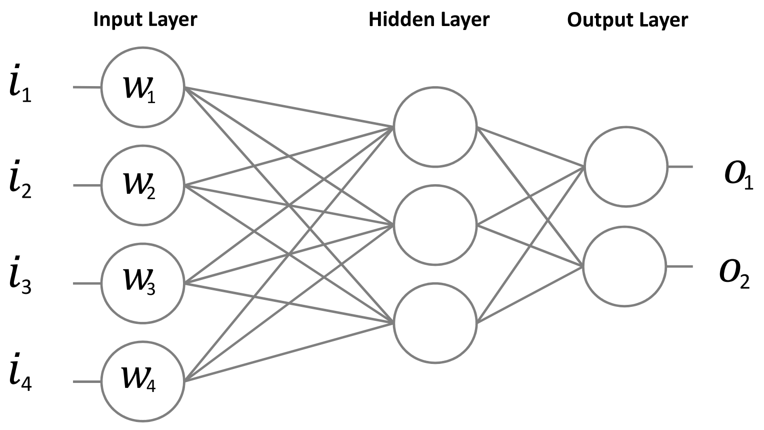


Figure 7: System of interconnected neurons in an MLP (adapted from (GARDNER; DORLING, 1998)).

Each neuron generates an output, which sized by the weight of its connection feeds neurons from the following layers, and is thus known as a feed-foward network. An MLP can have multiple hidden layers, but always an input layer and an output layer, and each neuron in a layer is connected to all neurons in the next layer.

In the input layer, also known as the sensory layer, no calculation is performed and in Figure 7 we can consider the value of i_1 as the input Bias with value of 1. In the output layer the number of neurons is normally associated with the number of classes to be classified. An MLP architecture can contain one or more hidden layers and is responsible for representing the non-linearities between the set of inputs and the desired outputs.

The neuron output value of the hidden layer is given through the activation function f. This function is represented in the equation below where the sum of the input values is multiplied by their respective weights W

$$f(i_1 * w_1 + i_2 * w_2 + ... + i_n * w_n)$$

It is possible to teach an MLP network using a set of annotated data as training. This supervised learning occurs thanks to the ability to adjust the weight until the mapping between the inputs and the desired output occurs. Once trained, an MLP can generalize new input data not yet seen by the network.

The MLP can be applied to tasks that involve prediction of previous and future trends in temporal data series, function approximation modeling relations between variables or classification of patterns in discrete classes (GARDNER; DORLING, 1998).

One of the algorithms used in MLP networks is the backpropagation, proposed by Rumelhart, Hinton e Williams (1986). Its operation is based on learning by correction through the implementation where at first a vector of input is propagated layer by layer by the network until the exit, in this phase the net weights are fixed. In the next phase, called backpropagation, the error obtained is propagated in the reverse direction, adjusting the people through a correction rule in order to approximate the desired response by the network.

2.3.4 Autoregressive integrated moving average

The auto-regressive integrated moving average (ARIMA) is one of the most popular and versatile model for forecasting time series. It consists of a linear function with several past observations and random errors used to generate future values. For this, it is necessary to generate an ARIMA(*p, d, q*) model, where p, d, and q are non-negative integers. The ARIMA model(*p, d, q*) is a generalization of the ARMA model (*p, q*), where *p* is the order or number of lags in the model, and q is the order of the model in the moving average part. In the ARIMA model is inserted a degree of differentiation or the number of times the data had past values subtracted so that the series becomes stationary (ZHANG, 2003).

Methodologies are provides for ARIMA model construction and (BOX; JENKINS, 1970) present a three-step methodology for this task as see in Figure 8, the steps of model identification, parameter estimation and diagnostic checking. In the identification step, it is necessary to achieve one of the conditions of the ARIMA model, which is to ensure that the time series is stationary, for this the data transformation is performed. This stationary time series transformation allows the data to have static characteristics, such as the mean and the autocorrelation structure constant over time. If there is a trend and heteroscedasticity in the transformed data, differentials and power transformation are often applied to the data to remove the trend and stabilize variance before an ARIMA model can be fitted. In the parameter estimation step the parameters are estimated so that a general error measurement is minimized. This can be done with a nonlinear optimization procedure. And finally, the diagnostic checking, by means of a residue analysis to make sure that the model is suitable for the purpose of the technique, the prediction. If the model is not suitable, the cycle is repeated, returning to the initial phase of identification.

2.4 Technique for order preference by similarity to ideal solution (TOPSIS)

Based on the concept where the choice of an alternative must have the shortest possible distance to the ideal solution, also called positive-ideal, and the longest distance to a so-called

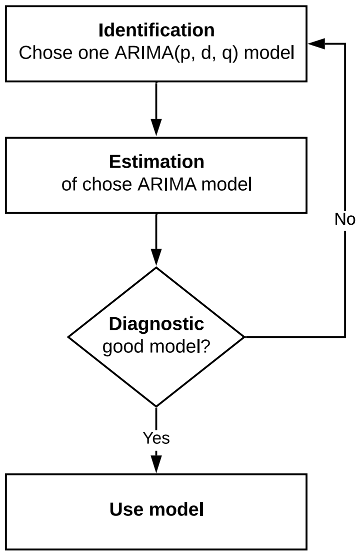


Figure 8: Three-step methodology of Box-Jenkins method to create an ARIMA model.

negative-ideal solution, Yoon e Hwang (1981) developed the technique for order preference by similarity to ideal solution (TOPSIS).

A solution is considered ideal when it has an ideal level or classification for all attributes, which in general is considered unattainable, so the objective becomes a search for the solution that comes close to a logical choice (YOON; HWANG, 1995). The equation used to denote the ideal-solution is present in equation 2.7

$$A^* = x_1^*, ..., x_j^*, ..., x_n^* \tag{2.7}$$

To represent the negative-ideal, the equation 2.8 is applied, this being an equation composed of the attributes with the worst classifications (YOON; HWANG, 1995)

$$A^- = x_1^-, ..., x_j^-, ..., x_n^- \tag{2.8}$$

In general, TOPSIS creates an index of similarity with the positive-ideal solution and as far away as possible from the negative-ideal solution. To achieve this solution, the following steps are performed (YOON; HWANG, 1981, 1995).

2.4.1 Calculate Normalized Ratings

This process tries to transform dimensions of non-dimensional attributes, allowing the addition of new attributes. To perform this step, the result of each criterion divided by the norm of the total result vector of the criterion in question can be used. An element r_{ij} of a normalized R decision matrix can be calculated using the following equation 2.9:

$$r_{ij} = \frac{x_{ij}}{\sqrt{\sum_{i=1}^{m} x_{ij}^2}} \tag{2.9}$$

2.4.2 Calculate Weighted Normalized Ratings

A decision matrix V is defined, this matrix corresponds to the multiplication of the attribute matrix with a set of weights $w = [w_1, w_2, ..., w_m]$ given to each attribute in order to satisfy $\sum_{j=1}^{n} w_j = 1$. In the standardized decision matrix presented below in equation 2.10 A_i denote the alternatives being evaluated and C_j refers to the criteria:

$$V = \begin{array}{c} \\ A_1 \\ \vdots \\ A_i \\ \vdots \\ A_n \end{array} \begin{array}{c} \begin{array}{cccccc} C_1 & C_2 & .. & C_j & .. & C_m \end{array} \\ \begin{bmatrix} d_{11} & d_{21} & .. & d_{1j} & .. & d_{1m} \\ \vdots & \vdots & & \vdots & & \vdots \\ d_{i1} & d_{i2} & .. & d_{ij} & .. & d_{im} \\ \vdots & \vdots & & \vdots & & \vdots \\ d_{n1} & d_{n2} & .. & d_{nj} & .. & d_{nm} \end{bmatrix} \end{array} \tag{2.10}$$

2.4.3 Determine ideal and negative-ideal solutions

In this step the equation 2.11 is applied to determine the ideal-positive, and similarly but to determine the negative-ideal solution the equation 2.12 is applied.

$$A^{+} = \left\{ {}_{j}^{MAX} n_{ij} | j = 1, 2, ...m \right\} = \left\{ n_1^{+}, ..., n_j^{+}, ..., d_m^{+} \right\} \tag{2.11}$$

$$A^{-} = \left\{ {}_{j}^{MAX} n_{ij} | j = 1, 2, ...m \right\} = \left\{ n_1^{-}, ..., n_j^{-}, ..., d_m^{-} \right\} \tag{2.12}$$

2.4.4 Calculate the separation measure

In this step, the distance between the normalized and weighted performance values of the matrix R is calculated with the values of the ideal-positive solution using the equation 2.13:

$$S_{i*} = \sqrt{\sum_{j=1}^{n} (v_{ij} - v_j^{*})^2} \tag{2.13}$$

Similar to the previous equation but to determine the ideal-negative solution, the one given by the equation 2.14

Row	Attribute normalization and weighing	Complexity analysis
1	R = zeros (n,n);	$O(1)$
2	Y = zeros (n,n);	$O(1)$
3	for j=1:n;	$O(n^2)$
4	TotX = 0;	$O(n)$
5	for i = 1:n;	$O(n)$
6	Totx = Totx + X(i, j) * X(i, j);	$O(1)$
7	end;	$O(n)$
8	LX(j) = sqrt(Totx);	$O(n)$
9	R(:, j)= X(:, j). / LX(j);	$O(n)$
10	Y(:, j)= W(j) *R(:, j);	$O(n)$
11	end;	$O(1)$

Table 1: Complexity analysis of attribute normalization and weighing (Adapted from (HAMDANI; WARDOYO, 2016))

$$S_{i-} = \sqrt{\sum_{j=1}^{n}(v_{ij} - v_j^-)^2} \tag{2.14}$$

2.4.5 Calculate the relative closeness to the ideal solution

This step aims to calculate the value of *i* and determine its global performance. The value of *i* is found through the equation 2.15

$$c_{i*} = S_{i-}/(S_{i-} + S_{i-}) \tag{2.15}$$

2.4.6 Rank the preference order

Finally, the last step is, through the values obtained in the global performance calculation, to sort the results in a descending way, generating the final classification.

2.4.7 Computational Cost

As already mentioned in the section 2.4, TOPSIS follows some steps for its execution. Here, the computational costs of executing each of these steps will be presented. For this purpose, the time complexity metric will be used. Starting with the first two steps of attribute normalization and weighing, shown in the table 1 where we have the complexity value of the algorithm of $O(n^2)$.

The next step is to find the computational cost for the next step responsible for finding the positive-ideal solution (PIS) and the negative-ideal solution (NIS). Following the values obtained and presented in the Table 2 the complexity obtained is $O(n)$.

Row	Attribute normalization and weighing	Complexity analysis
1	for i = 1:n	$O(n)$
2	if K(j) == 1;	$O(1)$
3	PIS(j) = max(Y(:, j));	$O(1)$
4	NIS(j) = min(Y(:, j));	$O(1)$
5	else;	$O(1)$
6	NIS(j) = min(Y(:, j));	$O(1)$
7	PIS(j) = max(Y(:, j));	$O(1)$
8	end;	$O(1)$
9	end;	$O(1)$

Table 2: Complexity analysis of PIS and NIS (Adapted from (HAMDANI; WARDOYO, 2016))

Row	Attribute normalization and weighing	Complexity analysis
1	for i = 1:n	$O(n)$
2	S1(i) = sqrt(sum((Y(I, :) – A1(1, :)).^{2}));	$O(1)$
3	S2(i) = sqrt(sum((Y(I, :) – A2(1, :)).^{2}));	$O(1)$
4	S3(i) = sqrt(sum((Y(I, :) – A3(1, :)).^{2}));	$O(1)$
5	S4(i) = sqrt(sum((Y(I, :) – A4(1, :)).^{2}));	$O(1)$
6	end;	$O(1)$

Table 3: Calculation of the distance of values compared to NIS and PIS (Adapted from (HAMDANI; WARDOYO, 2016))

In the table 3 the complexity of the algorithm related to the phase of calculating the distance of the values compared to the NIS and PIS is presented. The result obtained for the complexity of the algorithm was $O(n)$.

At the end, as the last step of TOPSIS we have in the Table 4 the calculation of the global performance and the creation of the ranking to generate the final classification. The result of the algorithm complexity obtained is $O(n)$.

2.5 ε-Greedy

First described by Watkins (1989), the ε-Greedy is one of the simplest and most widely used strategies to solve the multi-armed bandit problem (VERMOREL; MOHRI, 2005). Multi-armed bandit (or K-armed) problem it is a classic problem in decision theory that has been studied since the 50's.

Bearing in mind a scenario where there is a set of actions and rewards, where each action can bring a certain reward and considering that this reward can be used as knowledge to assist in

Row	Attribute normalization and weighing	Complexity analysis
1	V = S2./(S1 + S2)	$O(n)$

Table 4: Complexity analysis of ranking. (Adapted from (HAMDANI; WARDOYO, 2016))

future decisions, questions address the need to balance the maximization of the reward based on in the knowledge already acquired and in the attempt of new actions to increase the knowledge even more. We can call this exploration-based learning as reinforcement learning (MANNOR; TSITSIKLIS, 2004).

There are some algorithms to deal with the multi-armed bandit problem, one of the commonly used algorithms is the Greedy algorithms. Greedy algorithms are powerful methods that work for a variety of problems where problem optimization is being pursued. This optimization is done through choices made within a set of steps, always making the decisions that seem to be the best at the moment and being performed quickly (CORMEN et al., 2009).

In the ε-Greedy algorithm, the epsilon is used to find a balance between exploration and exploitation modes, with epsilon fixed, leaving the user to choose the value or leaving epsilon adaptive. The strategy adopted in the use of ε-Greedy selects the best lever in a proportion of 1 - εand a lever is selected randomly (with uniform probability) for a proportion being the value of εnormally 0.1, but can be variable. (VERMOREL; MOHRI, 2005).

There is more than one version of the implementation and use of ε-Greedy, where we can highlight the ε-first strategy, who does the exploration all in the beginning, still in the first rounds. We also have the ε-decreasing strategy where through the use of a function to perform the decrease, approaching the optimal strategy asymptotically.

2.5.1 Computational Cost

The complexity of the algorithm to execute the ε-Greedy can be seen in the Table 5, where the value of $O(n \log n)$ was found resulting from the need to order the values of the matrix so it is possible to obtain the lowest value as the best result.

Row	ε-Greedy	Complexity analysis
1	V = QuickSort(array);	$O(n \log n)$
2	R = Rand(0:100);	$O(1)$
3	if R < ε;	$O(1)$
4	R = Rand(0:n);	$O(1)$
5	else	$O(1)$
6	R = 1;	$O(1)$
7	end	$O(1)$

Table 5: Complexity analysis of ranking (Adapted from (HAMDANI; WARDOYO, 2016))

3 RELATED WORK

In order to identify relevant frameworks, architectures, and tools in the area of predictive maintenance in this section is presented a systematic literature review (SLR) methodology (KITCHENHAM et al., 2010). Also, we discuss the challenges and investigate the main contributions in the field of research over the last five years. Our study considers an initial corpus of 318 papers that are filtered and classified into four groups, considering the approach of each research: (I) Integration issues, (II) big data analysis, (III) machine learning approaches, and (IV) reasoning and ontologies. We discuss the top-rated papers in detail and use this corpus to answer four research questions that help one to understand the state-of-the-art and the main future challenges of predictive maintenance.

3.1 Research Methodology

A systematic literature review is an approach used to identify, evaluate, and interpret the papers published in a given field of research. This approach enables the identification of existing gaps and points out new research opportunities (KITCHENHAM, 2004). In this article, we follow the SLR approach proposed by Kitchenham et al. (2010). Specifically, we applied the methodological steps as follows.

1. **Definition of research questions:** it guides the elaboration of research questions to be used to search for relevant papers in the literature;
2. **Search process:** it presents the research strategy and the scientific sources used in the search for relevant papers;
3. **Studies selection:** definition of the criteria applied to select relevant papers;
4. **Quality assessment:** quantitative analysis of the quality of the selected studies;

We discuss each of the steps in the following subsections.

3.1.1 Research Questions

An important part of an SLR is the elaboration of research questions (KITCHENHAM, 2004). In this article, the research questions should provide the means to understand the use of ML along with ontologies in the context of PdM in Industry 4.0 scenarios.

We specify a general question to guide the search for challenges in the field of research. Based on the central question, we establish specific ones to emphasize existing solutions and to identify gaps and directions for future research.

What are the challenges and open questions regarding machine learning and reasoning for predictive maintenance in Industry 4.0?

With the general research question in mind, we created the following specific questions (SQ).

- SQ1: What are the challenges of applying machine learning towards predictive maintenance?
- SQ2: Which machine learning techniques are commonly applied in the context of predictive maintenance?
- SQ3: In which contexts are ontologies used for predictive maintenance?

3.1.2 Search Process

We perform two steps to conduct the search process. The first one is the creation of a search string, while the second involves the selection of the sources. The design of the search string demands a preliminary reading of selected papers related to the field of interest. Boolean operators are applied to improve search string performance. These operators contemplate terms that are synonymous with the keywords already defined.

("Industry 4.0" AND "Machine Learning" AND "Predictive Maintenance" AND ("Architecture" OR "Framework") AND ("Ontology" OR "Reasoning"))

In this article, all the results obtained came from electronic sources. The choice of the databases aimed at analyzing papers published in journals and conferences that cover the concepts of Industry 4.0. The selected electronic databases were IEEE[1], Google Scholar[2], Springer[3], ACM Digital Library[4] and ScienceDirect[5].

3.1.3 Papers Selection Process

Following the definition of the search string and gathering articles from the selected electronic databases, it is necessary to remove all studies that are not relevant to the goals of this article. To remove these papers, the following exclusion criteria (EC) applies.

[1]https://ieeexplore.ieee.org/
[2]https://scholar.google.com/
[3]https://link.springer.com/
[4]https://dl.acm.org/
[5]https://www.sciencedirect.com/

- EC 1: papers not directly related to PdM.
- EC 2: papers not directly related to ML.
- EC 3: papers that presented results of surveys or reviews.
- EC 4: papers published before the year 2015.

Two researchers conduct the filtering process following the steps presented below. The results are analyzed by a third one whenever a discrepancy occurred.

1. **Removal of duplicates:** in some situations, the same paper is available in different sources, like in IEEE and Google Scholar, for example. In this case, we remove the duplicates;
2. **Title analysis:** the researcher reads the title of the paper and judges whether or not it is sufficient to assess the importance of the paper to the study;
3. **Abstract analysis:** when the analysis of the title is not enough to make a decision, the researcher reads the abstract of the paper to get a better understanding of the approach;
4. **Entire text analysis:** it applies in situations where the title and the abstract are not very clear about the proposed solution. Nevertheless, the presented ideas look promising for the goals of this literature review.

As part of the SLR methodology, we exclude from the corpora papers published before 2015 and those classified as surveys or reviews. Besides, we disregarded any work with no scientific character, such as a blog post or magazine article. We also remove the duplicates of papers that appear in more than one database. After the conclusion of this step of the SLR, the remaining papers pass to the phase of quantitative evaluation.

3.1.4 Quality Assessment

According to the methodology (KITCHENHAM et al., 2010), in this step, we define the criteria for qualitative evaluation of the selected papers. The evaluation takes into account the following points: (I) the purpose of the research; (II) whether the authors contemplate a research methodology or propose an architecture or a framework; (III) the results accomplished; and (IV) whether the selected work uses ontologies. The following questions apply to select papers that meet the quality requirements.

- Is the purpose of the research presented?
- Is there an architecture/framework proposal or a research methodology?

- Are the research results presented and discussed?
- Does the paper use reasoning or an ontology?

Based on Kitchenham's methodology (KITCHENHAM et al., 2010), we define three possible answers, each one receiving a grade: Yes=1, Partial=0.5, and No=0. After two researchers have graded the papers, a discussion meeting was conducted to deal with discrepancies. At the end of such a meeting, to decide whether each study should be kept or excluded from the original corpora, we then applied criteria for excluding articles according to the grade. More formally, such criteria are:

- Articles mainly organized as comments or personal opinions are excluded from the set since they usually do not present a validation methodology;
- Articles that are graded below 2.5 by the researchers, since at least 2 of the questions received 'no' or 'partial' as the answer, indicating a not very relevant publication for this SLR;

After applying these above criteria onto the original set of papers, we read the remaining ones to answer the research question. The resulting analysis is presented in Section 3.2.

3.2 Search Results

This section discusses the result of the search process, the selection process, and the qualitative analysis of the selected papers. We summarize the results in figure 9. The description of each step and the number of remaining papers are also presented in figure 9.

We detail the application of the SLR methodology as follows. Subsection 3.2.1 discusses relevant papers that are applied to the context of Industry 4.0 but do not meet all the criteria to be part of this SLR. After analyzing the exclusion criteria, details on the quality assessment of the papers are presented in subsection 3.2.2.

3.2.1 Exclusion of papers from the initial corpora

The initial search returned a total of 383 papers from five publication sources. The first filter was the removal of impurities, which means the removal of duplicates and also of documents such as surveys, reviews, book chapters, or non-scientific papers like magazine articles, resulting in a total of 218 out of 383 articles. The next step comprised the application of the exclusion criteria mentioned in section 3.1.3. The number of papers considered relevant for these reviews reduced down to 89 in this phase.

We removed some papers because, despite clearly addressing issues related to PdM, they do not reflect the use of ML models. For example, Stojanovic e Stojanovic (2017) created an architecture that applies the concepts of big data in the context of self-healing manufacturing. The

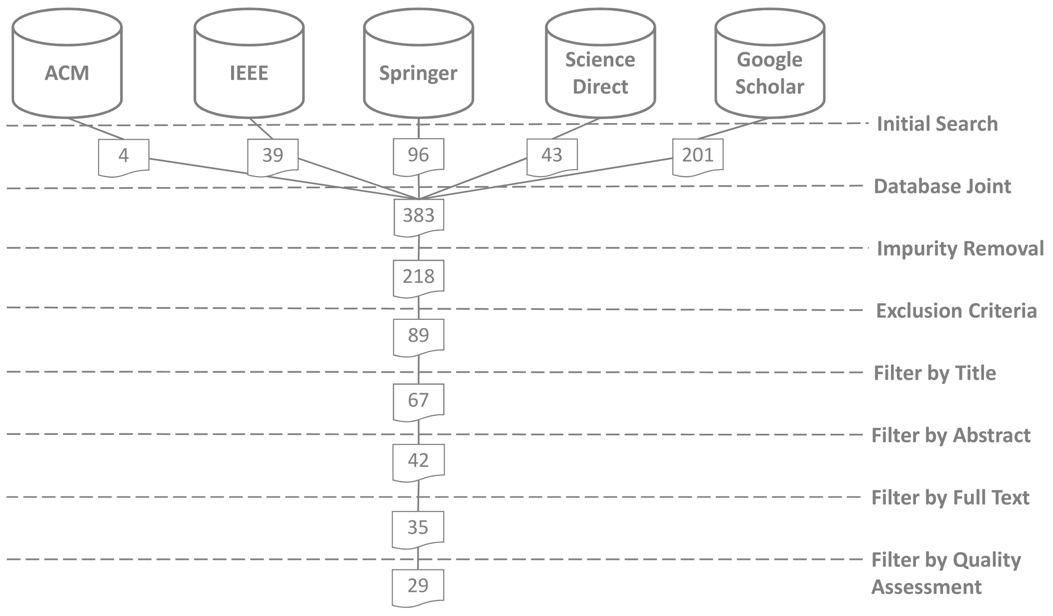


Figure 9: Filter process of selected articles by database

solution, named PREMIuM, briefly mentions ML models as part of a larger prediction architecture. Between the several layers of PREMIuM, the cloud layer is responsible for analyzing data using a strategy the involves at least two ML methods. Despite that, the paper does not explain which methods are applied or how training The exclusion criteria also removed a solution proposed by Zenisek et al. (2018). This solution generates the first data streams as a way of helping time series researchers to simulate scenarios with realistic monitoring conditions. We also removed the intelligent maintenance support framework proposed by Bumblauskas et al. (2017). In this case, the authors proposed an algorithm to minimize costs across the supply chain and propose a methodology for establishing predictive maintenance plans. Both approaches fail to apply ML models and therefore were considered not relevant for the purposes of this SLR.

May et al. (2018) proposed a novel approach to prevent failures. The approach is a set of strategies intended to extend the life of production systems. This set of strategies, called Z-strategy, is capable of predicting failures at the component, machine, or system level. This platform matches the subject of PdM but is not related to ML. Golightly, Kefalidou e Sharples (2018) concentrated on human factors such as data interpretation and visualization. The study identified factors to help the implementation of predictive asset management. Thus, through interviews with experts, the authors identified the organizational problems associated with the development and adoption of predictive maintenance systems. The results are recommendations on how to mitigate these problems.

The solutions show several cases where the application of PdM goes beyond the application of ML models. These solutions are generally related to the elaboration of frameworks that introduce the concepts of PdM with a more holistic view. We can also find opposite approaches in the literature. These solutions, although addressing ML in the context of Industry 4.0, do not apply for PdM. These works appeared in the initial corpora because they mention PdM-related terms in sections like related work or references.

In this context, Syafrudin et al. (2018) designed a real-time system to improve decision making. This system helps to prevent losses due to unplanned manufacturing failures. The system's workflow is split into three steps. The first one is the selection of IoT devices to monitor an automotive manufacturing environment. The second one involves processing large amounts of generated data. The final step is the creation of a hybrid model for fault detection. The hybrid model uses density-based spatial clustering of applications for anomaly detection, and Random Forest to classify events as normal or abnormal.

Romeo et al. (2020) provides an ML-enabled framework to help designers and laboratory technicians to make the best operating description of a machine. The solution relies on Decision Trees, k-Nearest Neighbors, and Neighborhood Component Features Selection algorithms to obtain the recommendations. The resulting solution predicts whether the machine specifications, e.g., the number of blades, speed, and shaft size, match the operating parameters, like torque, flow, pressure, and gate. The authors claim that the solution provides easier decision making, conserving company knowledge, saving working hours, and increasing computational speed and accuracy.

Zhang et al. (2019) propose an approach to apply ML techniques in the industry. The solution consists of a framework to provide a reference for planning, design, and the application of industrial artificial intelligence in different areas of manufacturing. The solution is theoretical and broadly covers seven dimensions related to industrial artificial intelligence: objects, domain, application stages, application requirements, intelligent technology, intelligent function, and solutions. The framework was evaluated considering five industrial fields and showed to be effective in helping industries to plan how to use artificial intelligence-related solutions.

Ali, Patel e Breslin (2019) designed a software middleware to collect and analyze data from different applications in a real-time fashion. The authors evaluate the solution in a scenario where the middleware provided information to ensure optimal production forecasts. Even though not dealing with PdM, the use of ML for production forecasting presents some challenges that look like those of PdM. These challenges include the difficulty of obtaining relevant data without lacks or gaps and the need to require domain knowledge for the selection of variables and construction of models. According to the nature of the collected data, Multiple Linear Regression, Support Vector Regression, Decision Tree, and Random Forest algorithms were applied by the middleware to make predictions.

Malek (2017) proposed a failure prediction methodology to provide reliability in different scenarios. The authors evaluated the methodology considering two scenarios: (I) malware detection and (II) computer failure detection. To do that, the authors applied Naive Bayes, Logistic

Regression, and J48 Decision Tree algorithms. The paper briefly mentions concepts related to Industry 4.0 and PdM in the related works section.

A framework combining data collection, pre-processing, and ML models training to identify behaviors that may influence manufacturing is the proposal of Carbery, Woods e Marshall (2018a). The solution uses artificial intelligence to assist engineers in increasing machine performance and supporting decision making. It results in a four-stage workflow: (I) data collection, (II) pre-processing, (III) training data generation, and (IV) artificial intelligence model creation. The authors focused on challenges and solutions for data pre-processing and feature selection.

To demonstrate the role of cloud computing and the use of artificial intelligence to improve factory performance, Wan et al. (2018) proposed a vertically integrated, four-tier cloud-assisted smart factory architecture. The layers that compose the architecture are: (I) Smart Device Layer, (II) Network Layer, (III) Cloud Layer, and (IV) Application Layer. ML models are implemented in the Network Layer to perform tasks involving network optimization. These models are also present in the Application Layer, where the use of ML aims to perform failure detection, but not predictive maintenance.

Costa et al. (2017) introduced a framework for knowledge representation. The framework transforms unstructured data such as logs or machine documentation into highly structured representations. The approach uses an ontology to assist in structuring and enriching information. It also applies machine learning techniques to process natural language. Although the solution uses reasoning-related techniques, its purpose is not related to predictive maintenance.

Finally, we can cite the work developed by Sala et al. (2018). The solution applies a data-driven strategy to predict temperature and chemical concentration in the Basic Oxygen Furnace Steelmaking process. To do that, the authors apply different machine learning models, like Ridge Regression, Random Forest, and Gradient Boosted Regression Trees. As in several other works, the authors only mention PdM-related terms.

After applying the exclusion criteria, we screened the 89 remaining papers considering three steps: (I) filter by title, (II) filter by abstract, and (III) filter by full text to perform the quality assessment of the papers. The quality assessment decides whether, although not eliminated by the exclusion criteria, a given paper is relevant for the SLR.

3.2.2 Performing the Quality Assessment to Select Relevant Papers

This section follows the quality criteria defined in section 3.1.4 to conduct the qualitative analysis of the papers. Researchers answered to each question according to Kitchenham's methodology (KITCHENHAM et al., 2010). We present the possible answers and the respective grades in table 6. Relevant papers for this SLR are those that received 2.5 points or more.

To score the papers, the researchers first applied a filter by the title that reduced the number of papers from 89 to 67. These 67 papers went through the abstract filtering process, which

Table 6: Answers and Grades

Answer	Description	Grade
Y (Yes)	the paper explicitly answers the question	1.0
P (Partial)	the paper answers to part of the question	0.5
N (No)	the paper does not mention the topic	0.0

selected 42 relevant ones. At the final filtering process, the remaining papers went through a complete analysis of the text. This phase excluded 7 studies and resulted in the 35 papers considered the most relevant for this systematic literature review.

These 35 papers had their quality assessed considering the following questions:

SQ1: Is the purpose of the research presented?

SQ2: Is there an architecture/framework proposal or a research methodology?

SQ3: Are research results presented and discussed?

SQ4: Does the paper implements the concepts of reasoning or proposes an ontology?

We present the answers to the questions and the resulting scores in table 7, in descending order. References (NUÑEZ; BORSATO, 2018; ANSARI; GLAWAR; NEMETH, 2019; SCHMIDT; WANG; GALAR, 2017) are marked with an asterisk (*) because, according to the filters, these papers should be out of the corpora. However, we kept them because they propose the use of ontologies to implement reasoning in the context of PdM, which is a topic of interest for this SLR.

Table 7: Quality Assessment Scores

Year	Authors	SQ1	SQ2	SQ3	SQ4	Score
2019	Cao et al.	Y	Y	Y	Y	4.0
2018	Nuñez and Borsato	Y	Y	Y	Y	4.0
2019	Ansari, Glawar and Nemeth	Y	Y	Y	Y	4.0
2016	Schmidt, Wang and Galar	Y	Y	Y	Y	4.0
2018	Carbery, Woods and Marshall	Y	Y	Y	N	3.0
2019	Xu et al.	Y	Y	Y	N	3.0
2019	Cerquitelli et al.	Y	Y	Y	N	3.0
2017	Ferreira et al.	Y	Y	Y	N	3.0
2019	Rivas et al.	Y	Y	Y	N	3.0
2017	Crespo et al.	Y	Y	Y	N	3.0
2016	Gatica et al.	Y	Y	Y	N	3.0
2018	Schmidt et al.	Y	Y	Y	N	3.0

Table 7 continued from previous page

Year	Authors	SQ1	SQ2	SQ3	SQ4	Score
2016	Chukwuekwe et al.	Y	Y	Y	N	3.0
2017	Diez-Olivan et al.	Y	Y	Y	N	3.0
2018	Strauss et al.	Y	Y	Y	N	3.0
2018	Zhou et al.	Y	Y	Y	N	3.0
2018	Peres et al.	Y	Y	Y	N	3.0
2018	Liu et al.	Y	Y	Y	N	3.0
2017	Li et al.	Y	Y	Y	N	3.0
2018	Adhikari et al.	Y	Y	Y	N	3.0
2018	Schmidt et al.	Y	Y	Y	N	3.0
2018	Kiangala et al.	Y	Y	Y	N	3.0
2018	Hegedus et al.	Y	Y	N	Y	3.0
2018	Kaur et al.	Y	Y	Y	N	3.0
2019	Bousdekis et al.	Y	Y	P	N	2.5
2018	Cachada et al.	Y	Y	P	N	2.5
2018	May et al.	Y	Y	P	N	2.5
2020	Ansari et al.	Y	Y	P	N	2.5
2019	Sarazin et al.	Y	Y	P	N	2.5
2018	Issam et al.	Y	Y	N	N	2.0
2015	Gao et al.	Y	Y	N	N	2.0
2019	Glawar et al.	Y	Y	N	N	2.0
2019	Talamo et al.	Y	Y	N	N	2.0
2018	Balogh et al.	Y	Y	N	N	2.0
2017	Wang et al.	Y	N	N	N	1.0

After the quality assessment performed over 35 papers, 6 references ((ISSAM; EL MAJD; EL GHAZI, 2018), (GAO et al., 2015), (GLAWAR et al., 2019), (TALAMO; PAGANIN; ROTA, 2019), (BALOGH et al., 2018), and (WANG; WANG, 2017)) were removed due to the score metric. Therefore, the 29 remaining papers were considered as relevant for this SLR. These papers were analyzed in detail. The results of such analysis and discussions regarding the papers are presented in section 3.3.

3.3 Answer to the Research Questions and Discussion

This section analyzes the contributions of the most relevant papers selected in this SLR. To do that, in each subsection, one of the research questions defined in section 3.1.1 is answered, considering the contributions of the papers found in the literature.

3.3.1 What are the challenges and open questions regarding machine learning and reasoning for predictive maintenance in Industry 4.0?

This research question contributes to the scientific community by identifying and classifying the current challenges and open issues regarding machine learning and reasoning in the context of PdM. To do that, we propose the taxonomy presented in figure 10.

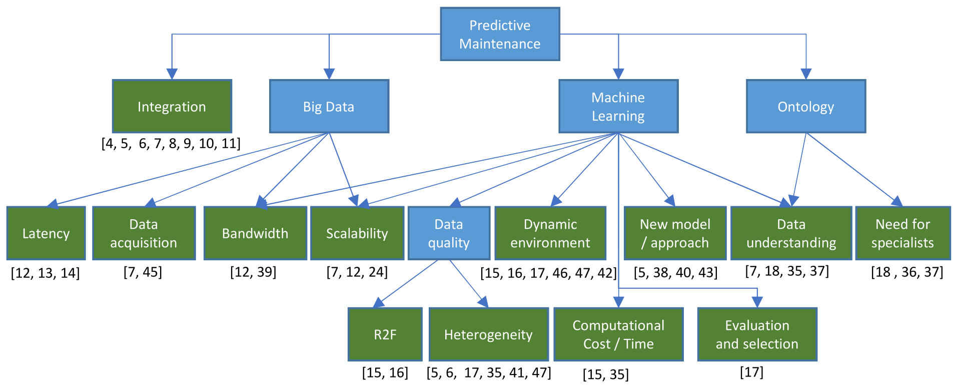


Figure 10: A Taxonomy to Classify Challenges and Open Issues in PdM

We designed the taxonomy considering two types of elements. The blue boxes represent broad fields related to predictive maintenance that can have different kinds of challenges related to them. On the other hand, the green boxes are specific challenges or open issues that we identified based on the results of the SLR. The numbers under each green box are citations to papers that propose solutions to tackle that specific challenge.

The first element of the taxonomy refers to the general field of *Predictive Maintenance*. The analysis of the literature showed that this field usually poses challenges related to the reduction of maintenance-related costs or aims at improving production efficiency by predicting necessary maintenance. Solutions to deal with these generic challenges can be divided into three groups: (I) *Big Data* analytics, (II) *Machine Learning* models, and (III) *Ontology* and reasoning-related proposals. Each of these three areas has specific challenges in the context of PdM. We present and discuss these challenges as follows.

The first specific challenge in the taxonomy is directly related to predictive maintenance and concerns to integration issues. This kind of issue typically affects the company as a whole. The literature presents several solutions to build integrated solutions and methods that are capable of handling different processes related to PdM. The most common approach of these works is to propose architectures, strategies, principles, and tools that seek to unify each step of the process for deploying systems to enable predictive maintenance (BOUSDEKIS et al., 2019; ANSARI; GLAWAR; SIHN, 2020; SARAZIN et al., 2019; HEGEDŰS; VARGA; MOLDOVÁN, 2018; CACHADA et al., 2018; FERREIRA et al., 2017; KIANGALA; WANG, 2018; KAUR et al., 2018). These tools are generally capable of integrating processes such as sensor data collection on machines, data processing, creation and training of machine learning models, as well as the integration with other information sources, e.g., ERP systems, to deliver the PdM alerts in a user-friendly interface.

Big data is one of the more challenging areas in the context of PdM. Some issues concern the need for real-time monitoring and consequently processing a large amount of data generated by the sensors. In this context, guaranteeing good values on metrics like *Latency*, *Scalability* and network *Bandwidth* a problem as some predicted events require immediate action to prevent failures. To mitigate these problems works like Liu et al. (2018) and Zhou e Tham (2018) propose the use of edge computing to bring the processing closer to the data collection point and delegate to the cloud only tasks that do not require immediate action. On the other hand, Crespo Márquez, Fuente Carmona e Antomarioni (2019) proposed a framework that implements the concept of distributed computing, so when the system capacity reaches high usage values, the system manages the resulting overhead by distributing the processing.

Still in the context of *Big Data*, another open challenge concerns *Data acquisition*. The open issues include the difficulty of obtaining quality data and interpreting it. A considerable portion of the collected data has missing values, is poorly structured, or has no annotations. To deal with this issue, Hegedűs, Varga e Moldován (2018) proposed a solution to pre-process data and turn it usable for predictive maintenance. Another approach to deal with data acquisition is the framework designed by Strauß et al. (2018) that enables the monitoring and data acquisition in legacy machinery.

Another major challenge in predictive maintenance is to establish the grounds for applying *Machine Learning* models. Like in the context of *Big Data*, the need for real-time decision making requires high levels of *Scalability* and network *Bandwidth*. In this sense, training learning models in the edge of the networks is a solution that has been proposed by Liu et al. (2018) and by Cerquitelli et al. (2019). Another challenge in the context of machine learning is to obtain data that shows the tendency of normal state behavior to failure, called run to fail (*R2F*). Xu et al. (2019) and ADHIKARI, RAO e BUDERATH (2018) deeply explore R2F. This kind of data is important to identify problems because, in this case, it is necessary to train the models with annotated failure-related datasets. In this sense, Gatica et al. (2016) proposed a top-down strategy consisting of first understanding machine operation and then taking action to deal with the problem.

Besides, the *heterogeneity* of the datasets also poses as an issue that deserves attention from the scientific community. Both the lack and the excess of heterogeneity harm the machine learning models. The lack of heterogeneity was explored by Selcuk (2017) and by Nuñez e Borsato (2018). In both approaches, the authors conclude that this characteristic makes training the ML models more difficult. The authors also found out that in several cases, only one data source is available, e.g., all data come from one specific machine. On the other hand, excessive heterogeneity also impacts negatively on the ML model training. In this direction, Sarazin et al. (2019) and Gatica et al. (2016) analyzed the behavior of ML models that receive large amounts of training datasets from many different sources. Another problem related to this topic was investigated by Ansari, Glawar e Sihn (2020). It involves the challenges related to processing data with different structures (ANSARI; GLAWAR; SIHN, 2020).

The general conclusion we can reach on this topic is that both lack and excess of *heterogeneity* can impact the predictability of the algorithms. Proposals to mitigate this problem are available. Different authors found out that one of the causes of data heterogeneity is the fact that manufacturing plants are dynamic environments. Schmidt e Wang (2018a) claim it is not recommended to use data obtained only in the laboratory. Moreover, Li, Wang e Wang (2017) showed that it is also not feasible to train ML models using data provided by only one model of equipment. Another aspect emphasized in related work is that the process executed by a given machine can also change dynamically (XU et al., 2019). Alternatives to deal with this challenge include training multiple learning models (ADHIKARI; RAO; BUDERATH, 2018), utilizing data produced on machines that have operated in comparable conditions (SCHMIDT; WANG, 2018b), or applying data mining techniques to generate context information (PERES et al., 2018).

Another challenge related to ML concerns the lack of a universal model that applies to multiple scenarios. In this context, Ansari, Glawar e Sihn (2020) discussed the possibility of proposing new models to deal with data heterogeneity issues. Other authors (RIVAS et al., 2019; CARBERY; WOODS; MARSHALL, 2018b; CHUKWUEKWE; GLESNES; SCHJØLBERG, 2016) advocate in favor of applying existing models to new scenarios. Generally, these approaches test different models to evaluate which one works better in a given situation. As an example, we can mention Schmidt e Wang (2018a), who propose a classification based on vibration limit values to predict failure and uses the accuracy of the models as an evaluation metric. The computational cost for training these ML models is also a challenge to be addressed, according to Xu et al. (2019) and to Nuñez e Borsato (2018).

The last major area identified in this SLR refers to the application of ontologies for predictive maintenance. This scenario typically associates ontologies with the need to understand the data. The process of understanding data involves gathering context information. Context information typically includes the identification of the machine and the process that is in execution at the moment of data collection. It can also include information about the environment, such as temperature and physical location. This area is still incipient, however, some works have been proposed so far to investigate the topic (Hegedűs, Varga e Moldován (2018), Cao et al. (2019), Nuñez e Borsato (2018), and Schmidt, Wang e Galar (2017)).

3.3.2 What machine learning techniques are being used?

The literature covers a wide variety of ML techniques. Each with specific characteristics and applications. The focus of this section is the applications of these models for predictive maintenance. We intend to highlight the commonly used ML techniques and the reasons for selecting specific techniques.

Figure 11 shows a taxonomy we designed to present machine learning techniques applied in the context of PdM. The taxonomy also highlights the connection among classes of algorithms.

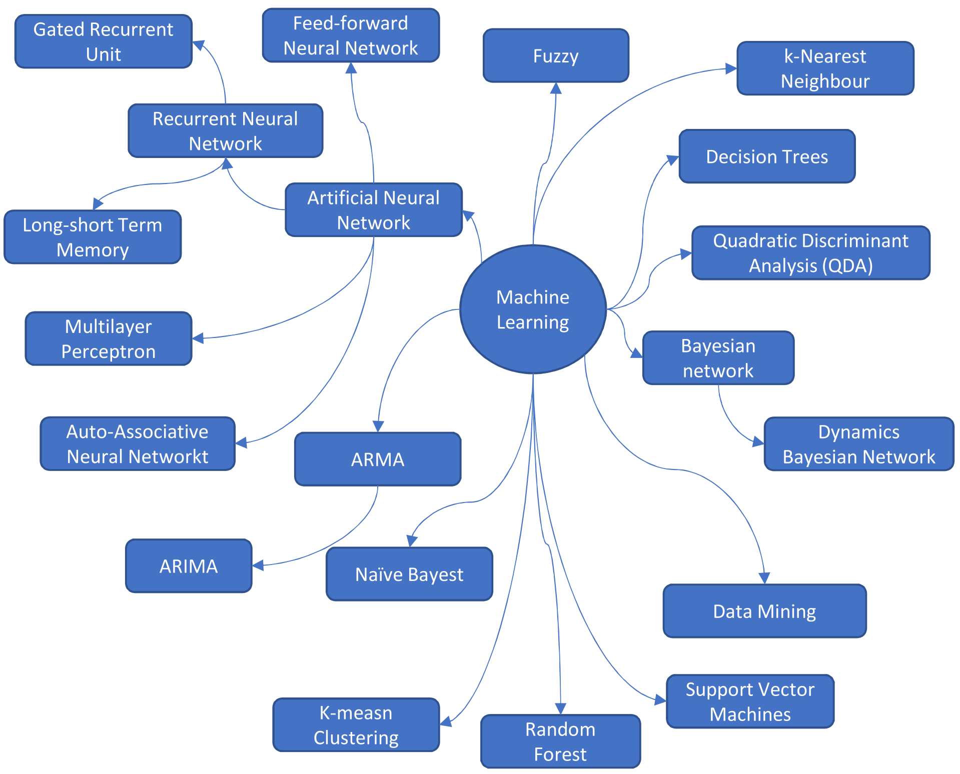


Figure 11: Taxonomy of Machine Learning Techniques

Several authors use Artificial Neural Networks (ANN) to tackle problems related to predictive maintenance. Li et al. (LI; WANG; WANG, 2017) proposed a framework for fault prediction, which is also capable of performing error correction regardless of machine or process type. According to the authors, the selection of this technique occurred because ANNs are already widely used to compensate for slack errors in computer-controlled machine centers. Peres et al. (PERES et al., 2018) applied ANN to identify abnormal behaviors to deal with prediction and classification problems in the context of Industry 4.0.

Other solutions involve the implementation of Recurrent Neural Networks (RNN) that are a type of ANN capable of incorporating memory. Rivas et al. (RIVAS et al., 2019) adopted Long Short-Term Memory (LSTM) RNN model for failure prediction. Cachada et al. (CACHADA et al., 2018) also used LSTM along with a second technique called Gated Recurrent Unit (GRU) for a similar purpose. Both models were applied because they implement the ability to consider historical data to predict future behavior. Several authors ((SCHMIDT; WANG, 2018a; ZHOU; THAM, 2018; NUÑEZ; BORSATO, 2018)) assess the performance of ANN and compare it with other techniques like Support Vector Machine (SVM) and Randon Forest (RF). Yet in the area of ANN, some works consider the implementation of Auto-Associative Neural Networks (AANN). Liu et al. (LIU et al., 2018) proposed a PdM framework in which an AANN identifies irregularities in railways. This information is used to predict failures and suggests actions to be taken in advance.

Another common approach identified in the SLR is the application of ML models to prove a concept. Schmidt et al. (SCHMIDT; WANG, 2018a) evaluated the performance of k-Nearest Neighbor (kNN), Back-propagation Feed-forward Neural Network (FFNN), Decision Tree (DT), and Naïve Bayesian (NB) in various scenarios to obtain the best combination of techniques to deal with time-series prediction. Zhou et al. (ZHOU; THAM, 2018) also used KNN and DT along with SVM and Multi-layer Perceptron (MLP), to evaluate a framework for diagnostics and prognostics proposed in the same study.

Dealing with large amounts of data to predict failures was the goal of Carbery et al. (CARBERY; WOODS; MARSHALL, 2018b). To do that, the authors used a Bayesian Network (BN). The selection of this technique was justified because BN is known for performing well under uncertainties. Moreover, this technique can decompose complex problems in more manageable ones using conditional probabilities. A special case of BN, called Dynamics Bayesian Network, is proposed by Ansari et al. (ANSARI; GLAWAR; SIHN, 2020). The proposal is part of a framework designed to predict failures and to measure the impact of such a prediction on the quality of production planning processes and maintenance costs.

Another class of models that is gaining attention from the scientific community is Autoregressive Moving Average (ARMA). One example of an application is the solution presented by Chukwuekwe et al. (CHUKWUEKWE; GLESNES; SCHJØLBERG, 2016). The solution considers the vibration date to predict failures in the context of Industry 4.0. Auto-regressive Integrated Moving Average (ARIMA), which is a variation of ARMA, was applied by Adhikari et al. (ADHIKARI; RAO; BUDERATH, 2018) in a predictive maintenance framework to predict the remaining useful life of components. The selection of ARIMA was due to its ability to use historical data to estimate future behavior.

3.3.3 What are the contexts in the use of ontology in predictive maintenance?

Predictive maintenance can rely on ontologies for various purposes. In figure 12, we propose a taxonomy that presents the main applications of ontologies in predictive maintenance. This section discusses these applications.

One of the main goals of an ontology is to provide *context awareness*. Prima framework (ANSARI; GLAWAR; NEMETH, 2019) applies this concept. Prima models real-world objects according to their properties and functions. The solution is capable of gathering and associating sensor data. The framework allows, for example, the association of a motion sensor with a specific machine process and, at the same time, with energy consumption information. This kind of approach *models knowledge and shares* it for decision-making purposes. Prima also *stores information semantically*. The solution models and stores data in a standardized manner, which allows the framework to access various data sources. This characteristic also copes with another feature, which is to provide *interoperability* among different domains, achieved by

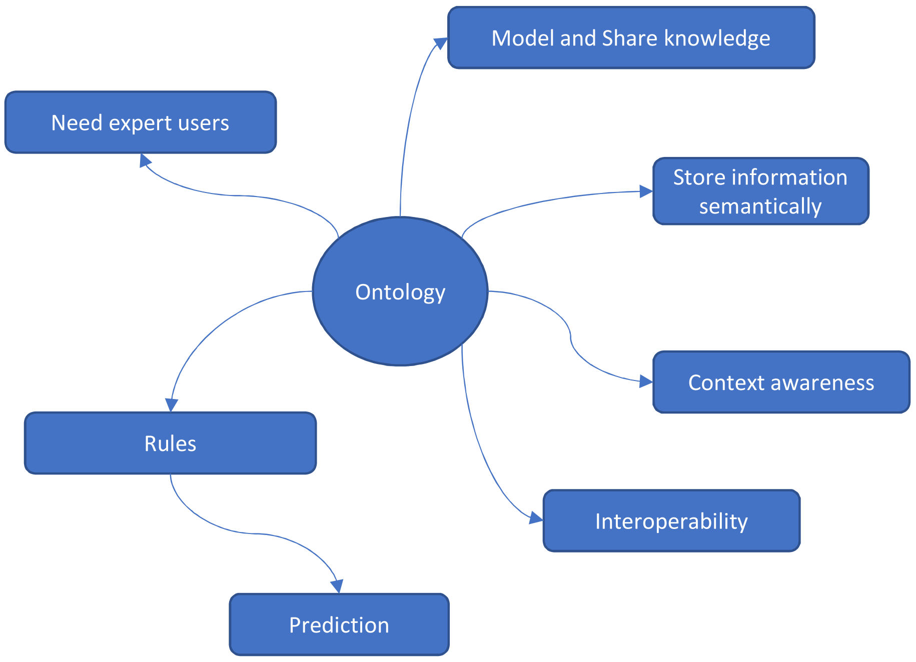


Figure 12: Taxonomy of Ontologies Applications in Predictive Maintenance

implementing the so-called domain ontology concept.

Cao et al. (CAO et al., 2019) explore the ability to *store information semantically*. The work explores the context of condition-based maintenance. Using the concept of *rules*, the authors propose an algorithm that generates Semantic Web Rule Language (SWRL). This solution implements reasoning to describe events and temporal constraints. The results facilitate decision making through the *prediction* of failures.

A challenging issue, which affects both ML models and ontologies, is the need for *expert users* to analyze the context and provide crucial information to set the parameters of the models. This need is discussed by Nunez et al. (NUÑEZ; BORSATO, 2018). In their solution, the authors use expert knowledge to formalize an ontology to perform vibration analysis of machine components. The authors use information stored in the ontology to create SWRL rules for failure prediction and to determine the cause of the potential failure.

3.4 Systematic literature review conclusions

The development of this systematic literature review aimed to discuss the main issues related to machine learning and reasoning for predictive maintenance in the context of Industry 4.0. We discussed the concepts and technologies applied in this area. We also presented the challenges faced in its application in the real world. The review focused on identifying architectures or frameworks proposals that use reasoning as part of their decision process. Achieving reasoning in this context is possible using ML techniques and models or through the adoption of ontologies. The study was limited to predictive maintenance of cyber-physical systems, not including related works that apply predictive maintenance in other contexts, such as predicting software failures.

Three research questions were defined to guide this systematic literature review. The answer to these questions showed that the need for data integration across the company is a topic of interest because it impacts on the overall business performance. Collecting data from a piece of equipment and giving contextual information, semantically improving that data, and providing meaning through the use of information from various sources received attention too. Moreover, using formal methods, although not yet deeply investigated, is also an important matter. For instance, the use of ontologies in the context of predictive maintenance appears as a tool applied for data standardization, aiding in the interoperability of systems, and consequently collaborating with the integration of the company's information as a whole.

Machine learning models applied to predictive maintenance also received attention from the scientific community recently. In this sense, we identified that there is a large number of different models being proposed and applied in this field. Nevertheless, the results of the systematic literature review showed that no algorithm is capable of dealing with all existing scenarios in a company. In many cases, an expert needs to tune the ML models to meet the characteristics of the equipment.

This work showed that predictive maintenance is a hot topic in the context of Industry 4.0. We conclude that because the papers that bring novelty to the field are concentrated in the years 2018 and 2019. Many relevant works are available so far. However, there is still room to deal with several challenges in this field. Taking into account the achieved results, we envision the necessity of implementing the theoretical frameworks found in the literature in real industrial environments. This implementation would allow a more precise evaluation of their effectiveness through metrics such as cost reduction and time spent on the maintenance task. Challenges related to big data are also of interest because predictive maintenance is a field that relies on large amounts of data. Therefore issues like scalability, latency, and data security deserve further investigation.

4 ELFPM FRAMEWORK

This chapter will present the ELFpm, a framework model for PdM. In the two sections presented in this chapter, the first sections presents the architecture of the proposed framework, detailing the functioning of each layer and the sub-processes of each of these layers. The second section presents the elaboration of the ontology following the Grüninger and Fox methodology.

4.1 Architecture of ELFpm

To facilitate the understanding, high level view of ELFPM architecture is presented in Figure 13. In this framework, the use of an ontology combined with prediction algorithms makes it possible to process data obtained by installed sensors in machines or equipment. Then these data will be used to perform predictions of failures. The predictions made will be available through an application so that it is possible to make queries and monitoring the information by a responsible agent.

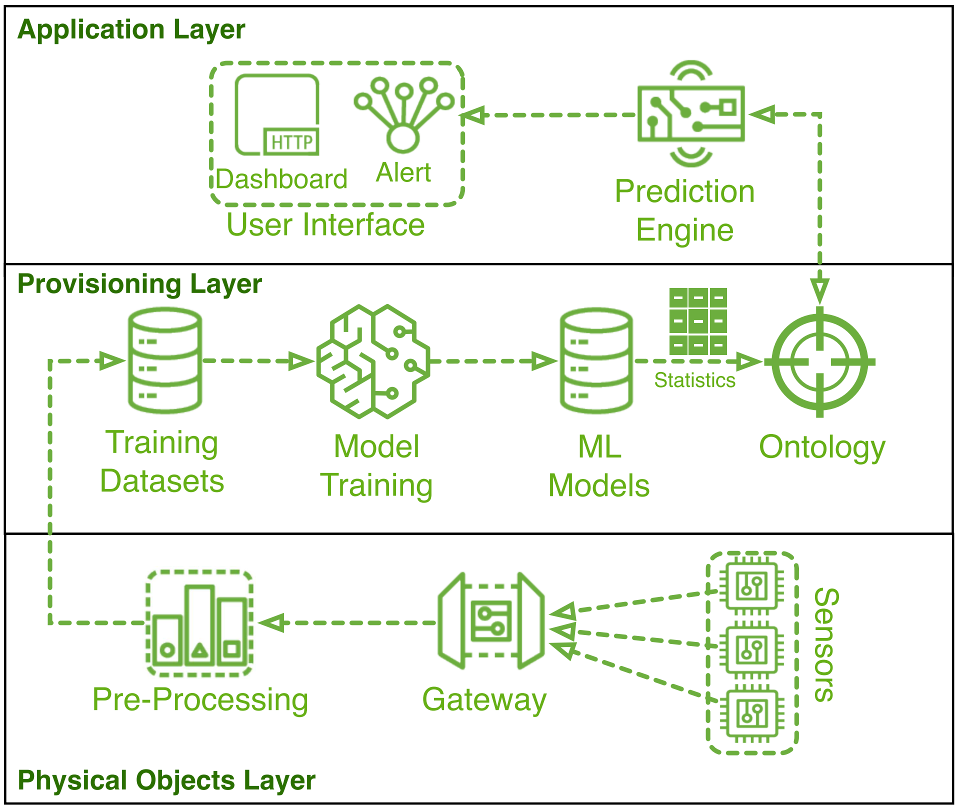


Figure 13: Framework overview.

The proposal is divided into tree layers that operate simultaneously and independently, the Physical Objects Layer, the Provisioning Layer, and the Application Layer. In the following subsections, each layer will be presented and its components will be detailed.

4.1.1 Physical Objects Layer

The first layer, called Physical Objects, is the layer closest to the physical world, being responsible for collecting and processing the data, so that it is sent to the next layer. The physical objects layer consists of three steps, the first step being the collection of data through sensors connected to industry equipment, and different types of sensors can be connected to the framework, according to the desired use case. These sensors can be spread over several points of a machine, location, or company, collecting a diverse range of data at an unknown volume. With this in mind, a second step of this layer was proposed with the responsibility of performing this task of centralizing the receipt of data.

The second step is to receive this data, a task performed by a central IoT gateway. This gateway can receive data from the most diverse possible sources, so some characteristic problems in data collection such as bandwidth, latency, among others discussed in the section 3.3.1 may appear, and in this step these problems must be addressed to mitigate any negative impact on framework.

Finally, we have the pre-processing step, responsible for dealing with other challenges, among the tasks performed by this step is the application of algorithms to reduce the dimensionality of the large amount of data received, dealing with the quality of the data as eventual gaps in the data. The data resulting from this pre-processing is then made available for use in the upper layers of the framework.

4.1.2 Provisioning Layer

The resulting data from the Physical Objects Layer is processed by the Provisioning Layer. This layer is composed of four components. The first one is a database designed to store training datasets. This component is updated as new sensor data arrives and provides data to the machine learning models to be trained. Virtually any kind of machine learning technique is supported by ELFPM, but the parameters of the model must be provided.

In the next step, the model is trained and the resulting model is stored in a database. Each machine learning algorithm results in an n number of models due to the fact of using hyper-parameters in training. The number of models generated is directly linked to the number of parameter variations used in each learning algorithm.This machine learning models database provides statistics regarding the trained models to an ontology that is designed to help selecting the best machine learning model for a given scenario.

To help achieve the goal of creating a fault prediction framework that can be used in a given scenario, it is necessary to have an agent that will perform the initial configuration of the ontology, this configuration of the ontology is done by a Ontology agent as presented in Figure 14. It is at the configuration step that the necessary configuration, will be informed, such as, which component and process will be monitored and the machine learning statistic will be stored. These configuration will serve as the basis for the creation of the ontology.

In the second step, machine learning algorithms based on the test database will be trained. Initially, for validation of the framework, only supervised learning algorithms will be used. The machine learning algorithms that will be used in this first moment will be the random forest (RF), support vector machine (SVM), artificial neural networks (ANNs). This supervised machine learning algorithms processes are widely used in industry (GE et al., 2017). Beside that, as test, will also be used recurrent neural network (RNN) that was been user in bearing fault prediction (GUO et al., 2017). The results obtained by the algorithm will be saved in ontology.

The last step is to generate the ontology file by associating the component information, sensors, processes and training bases with the information of the machine learning statistics to create the ontology.

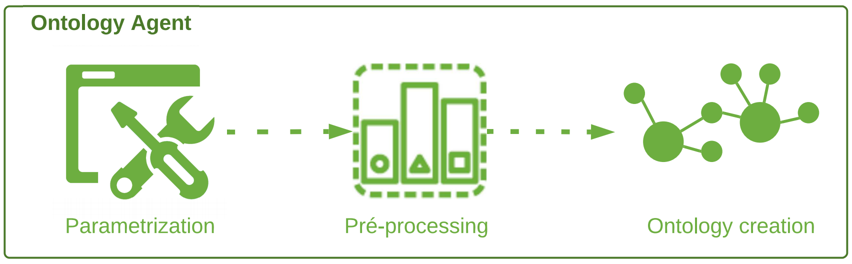


Figure 14: Ontology Agent.

4.1.3 Application Layer of the framework

ELFPM ontology outputs a list of candidate machine learning models by means of queries applied to the ontology using SPARQL[1], that are input to the Prediction Engine component, which is part of the Application Layer of the framework. This component runs an optimization algorithm to select the model that fits better for the given use case. Different optimization algorithms can be deployed in this phase and take into account various metrics, such as the error rate and the training time of the models. In this work, the decision making algorithms called TOPSIS and the ε-Greedy algorithm will be used. Together with these two algorithms, an index will be used which is obtained in a simpler way for comparison.

We call these indexes by the name ELFIndex and they work as follows. As will be presented in section 4.2.3, every trained model will result in some information that within the ontology was called statistical data. As statistical data we have the RMSE, the prediction time, and the training time.

The RMSE is the result of the error obtained in training the model, in general this value represents the distance estimated by the model in relation to the obtained value. The lower the

[1]https://www.w3.org/TR/rdf-sparql-query/

RMSE, the closer to the real value and more accurate the trained model is.

Regarding the prediction and training times, both are stored in seconds. The training time is the resultant measurement in seconds that a given model took to perform the training using the training base of the IMS dataset, while the prediction time refers to the time in seconds that the model already trained required to perform the prediction using the test database of the IMS dataset. All models were trained with the same test and training database.

With the stored statistical data, a combination of RMSE values and times obtained through a multiplication is made to arrive at ELFIndex. As these values are all available through a consultation in the ontology, the use of ELFIndex has the advantage of being a simple solution for selecting the best learning model to be chosen by framework when compared to TOPSIS and ε-Greedy algorithms. After the selection of the model, the Prediction Engine runs the model to predict the behavior of the equipment under analysis.

Before any failure occurs it is necessary that there is some kind of warning or alert to be issued in advance so that actions can be taken in order to prevent any failure from occurring and mitigate any consequences that may arise from one of that failure.

To trigger this warning, it is necessary to define a threshold value. If the threshold value is reached, the Alert component of the ELFpm framework will be responsible for triggering the alerts with the estimated time remaining for the failure to occur.

There are in the literature several models built to define a Threshold for time series, which is the scenario covered in this work. In the review made by Tong (2011), some of these probabilistic and statistical models developed in the 30 years prior to the publication of the work were presented and which threshold principle probably continued to make valuable contributions in the analysis of time series over the next 30 years.

Other works address the definition of threshold in order to follow some type of equipment manual or pre-established standard, following that line Kiangala e Wang (2018) used the severity levels defined by ISO IS2372 to separate the severity level into classes and thus trigger some type of warning when a certain already known and pre-established value is reached.

We can have the definition of value for the threshold made through the prior knowledge of specialists, who when applying a visual analysis of the data defines the values to be followed. In the work presented by Ali, Patel e Breslin (2019) threshold values were defined based on analysis of historical data and through knowledge of the team's existing domain or jointly through as presented by Schmidt e Wang (2018c) where the definition of various threshold value, then perform a comparison test of the results obtained to choose the most suitable. In this work, the definition of the threshold value for the dataset used in the experiments took place through the use of a moving average.

The final component of ELFPM framework is the User Interface. This component is responsible for the presentation of the data in a friendly and easy way to the users understand. The user interface is currently able to send alerts to the users whenever the threshold is met. Moreover, the component is also designed to display summarized information in a dashboard.

In Figure 15 we present a sequence diagram using the UML language to help understand the creation of the Ontology. The agent *OntologyCreatingAgent* would receive the training dataset and all the information about it, such as the source sensor and which component it belongs to. After receiving the initial data, the ontology will be generated, at which point the models will be generated for each learning algorithm, resulting in values such as the RMSE for each model. The learning algorithm, its parameters and its prediction statistics will be used for the creation of the entities in the ontology.

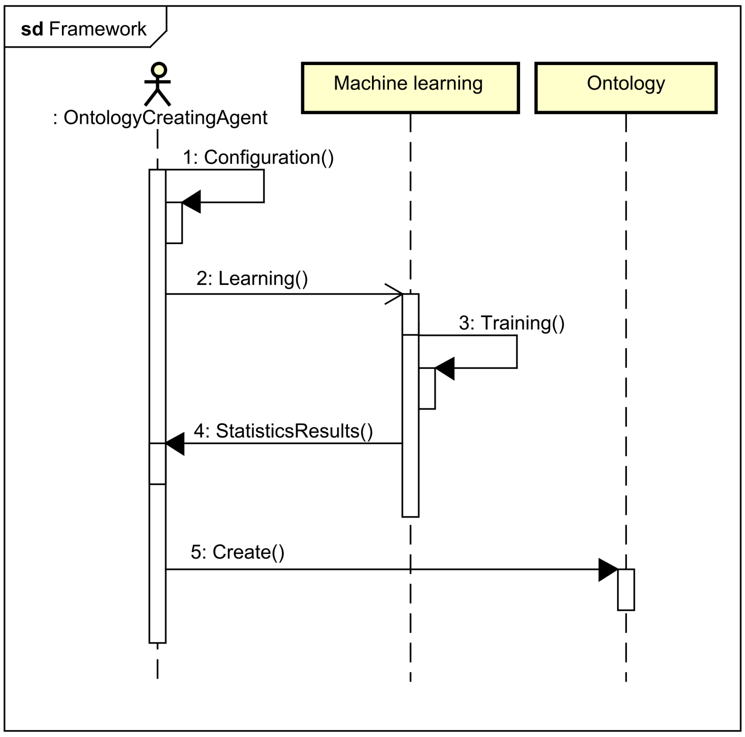


Figure 15: Sequence diagram of the ontology creation.

In Figure 16, is present the data flow from the collection, analysis and notification, demonstrating the general functioning of the framework after ontology was created.

4.2 Ontology Modeling

The development of the ontology was based on the Gruninger e Fox (1995) methodology. The Gruninger and Fox methodology divides the development process into six stages, namely: identify motivating scenarios, elaborate informal competency questions, specify the terminology using first order logic, elaborate formal competence question, formal axioms, and completeness theorem.

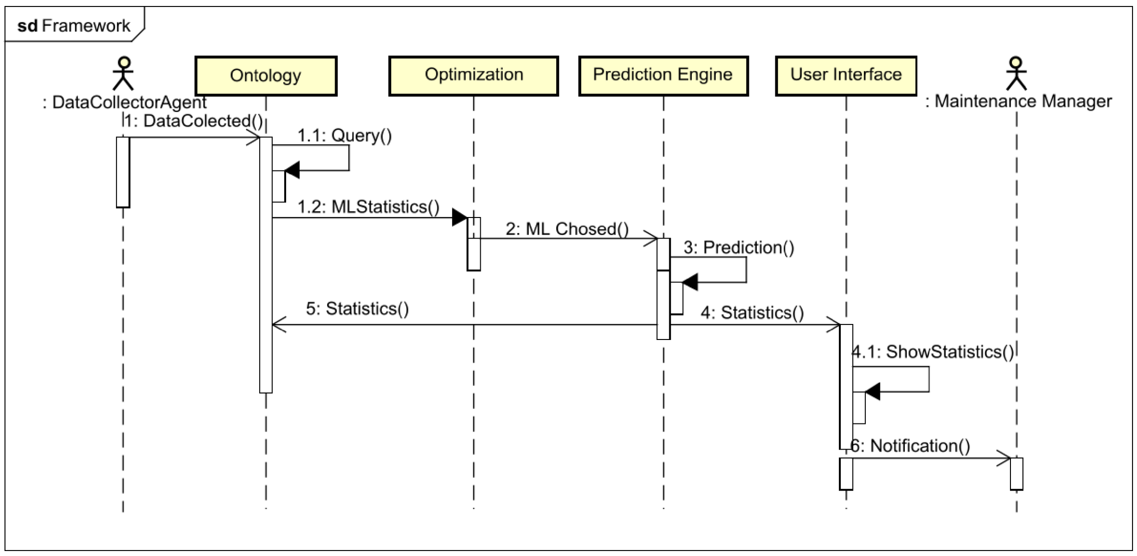


Figure 16: Sequence diagram of the data collection, analysis, and notification.

4.2.1 Motivating scenarios

Within the scenario of Industry 4.0, it is possible through the sensing of a machine to analyze its behavior(LEE; KAO; YANG, 2014). This analysis of behavior occurs through the extraction of characteristics such as vibration, temperature and sound. These characteristics are obtained by sensing the machine and help in understanding the normal behavior of the system being monitored. Thus it, is possible to identify behavioral changes and predict possible failures through the use of artificial intelligence methods (LEE; KAO; YANG, 2014). The construction of the ontology will be guided by the following scenario:
"Module of a framework that will use information from the sensing of a machine to suggest the input attributes of the machine learning module of the framework. In addition, this module should suggest which machine learning algorithms are recommended for each scenario."

4.2.2 Elaborate informal competency questions

Although the ontology has the capacity to answer several questions, only attending to the previously presented motivation scenario, there is only two question that the ontology must answer.

1. "Given a piece of data captured by a sensor in a component, which machine learning algorithms should be used to predict its failure?"

2. "Given machine learning algorithms, which parameters should be used to training the model?"

4.2.3 Specify the terminology using first order logic

Using the previous steps as the basis for formalizing the concepts of the ontology, the main classes of the ontology and the properties of objects and data properties were created as can be seen in Figure 17. To simplify the visualization, UML[2] notation was used.

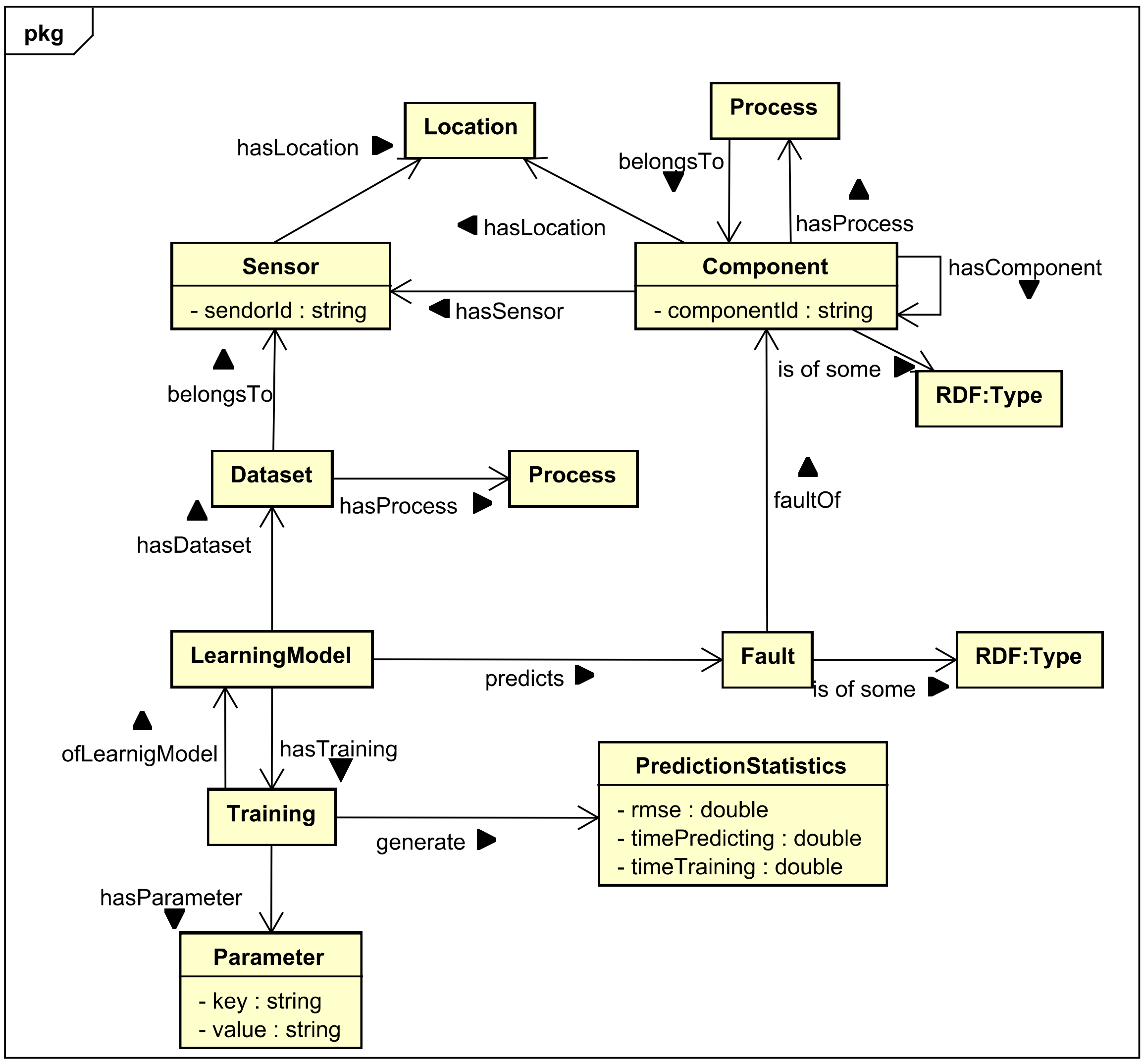


Figure 17: Classes of the ontology.

The *Component* class represents the object of the physical world being observed, this object can be a machine or one of its components. The observation of a component will occur through a sensing, represented by the *Sensor* class.

Process represents both the process that a component can perform and which process this component is performing at the time of reading a sensor, just as the *Location* class is intended to represent the location of both a component and a sensor.

The *LearningModel* class represents the machine learning techniques that can be used by the framework to generate the *Training*. Each execution will have different parameters repre-

[2]http://www.uml.org/what-is-uml.htm

sented by the class *Parameters* and prediction statistics as a result, represented by the *PredictionStatistics* class. The data generated by the machine learning algorithms will serve to classify the occurrence of a fault, represented by the class *Fault*.

Finally, the *DataSet* class represented the data that will serve as training for the machine learning algorithms.

4.2.4 Formal axioms

Formal axioms should define the conditions for competency question to be complete (GRUNINGER; FOX, 1995). The agent responsible for creating the ontology will execute the learning algorithms several times with parameters that will change with each execution. The executions will result in a statistic class that will be used to define the best method for each scenario. Later it will be possible to recreate the ideal learning model for each situation. The fallowing code 4.1 represents the creation of the classes using the SPARQL 1.1 Update [3] language.

```
INSERT INTO GRAPH
https://ELFpm/ontologia
{
$execucao a minhaOntologia:Training.
   $prediction_statistics_rf a
    ELFpm:PredictionStatistics.
    $execucao ELFpm:generate $predictionstatistics_rf.
    $execucao ELFpm:hasParameter $parameter_rf.
    execucao ELFpm:ofLearnigModel $randonForest.
    $predictionstatistics_rf ELFpm:rmse 0.01
    $predictionstatistics_rf ELFpm:r2_score 0.89
    $predictionstatistics_rf ELFpm:mae 0.002
    $parameter_rf ELFpm:key "max_depth"
    $parameter_rf ELFpm:value 10
}
FROM https://ELFpm/ontologia
WHERE
{
    $randonForest a ?type .
    $type rdfs:subClassOf ELFpm:RandonForest
    BIND(IRI("ELFpm:execucao1")) AS $execucao).
    BIND(UUID()) AS $predictionstatistics_rf) .
}
```

Listing 4.1: Class creation example using SPARQL

The creation of the *Training* class takes place from the execution of the a Random Forest

[3]https://www.w3.org/TR/sparql11-update

algorithm, where the parameter "max_depth" with value 10 resulting in a *Parameter* class. The result of the execution is represented in the *PredictionStatistics* class with a RMSE error as a value.

With the insertion of the information of each execution it will be possible to make queries to answer the formal competencies questions that will be presented in the next section.

4.2.5 Formal competence question

Using the axioms of the ontology, the questions presented in section 4.2.2 are solved using the formal competition questions. Thus, through the query performed with SPARQL and presented in the code 4.2 it is possible to present the best machine learning technique for the scenario created in the previous section.

```
PREFIX rdf:http://www.w3.org/1999/02/22-rdf-syntax-ns#
PREFIX owl:http://www.w3.org/2002/07/owl#
PREFIX rdfs:http://www.w3.org/2000/01/rdf-schema#
PREFIX xsd:http://www.w3.org/2001/XMLSchema#
PREFIX ELFpm:http://www.semanticweb.org/ELFpm#
SELECT ?component ?dataset ?process ?lm  ?rmse ?timeTraining ?
   timePredicting
WHERE {
    ?component ELFpm:hasSensor ?sensor.
    ?dataset ELFpm:belongTo ?sensor.
    ?dataset ELFpm:hasProcess ?process.
    ?lm ELFpm:hasDataset ?dataset.
    ?lm ELFpm:hasTraining ?tn.
    ?tn ELFpm:generate ?ps.
    ?ps  ELFpm:rmse ?rmse.
    ?ps  ELFpm:rmse ?timeTraining.
    ?ps  ELFpm:rmse ?timePredicting.
}
ORDER BY (?rmse)
```

Listing 4.2: SPARQL query example to select the model with the lowest RMSE

In this scenario, where there is a bearing being monitored by a vibration sensor, the result is show in the Table 8 where the best machine learning technique for predicting faults using RMSE value is the Random Forest.

As already described in section 4.1.3, the lowest value obtained for the RMSE is not necessarily the best choice, since it is necessary to take into account the variables of training time of the models and prediction time. Thus, in the next section of results, the use of ontology in conjunction with the techniques employed to select the ideal model will be demonstrated.

Component	Learning model	RMSE
bearing	random forest	0.00208
bearing	random forest	0.00252
...	...	...
bearing	Long short-term memory	0.11300

Table 8: Ontology query return

4.2.6 Completeness theorem

After the formalization of competence questions, it is necessary to define the conditions under which the solutions to the questions are complete. This is the basis of the completeness theorems for ontology.
In order for the competition question 1 to be complete, it is necessary to create the classes *Component*, *Sensor*, *Database* and *MachineLearning* with their relationships. In addition, it is necessary to create the *PredictionStatistics* class and its relation with the *LearningModel* class, this is done through the *hasTraining* relationship of the *LearningModel* class with the *Training* class and the *generate* relationship of the *Training* class with the *PredictionStatistics* class. In order for question 2 to be complete, it is necessary, besides the creation of the classes of question 1, the creation of the relationship class *hasParameter* with the of the *Training* and the created class *Parameters*. In this way the questions can be solved by the proposed methodology.

However, we have other statistics besides the RMSE value that each machine learning model generates, such as time to train the model and time to generate the prediction. These are information that the ontology is able to store and make available to the Prediction Engine component to make the choice of the best model in relation to the prediction time and the error generated.

5 RESULTS

At first, this section will present the database used, the necessary transformations applied to the database, the machine learning algorithms used and the best results obtained from each method for then present the results obtained in order to find the model that appears to be the ideal choice to perform the prediction.

5.1 Methodology

All tests with the dataset and algorithms were performed in the *Jupyter Notebook*[1] environment, using Python version 3.7.3. In order to perform the loading of the datasets, the *Python Data Analysis Library*[2] was used and the graphics display were enhanced with the library *matplotlib*[3]. For the networks the tests with the LSTM and MLP were used the *Keras*[4] library, for the RF the *scikit-learnig*[5] library was used, and for ARIMA the python library *pmdarima*[6] was used.

The IMS dataset it was the database used to train the models, this dataset has been widely used in the literature for diagnosis and prognostics and consists of a test-to-failure in a rolling of an AC motor made by University of Cincinnati and released in 2014 (QIU et al., 2019). The AC motor was used to maintain a shaft at a constant rotational speed. Four bearings were installed on the shaft and each received vibration sensors (two sensors in each bearing for the x-axes and y-axes in the dataset 1 and one sensor for each bearing in dataset 2 and 3).

Each data capture was performed for a period of one second with a frequency of 20 kHz and each collection occurred within a ten minute interval (except in dataset 1 where the first 43 files were taken every 5 minutes).

At the end of the test the dataset 1 an inner race defect occurred in bearing 3 and a roller element defect in bearing 4, the dataset 2 an outer race failure occurred in bearing 1 and the dataset 3 a outer race failure occurred in bearing 3.

The first task for the experiment was to create and use a reduced version from the original dataset. To reduce the size of the data, a number of techniques were used to achieve a size that is computationally viable for the experiment. The technique choose is to use only the data from bearing 3 that has a fault in addition to using a sampling of the data. The last 75 data were collected for every 15 observations recorded and always making use of the last observation collected.

Taking into account that the Bearing dataset has a total of 2155 observations, as a result we had a dataset with 10850 vibration data.

[1]https://jupyter.org/
[2]https://pandas.pydata.org/
[3]https://matplotlib.org/
[4]https://keras.io/
[5]https://scikit-learn.org/stable/
[6]https://pypi.org/project/pmdarima/

With the experiment dataset created a decomposition was applied to determine whether the data show some form of general trend or seasonal trend. The seasonal_decompose function was used for this task. As seen in Figure 18 a trend is visible at the end of collection.

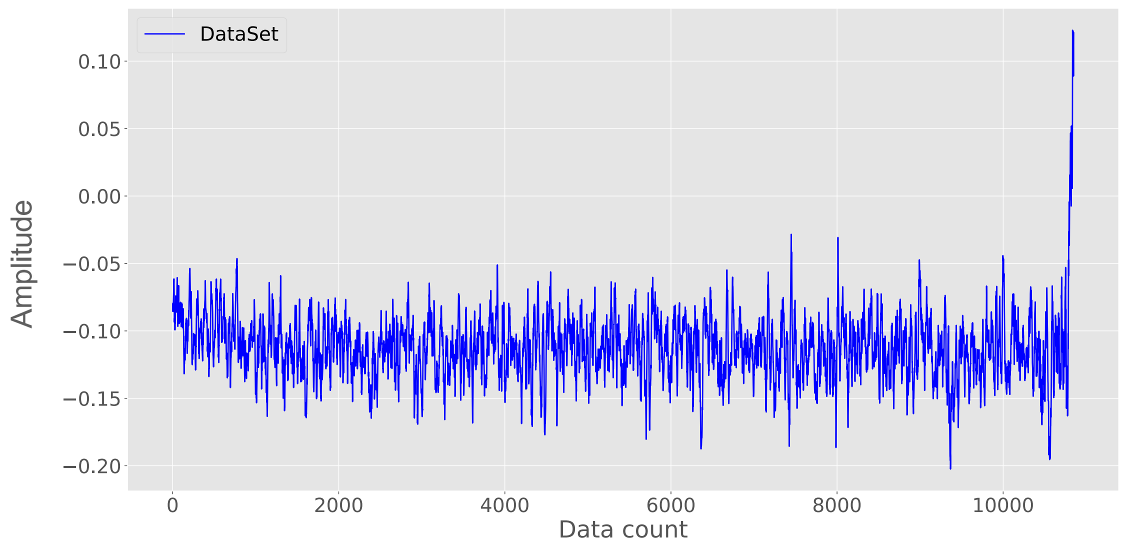


Figure 18: Vibration trend over the time

5.2 Machine learning model training

Using the trend data, a training base and a test base were created to evaluate which machine learning algorithms presents the best performance for predicting the time series. In all tests, a training database was created using 66% of the original data and a test database with 33% of the original data. The metric selected to evaluate the technique with the best performance was the lowest value reached for root mean squared error (RMSE).

To perform all the tasks required for prediction, Python 3.5 was chosen because they have all the necessary libraries.

5.2.1 Random Forest

In order to estimate the best parameters of the linear regression model of the random forest, the GridSearchCV () method was used, with a combination of parameters where 10, 673, 1336 and 2000 number of trees (n_estimators), *auto* and *sqrt* as the number of features when looking for the best split (max_features), 1, 25, 50 and None for the maximum depth of the tree, 2, 5 and 10 for the minimum number of samples required to split an internal node, 1, 2 and 4 as the minimum number of samples required to be at a leaf node (min_samples_leaf) and finally true and false to bootstrap. The combination of all parameters resulted in a total of 576 training's performed.

The best and worst value for the RMSE obtained is shown in the Table 9 below for the test base. It is possible to observe that the value that stands out most for the reduction of the error was the parameter responsible for the number of trees to be created.

n_estimators	max_features	min_samples_leaf	RMSE
1336	auto	10	0.0137
10	sqrt	2	0.0240

Table 9: Parameters comparison using RF

Through Figure 19 presents the prediction result for bearing fault for training and testing data. It is possible to identify that despite maintaining a good result during the course of the dataset, at the end of the graph, where the failure behavior occurs, the prediction of the algorithm represented in green was not able to follow the trend of the dataset, represented in blue

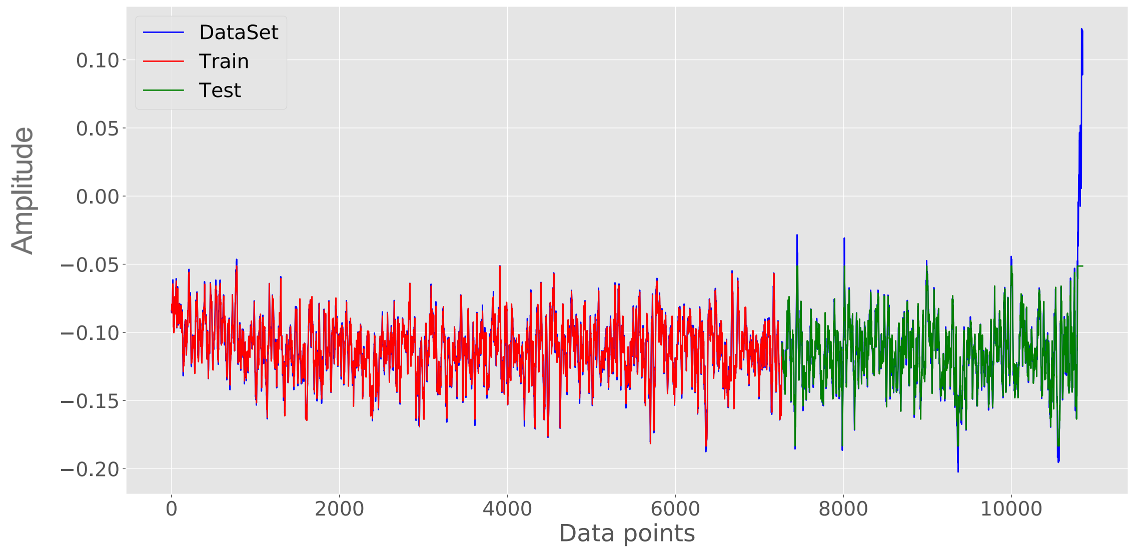


Figure 19: Performance of best RF model for training (red) and test (green).

5.2.2 LSTM

Different from the RF, that to find the best set of parameters to train the model was used the GridSearchCV() method, for the LSTM a specific method was developed that uses a predefined parameter set to test all possible combinations to the model using the Keras LSTM RNN. The table shows the best and worst RMSE obtained in the tests with the test base. We created a model for each combination of parameters, where 20, 40 and 80 used for number of neurons, 100 and 500 for the number of epochs and 20, 50 and 100 for batch size. As show in Table 10, the results showed that increase neuron number and epoch number does not necessarily result in a decrease in RMSE.

Figure 20 presents the prediction result for bearing fault for training and testing data. Unlike the result obtained by the RF, here we have the green prediction line in the graph following the

layer	neuron	epoch	batch size	RMSE
2	(80, 40)	100	100	0.0057
1	(20)	100	100	0.0058
2	(20, 10)	50	20	0.0058
1	(40)	500	20	0.0064
1	(20)	500	20	0.0067
1	(80)	500	20	0.0068

Table 10: Parameters comparison using LSTM

failure trend, even at the end of the dataset. This behavior at the end of the dataset justifies the smaller RMSE when compared to the RF.

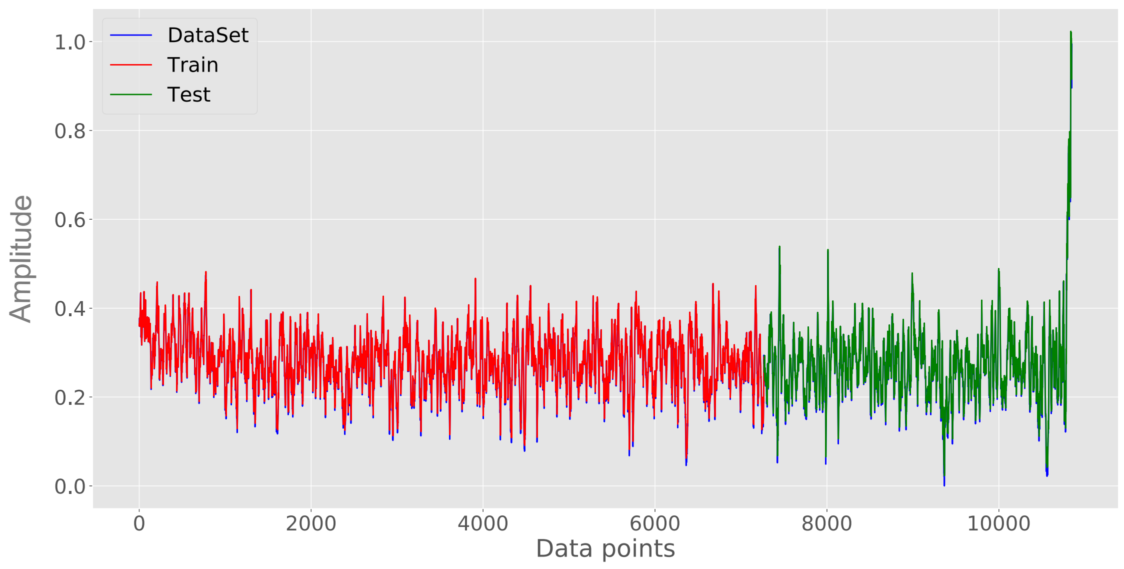


Figure 20: Performance of best LSTM model for training (red) and test (green).

5.2.3 Multilayer Perceptron

Using the Keras Sequential method for MLP, a specific method was developed that uses a predefined parameter set to test all possible combinations to the model. Each combination is trained with a training base and validated with the test base, the RMSE is saved to then define the best parameters for the bearing dataset. To find the best model we tested all combinations of parameters for the model with one layer of neurons with 10, 20, 40 and 80 neurons and with two layers of neurons ([20,10], [20,20], [40, 20], [40, 20], [80, 40] and [80, 80]). Each neuron parameter was trained with 100 and 500 epoch, 20, 50 and 100 batch size and as activation *tanh* and *relu*.

As the possible number of combinations is very large, Table 11 shows the three best and the three worst results obtained. As can be observed the smaller error obtained was with the model trained with the largest number of neurons available with activator *tanh* in contrast with the worst results obtained with the smallest number of possible neurons and activator *relu*.

layer	neuron	epoch	activation	batch size	RMSE
2	(80, 80)	500	tanh	20	0.0050
2	(20, 20)	100	tanh	100	0.0051
2	(80, 40)	500	tanh	20	0.0051
2	(20, 10)	500	tanh	100	0.0222
1	(40)	500	relu	20	0.0294
1	(50)	500	relu	50	0.0305

Table 11: Parameters comparison using MLP

Figure 21 presents the best prediction result for bearing fault for training and testing data. With results similar to that obtained by the LSTM, we can see that the red prediction lines followed the blue line, which represents the dataset, just as it had already been seen with the RF and LSTM, already the test base prediction line obtained results similar only to LSTM, which can be confirmed by the RMSE values shown in the Table 11 and 10.

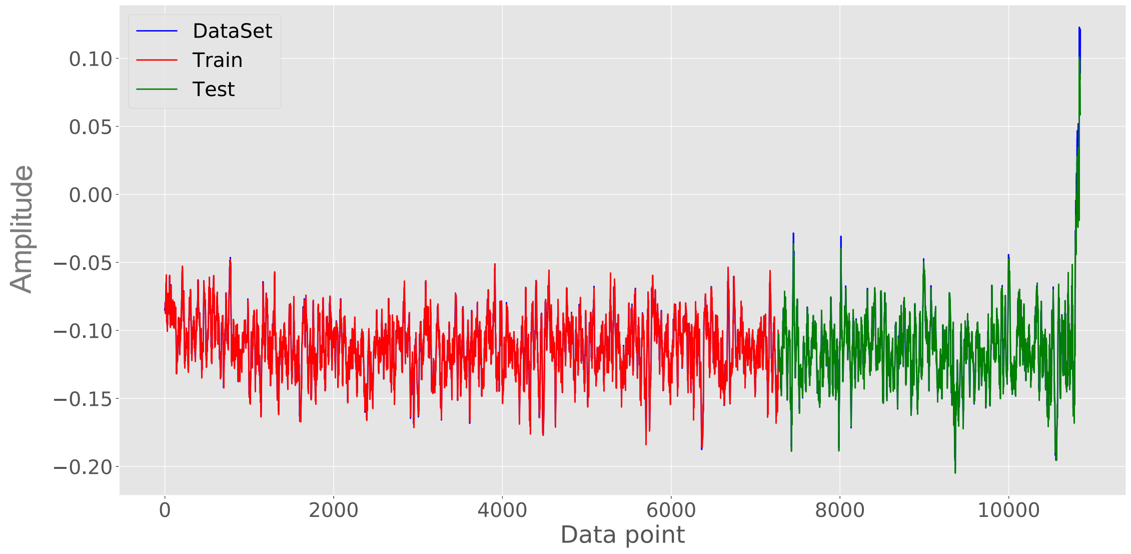


Figure 21: Performance of best MLP model for training (red) and test (green).

5.2.4 ARIMA

As in previous models, a set of parameters was used to train the ARIMA in order to find the best model with the lowest RMSE. Since as can be seen in Figure 22 the best autocorrelation values remained before the first ten lag so the parameters used for p was 0, 2, 4, and 8, and values of 0, 1, and 2 for d and q where p is the order of the auto regressive model, d is the degree of differentiation and q is the order of the moving average model (ZHANG, 2003).

The smallest RMSE achieved was 0.0038 and it was reached with the model constructed from the values 4, 2, 0 for p, d and q respectively. In Figure 23 the complete dataset in the

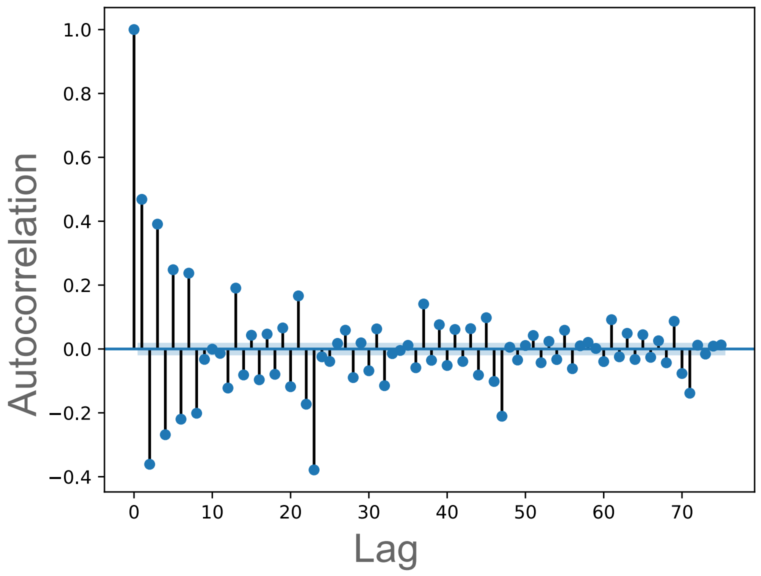


Figure 22: Autocorrelation of the dataset.

blue line is presented with the test base being used to test the prediction and demonstrated in the green line.

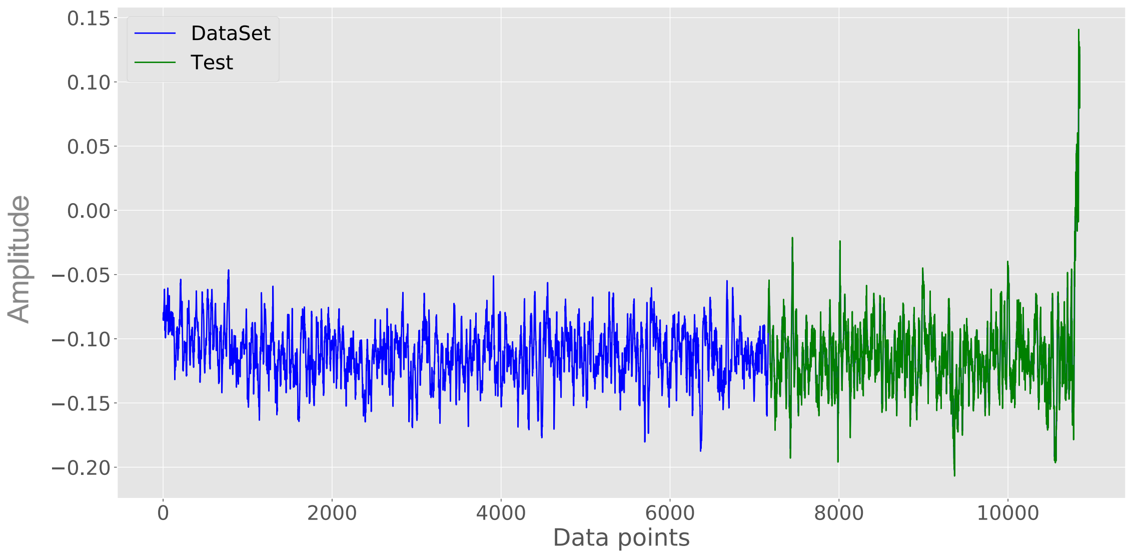


Figure 23: Performance of best ARIMA model for test (green).

5.2.5 Machine learning model training results

The result obtained by each technique presented in this work is stored in the ontology created previously. Thus, when a similar scenario that has been already analyzed by this work arises, the ontology will already be able to answer which is the most appropriate machine learning method. Taking into account just the results of the RMSE presented in table 12, for the bearing dataset the method to be chosen must be ARIMA.

Method	RMSE
MLP	0.0050
RNN	0.0058
RF	0.0137
ARIMA	0.0038

Table 12: Best value for RMSE in each algorithm

However, the graph presented by Figure 24 presents some characteristics about the parameters used for training. As the hyper parameterization technique was applied in this work , each learning algorithm was trained with a variation of the input parameters, thus generating different training models for each algorithm. Each model had its RMSE value collected and the average RMSE for each algorithm is shown in the horizontal line of the graph, while the vertical line represents the standard deviation obtained through the different models generated in each learning algorithm. Thus, we can say that although an ARIMA model obtained the least error, ARIMA showed a greater dispersion when compared to the other algorithm, which means that the parameterization process has a great influence on the RMSE value. On the other hand, the LSTM obtained a significantly smaller distribution, in which case other analyzes can be made.

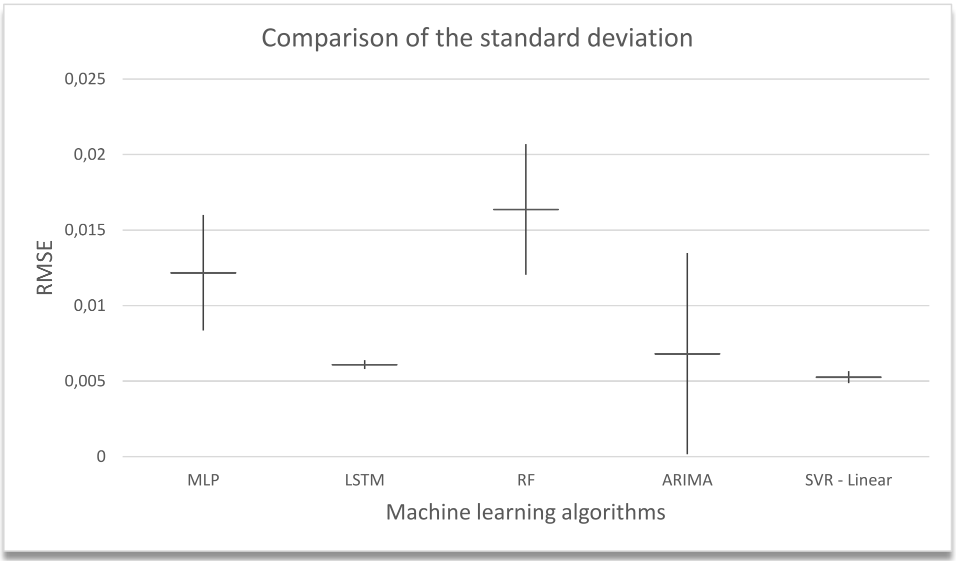


Figure 24: Comparison of the standard derivation.

If other metrics are considered for the definition of the best model, such as training time, the parameterization that obtains the lowest RMSE value for the ARIMA model can have a high training time. In the LSTM due to the low distribution of values, the training time can be applied as a tie-break criterion. Since this choice is a result of the inference made by the ontology, making changes in these criteria as well as other changes and expansions are possible to realize.

After training the models and collecting the generated statistical data, it is necessary to determine the best machine learning algorithms for prediction taking into account the error and times of each model. This step of generating the models is part of the Reasonig Layer of the framework. The results obtained will be used by the Prediction Engine Component layer to define the best possible model.

As each algorithm has several parameters that can be informed at the time of training due to the use of hyper-parametrization, a large number of models can be generated. With the algorithms used in this work, more than 2000 thousand models were generated. Due to computational limitations it was necessary to reduce the number of models that will be used in the tests. This reduction was made through the use of the TOPSIS technique with equal weights for the time and error variable. The TOPSIS ranking was used to select the models, so for each algorithm used, the 15 best models were selected according to the ranking generated by TOPSIS. Finally, 60 models were selected, which will then be used to carry out the tests. This initial step was necessary to allow the execution of the tests, since it would be impracticable to use all the more than 2000 thousand trained models

With the 60 models selected, and already having in hand the results of the RMSE, the training time of the model, and the time for prediction, some techniques and algorithms were then applied to find the best possible model, optimizing the times and the RMSE. The tests were performed using ε-Greedy Algorithm, TOPSIS, and the indexes created in this work. The following methodology was applied in the execution of the tests for each optimization technique:

1. Select the best model according to each selection method:
    - For the method with the indexes created in this work, choose the model that has the lowest value for the index;
    - For the ε-Greedy Algorithm, run the selection twice, one using the shortest RMSE and the other using the shortest time;
    - For the TOPSIS method choose the best model according to the weights used.
2. Run the prediction with the chosen model;
3. Obtain the RMSE values and time to perform the prediction;
4. Update the RMSE and the time by averaging the values already stored with the new values obtained;
5. Repeat the process 100 times.

5.2.6 ε-Greedy Algorithm Results

Two strategies were applied in the tests performed using the ε-Greedy Algorithm. In the first, the results obtained considered the model that has the lowest RMSE within the models previously selected as the best option. The results of the errors obtained in this case can be seen

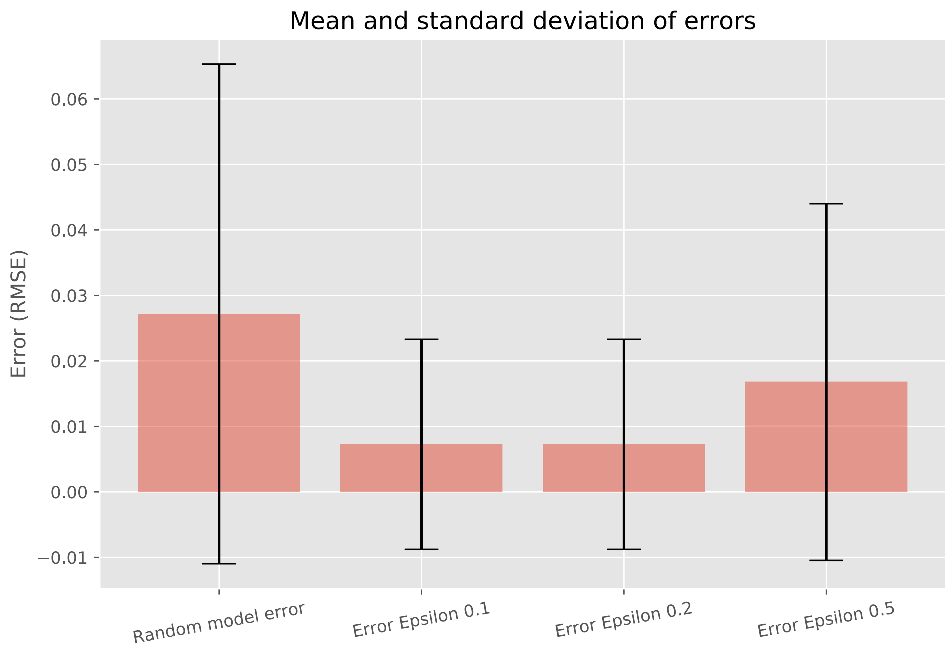


Figure 25: Model error selected by ε-Greedy Algorithm based on the smallest error

in Figure 25, where the red bars represent the average of the RMSE obtained for each test round. On the black line, the standard obtained is shown. Both the mean and the standard deviation were calculated after turning the model choice 100 times through the lowest RMSE or through a random choice according to the εparameterization.

The first bar shows the average of errors and standard deviation obtained through a random selection of the models. In the columns to the side we have the errors obtained through the choices made by ε-Greedy Algorithm, where εwas parameterized with the values of 0.1, 0.2, and 0.5 respectively. It is possible to observe a tendency to increase the average of RMSE as the randomness from εincreases. Such a trend may indicate that, even during the 100 executions carried out, there is a low possibility of finding a model with a better result than that carried out in the first choice.

The same pattern occurs in relation to the prediction execution time during the test round. Figure 26 shows the average of the times obtained to execute the selected models based on the smallest error. Following the pattern presented above, as the possibility of choosing another model other than the one with the least error increases, the longer the average time for training. In addition, another unwanted effect of a high εis a high standard deviation, which indicates that although the model choices have been varied, none of them resulted in a better model than the one initially selected.

The second strategy used in the application of ε-Greedy Algorithm was to use the model with less time, with this parameter of choice different from the one applied previously, the same tests applied were performed using the RMSE as a selection variable. In this new scenario, the error values obtained were slightly higher than those obtained by the first strategy used, as shown in figure 27. However, the pattern of increase in the RMSE as the value of εincreases has

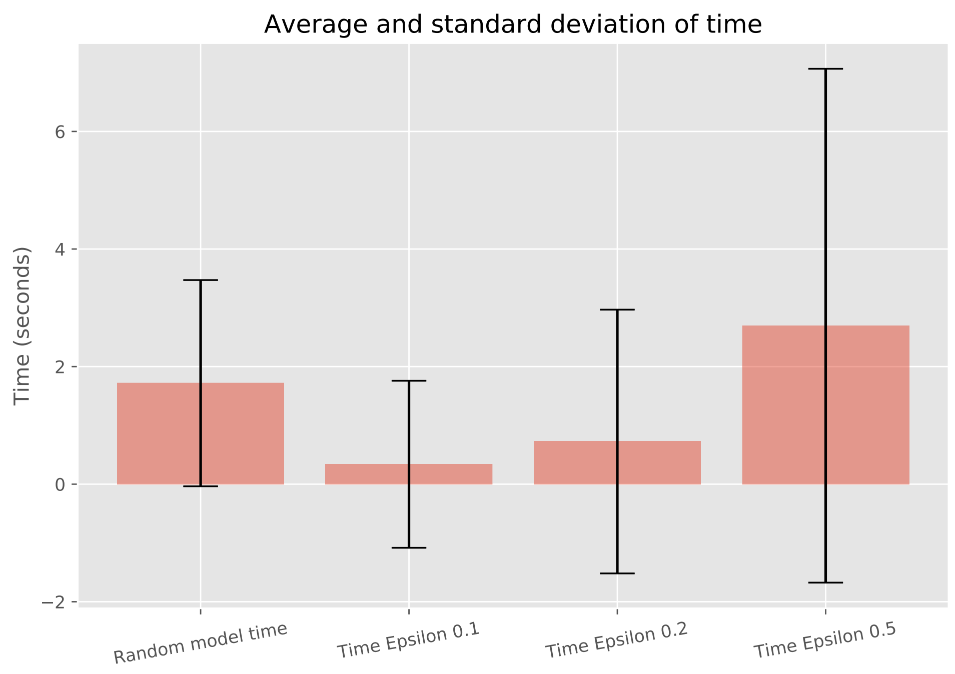


Figure 26: Model time selected by ε-Greedy Algorithm based on the smallest error

been maintained, which shows that there is a difficulty for better models to initially appear, in addition to causing a mean increase in error as a side effect. Another advantage of a smaller εis the standard deviation, significantly lower than the values obtained by the larger ε.

The same can be seen when we analyze the prediction time to determine the best model. Although Figure 28 shows a slightly shorter time than the one obtained in the tests, with the error as the parameter of choice, the pattern of time increase is maintained according to the variability of the model choices caused by a greater value ε.

5.2.6.1 TOPSIS Results

The tests carried out with TOPSIS were designed with three different scenarios in mind. The first scenario is to evaluate the models that give the same weight to the RMSE values and the prediction time displayed for each model. No second weight test is changed, giving more importance to time to the detriment of RMSE in a value of 75% by weight for a time variable and 25% of import for a RMSE variable. Finally, as the last test scenario, the weights were inverted, the residence time with 25% of the weight and the RMSE with the remaining 75% of the weight. As explained earlier, in each scenario elaborated, model choices were performed 100 times. Each time a model was chosen, its prediction time and the RMSE were shown, given the possibility of changes in the choice of the best model during the execution of 100 interactions.

The results obtained and presented in Figure 29 indicate a value for the RMSE significantly lower than those generated by the models chosen at random. As expected, the lowest value for

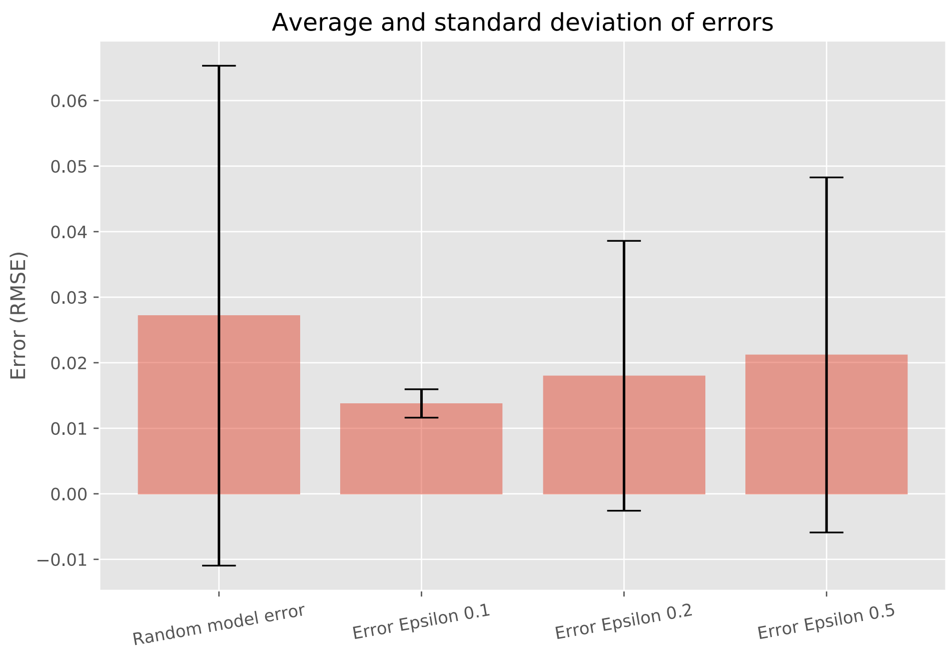


Figure 27: Model error selected by ε-Greedy Algorithm based on the smallest time

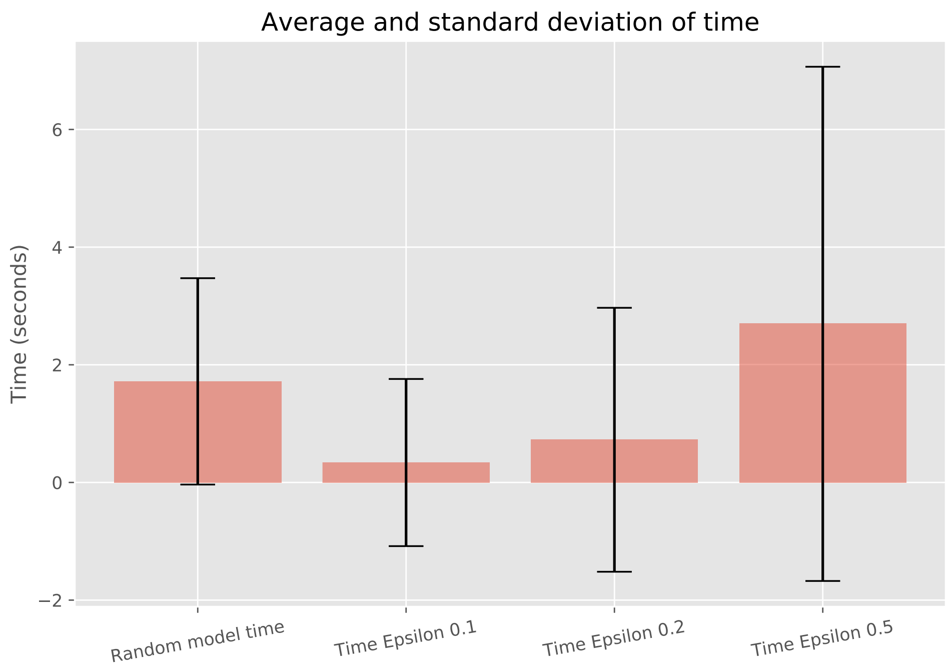


Figure 28: Model time selected by ε-Greedy Algorithm based on the smallest time

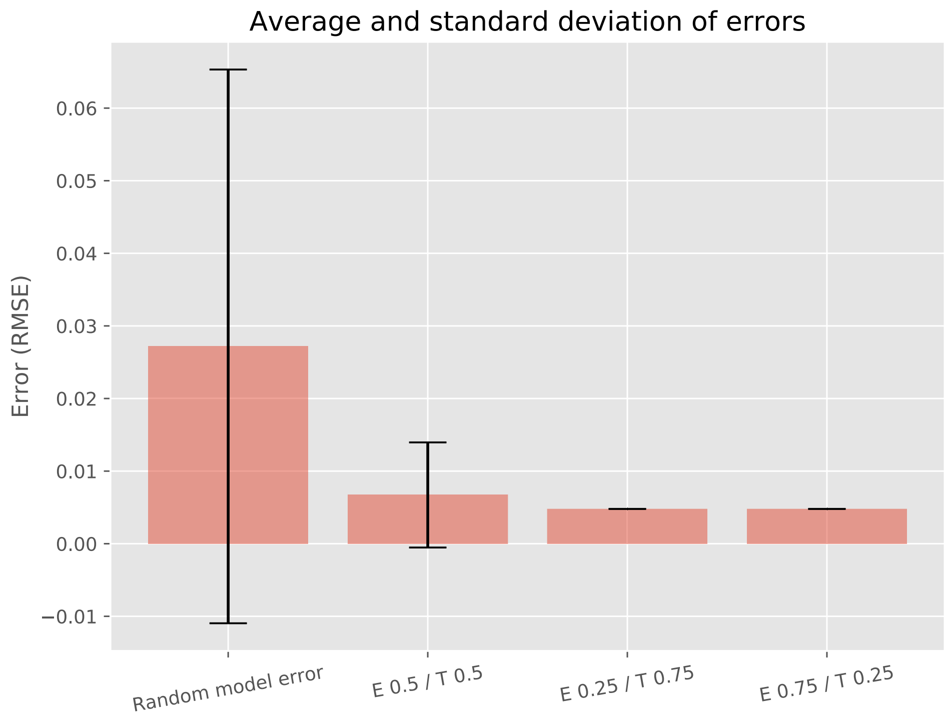


Figure 29: Model error selected by TOPSIS

the RMSE obtained was in the scenario where the weight of the error is most relevant, what is interesting to note is the fact that changes in weights giving greater importance to the prediction time of the models did not bring an increase in RMSE, as might be expected.

When we look at Figure 30 we can see that there is an increase in the prediction time. As expected, the worst time average was obtained when the time variable receives the least weight within the TOPSIS execution. However, all the results obtained have higher values than those observed by a random choice. It is interesting to note that although we achieved a significantly lower average of RMSE using the TOPSIS method when compared to a random choice, the same cannot be said with the average time spent to make the predictions. Here the time of the random models had a performance with significantly better values, reaching means and a standard deviation lower than TOPSIS.

5.2.7 ELFIndex Results

Having in hand the results obtained by applying the ε-Greedy Algorithm and the TOPSIS method, these were then compared to the results obtained by the index created in this work. Thus, in Figure 31, the lowest averages of RMSE obtained in each selection technique were selected. For the ε-Greedy Algorithm the lowest value of RMSE obtained was through the use of εwith a value of 0.1, already in the tests applying the TOPSIS technique, we have as the best model the one that presents a weight of 0.75 error and 0.25 for time. The average of the RMSE obtained was used for comparison using the lowest index resulting from the multiplication of

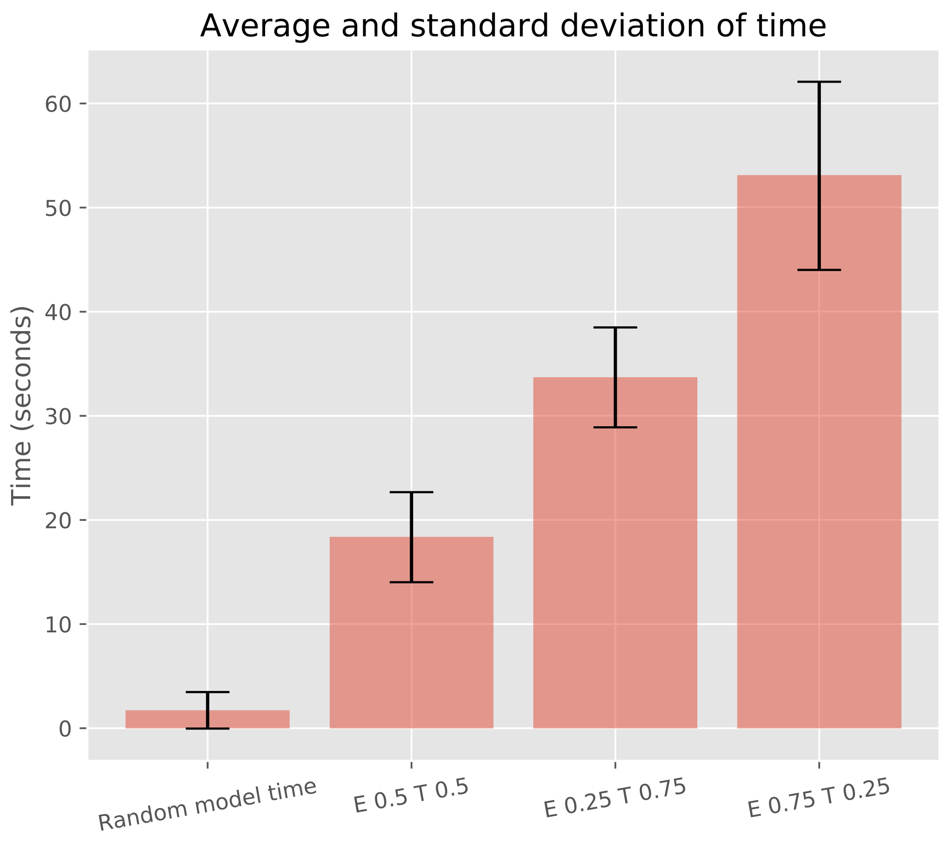


Figure 30: Model time selected by TOPSIS

the RMSE by the training time, the multiplication of the RMSE by the prediction time and, finally, the multiplication of the RMSE by the prediction and training times.

Index	p
Training Time X Prediction X Error	0.00050954
Training Time X Error	0.00050954
Prediction Time X Error	0.00000002

Table 13: Confidence value for the proposed indexes

To ensure that the averages are statistically different from the average obtained through the random choice of models, a t-student test was performed. The values obtained were within the confidence interval of 0.5% for all index as can be seen in the Table 13.

When observing the average RMSE obtained for each selection method, all obtained better results than a simple random choice. In addition, the standard deviation resulting from random choice is also significantly higher, which is expected when it comes to random choices. The ε-Greedy despite having an average of the RMSE similar or even better than the other selection methods, had a standard deviation with a higher value, resulting from the use of ε, which by bringing a randomness value and a variation of model choices generate more dispersed RMSE values. The other selection methods have a standard deviation of zero, which indicates that a model chosen at first because it was considered the best at that moment remained until the end of the 100 runs. In conclusion, with the RMSE as a measure of selection of the best model, the chosen one would be the TOPSIS technique.

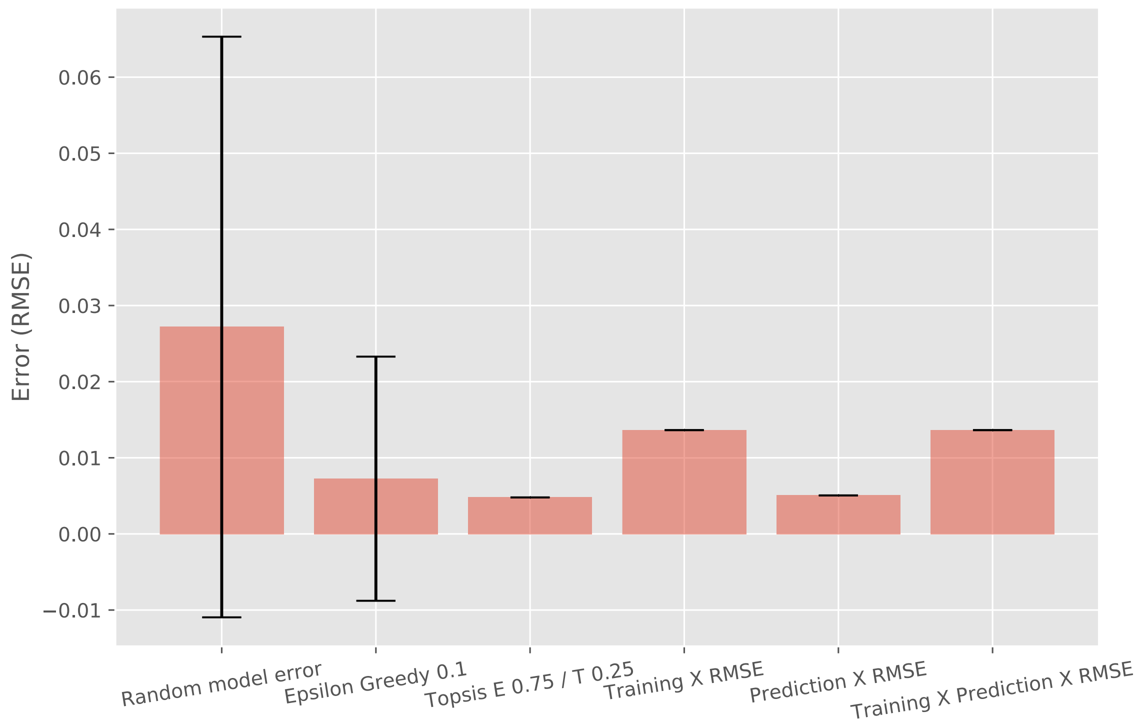


Figure 31: Average and standard deviation of the best by error

However, when the time taken for the prediction is taken into account, this perception of TOPSIS as the best choice can be reconsidered. In Figure 32 it is evidenced that the shortest time was obtained using the indexes, which despite being relatively simple to obtain, requiring little computational cost, end up appearing as a good option to help in choosing the best model.

5.2.8 Computational Cost

In this section, the computational costs of the methods for selecting the best algorithms will be presented. This analysis is part of the definition of the computational cost that each selection method applied in this work has. The algorithm complexity metric was used to measure efficiency in the execution time of the algorithms. This analysis was done for the TOPSIS, ε-Greedy algorithm and for the indexes created in this work.

Taking into account that we have a n equal to 60 and we have the TOPSIS complexity divided into four stages we can calculate the cost in $O(n^2)$ + $O(n)$ + $O(n)$ = (60^2) + 60 + 60. As a final result the value of 3720. The result obtained from the complexity of ε-Greedy with the value of $O(n \log n)$ equal (60 log 60). As a final time result the value of 106.69.

The indexes have the lowest computational cost of all, with a value of $O(1)$. This low computational cost is due to the fact that it is a simple multiplication.

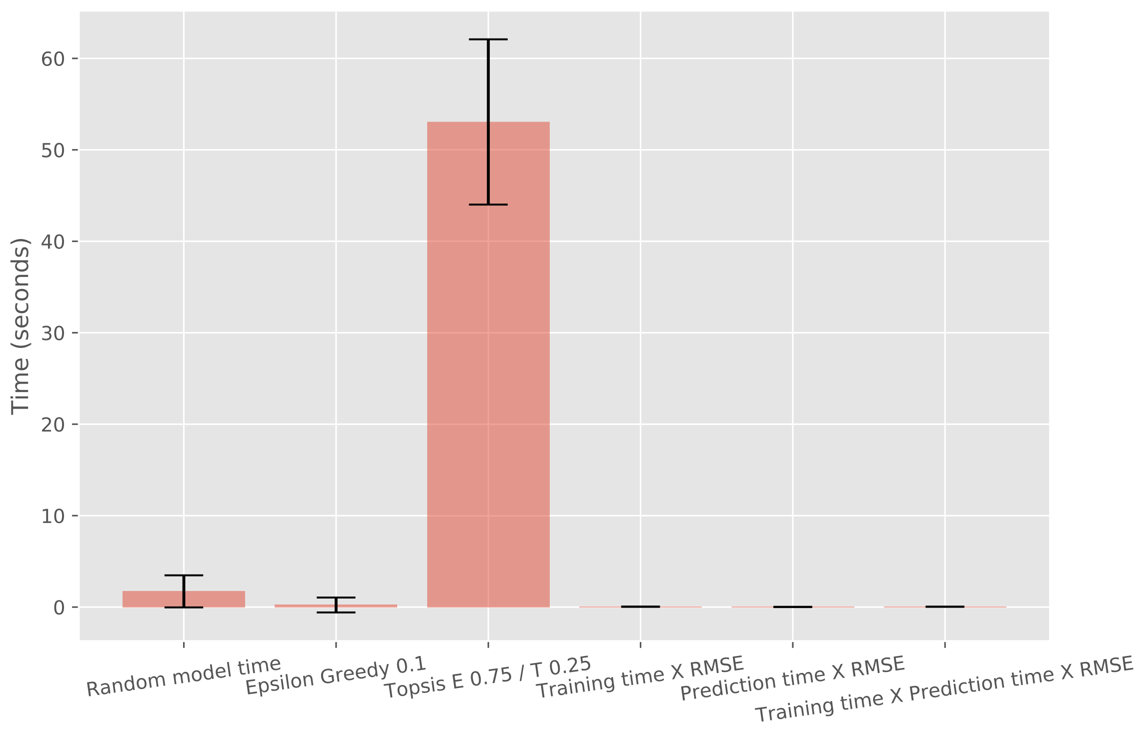


Figure 32: Average and standard deviation of the best by times

5.2.9 Selection of the best model

When analyzing the results obtained, we have among the indexes presented the index that applies a multiplication of the RMSE with the prediction time as the one that results in the lowest error and time values. In addition, when evaluating the computational cost of each optimization model, we again have the indexes as the lowest costs, proving to be the best option among those endorsed in this work to select the ideal machine learning model to perform the failure prediction.

It is possible within the proposed framework to obtain the ELFIndex by executing a simple query in the ELFpm ontology, as shown in code 5.1. In this case, the query obtained the ARIMA model with the parameters of p, d, and q with the values of 2, 0, 0 respectively. The selected model obtained an RMSE value of 0.0050, a value greater than the best result obtained by ARIMA, as can be seen in the Table 12. This higher RMSE value occurs precisely because the measurement of time is taken into account when choosing the model.

```
PREFIX rdf:http://www.w3.org/1999/02/22-rdf-syntax-ns #
PREFIX owl:http://www.w3.org/2002/07/owl #
PREFIX rdfs:http://www.w3.org/2000/01/rdf-schema #
PREFIX xsd:http://www.w3.org/2001/XMLSchema #
PREFIX ELFpm:http://www.semanticweb.org/ELFpm #
SELECT ?component ?dataset ?process ?lm   (?timeTraining * ?timePredicting
    AS ?index )
WHERE  {
    ?component ELFpm:hasSensor ?sensor.
    ?dataset ELFpm:belongTo ?sensor.
    ?dataset ELFpm:hasProcess ?process.
    ?lm ELFpm:hasDataset ?dataset.
    ?lm ELFpm:hasTraining ?tn.
    ?tn ELFpm:generate ?ps.
    ?ps  ELFpm:rmse ?rmse.
    ?ps  ELFpm:rmse ?timeTraining.
    ?ps  ELFpm:rmse ?timePredicting.
 }
ORDER BY ASC  (?index)  LIMIT 1
```

Listing 5.1: Best model selection example using SPARQL

5.2.10 Threshold definition

To find the ideal threshold, the moving average was applied within the dataset with several windows to identify the most appropriate value. The moving average was applied with a window of 38, 75, 113 and 150 points, the result of the application of the moving average can be seen in figure 33, where the larger the window applied, the more behaved is the line within the graph. For better visualization the last 400 points of the dataset were plotted.

As the dataset used shows an abrupt tendency of failure in the last two collections, windows of the moving average were chosen, ranging from half of a collection to the use of the points read in two different collections. The line defining the threshold was applied using the last value obtained from the moving average of the dataset. This was done in this scenario where the display of the fault signal grows abruptly at the end of the dataset. So in Figure 34 the Threshold value using a window of 150 points (values of two collections of 75 points each) is not viable, and the windows of 113 points (values of one and a half collection), 75 points (one collection) or 38 points (half a collection).

If the Threshold value is defined as the 113 point moving average value, we have an anticipation of at least three days. This anticipation occurs exactly at the point where the graph begins to show a slight increase in vibration.

To better visualize the Threshold obtained by applying a window of 75 and 38 points, Figure

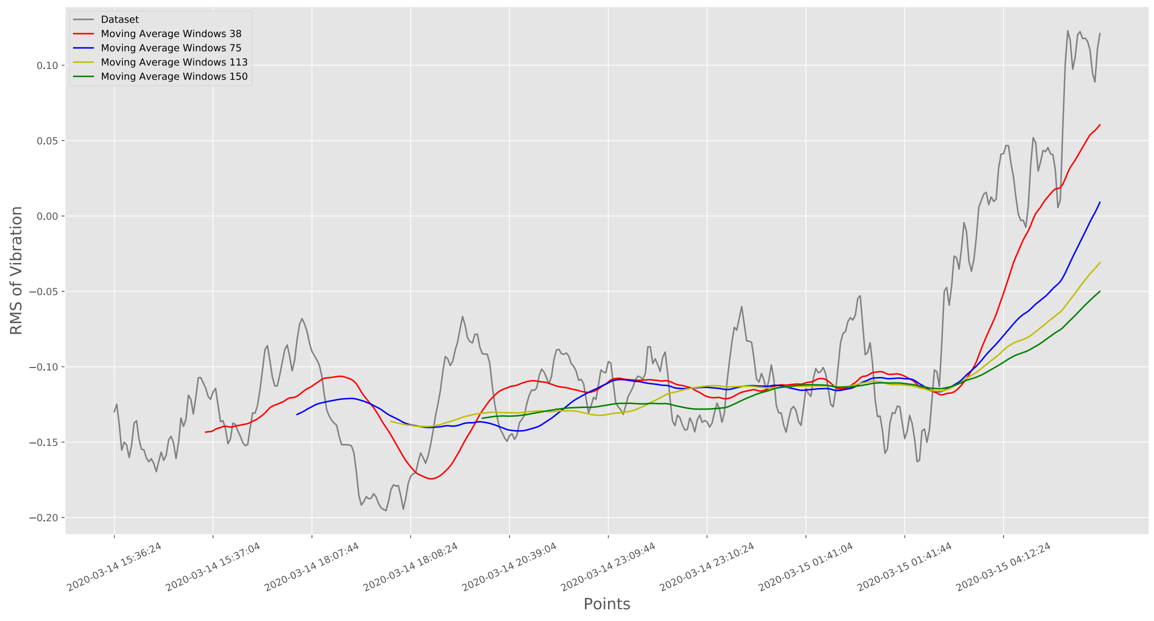


Figure 33: Mean average of the last 400 points.

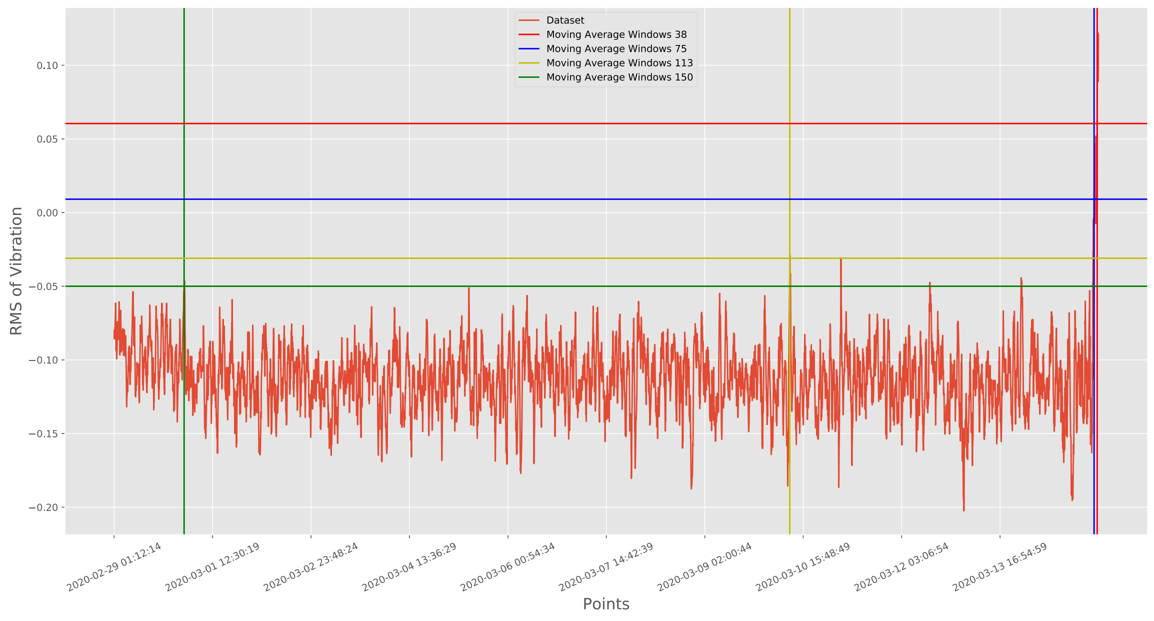


Figure 34: Threshold value in the dataset.

35 displays only the last 400 points collected in the dataset. In this way the ideal Threshold is the value obtained by applying the window with 75 points, where the failure warning occurs at least three hours in advance. This relatively short time is due to the fault behavior of the dataset, which, as can be seen in the graphs presented, shows a fault signature only in the last vibration data collections made.

Therefore, when using the moving average, we can define a threshold value. with that it is possible to anticipate a failure before it occurs. As already discussed, we can define a threshold value according to the analyzed scenarios. In the dataset used in this work, an adequate thresh-

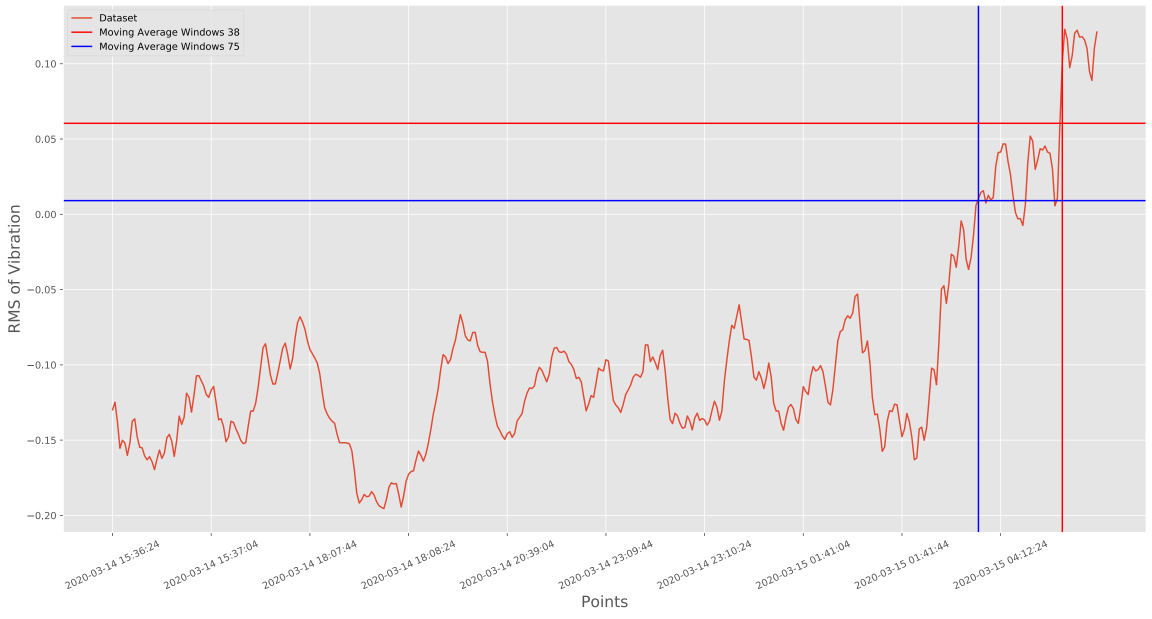


Figure 35: Threshold values in the last 400 points collected

old value was able to anticipate a failure by 3 hours. If the dataset has a more behavioral failure trend, with a failure that evolves steadily over several weeks, this threshold value can anticipate a failure by days. In the dataset used, the failure behavior occurs suddenly at the end of the machine's last life, which left a 3-hour interval for actions to be taken to mitigate the effects of the failure in the industry process.

6 CONCLUSION AND FUTURE WORK

Industry 4.0 is receiving great interest from various industry sectors. Developing and implementing technologies such as PdM can mean a competitive edge. With the advancement of IoT and the availability of a large volume of data together with increased processing capacity and lower costs, it is possible to develop new systems that are capable of increasing productive capacity and reducing the idleness of the industrial park with actions more assertive maintenance tasks.

Focusing on this, this work proposed the ELFpm, a framework for predicting failures in Industry 4.0. As part of ELFpm an ontology to assist in managing the various scenarios involving a manufacturing is designed. We evaluate the framework and the feasibility of its implementation considering a use case based on a widely accepted failure dataset.

This work relies on specific machine learning techniques. These techniques can be expanded and added as new modules of the framework. The learning algorithms used in this work were trained and the RMSE values obtained by each one were analyzed together with the time needed to perform the tests and the time consumed to obtain the predictions from the use of new input data. The analysis of these data aimed to define the best algorithm for the evaluation scenario.

To determine the choice of the best model, the TOPSIS and ε-Greedy algorithms were compared with the ELFpm Index, a calculation methodology proposed in this work. As a result, ELFpm Index, although simpler to execute, selected algorithms more efficiently than TOPSIS and ε-Greedy considering simultaneously RMSE, training time, and prediction time.

In addition to choosing the ideal learning model, ELFpm must be able to warn of possible failures in advance. Thinking about it, the definition of a Threshold was necessary. In this work, a moving average was applied with several different parameters. The best parameterization obtained in this work was able to predict a failure three hours before the occurrence. This result is interesting, considering that the behavior change of the dataset occurs abruptly in the last collections.

It is important to highlight the use of ontology in ELFpm, which allows the storage of the framework parameters. This allows the insertion of new learning models as well as the expansion of the use case scenarios what provides flexibility on the implementation of ELFpm. Another gain related to the usage of an ontology is to understand the scenario in which the framework is being applied, since its parameterization tends to require an expert user.

However, it is important to highlight some of the challenges faced in the development and application of ELFpm. First, within the use of machine learning models, we can highlight the possible delay in the training. This delay can occur according to the algorithm being trained and the parameters used in this training. Another point to be highlighted concerns the dataset used for training purposes, which may eventually need to be previously treated, either to format the data in a way that can be used to feed the learning models or to reduce the size of the data and make its use computationally feasible.

These challenges make ELFpm dependent on a specialist to be properly implanted and operated. It is this specialist who will, for example, insert the parameters of the ontology for each scenario in which the framework will be used. Moreover, in many cases, this expert is required to prepare the models training dataset and finally make ELFpm operational.

As future work we can highlight the implementation and testing of ELFpm in other scenarios with new datasets. Besides, new algorithms can be added to the framework in order to validate its ability to adapt to new scenarios. Other implementations that can be developed involve making use of the expansion capacity of the ontology. Currently, the ontology stores statistics resulting from the machine learning algorithmic training, such as the RMSE, the prediction times, and the training times. These are the values used to choose the ideal prediction model to make the predictions in the scenario in which ELFpm is operating. However, there are other statistics that can be considered relevant and that are not currently being used, such as the energy and computational cost for the training of the models in addition to the use of other metrics that can also be applied as the mean absolute error (MAE) and R Squared (R^2), among others.

Finally, we can mention the model used to define the threshold to predict when a failure is going to occur. The definition of a threshold value corresponds to an entire field of research, where several models and methodologies were developed with the purpose of reaching an ideal threshold value. Therefore, testing new models for setting a threshold value can be seen as work that can be explored more deeply in the future.

REFERENCES

ADHIKARI, P.; RAO, H. G.; BUDERATH, D.-I. M. Machine learning based data driven diagnostics & prognostics framework for aircraft predictive maintenance. , 2018.

ALI, M. I.; PATEL, P.; BRESLIN, J. G. Middleware for real-time event detection and predictive analytics in smart manufacturing. In: INTERNATIONAL CONFERENCE ON DISTRIBUTED COMPUTING IN SENSOR SYSTEMS (DCOSS), 2019., 2019. **Anais. . .** . p. 370–376.

ANSARI, F.; GLAWAR, R.; NEMETH, T. Prima: a prescriptive maintenance model for cyber-physical production systems. **International Journal of Computer Integrated Manufacturing**, p. 1–22, 2019.

ANSARI, F.; GLAWAR, R.; SIHN, W. Prescriptive maintenance of cpps by integrating multimodal data with dynamic bayesian networks. In: **Machine learning for cyber physical systems**. p. 1–8.

BALOGH, Z. et al. Reference architecture for a collaborative predictive platform for smart maintenance in manufacturing. In: IEEE 22ND INTERNATIONAL CONFERENCE ON INTELLIGENT ENGINEERING SYSTEMS (INES), 2018., 2018. **Anais. . .** . p. 000299–000304.

BERNARAS, A.; LARESGOITI, I.; CORERA, J. M. Building and reusing ontologies for electrical network applications. In: EUROPEAN CONFERENCE ON ARTIFICIAL INTELLIGENCE, BUDAPEST, HUNGARY, AUGUST 11-16, 1996, PROCEEDINGS, 12., 1996. **Anais. . .** . p. 298–302.

BOUSDEKIS, A. et al. A unified architecture for proactive maintenance in manufacturing enterprises. In: **Enterprise interoperability viii**. p. 307–317.

BOX, G.; JENKINS, G. **Time series analysis**: forecasting and control. (Holden-Day series in time series analysis).

BREIMAN, L. Random forests. **Machine learning**, v. 45, n. 1, p. 5–32, 2001.

BUMBLAUSKAS, D. et al. Smart maintenance decision support systems (smdss) based on corporate big data analytics. **Expert Systems with Applications**, v. 90, p. 303–317, 2017.

CACHADA, A. et al. Maintenance 4.0: intelligent and predictive maintenance system architecture. In: IEEE 23RD INTERNATIONAL CONFERENCE ON EMERGING TECHNOLOGIES AND FACTORY AUTOMATION (ETFA), 2018., 2018. **Anais. . .** . v. 1, p. 139–146.

CAO, Q. et al. Combining chronicle mining and semantics for predictive maintenance in manufacturing processes. , 2019.

CARBERY, C. M.; WOODS, R.; MARSHALL, A. H. A new data analytics framework emphasising pre-processing in learning ai models for complex manufacturing systems. In: **Intelligent computing and internet of things**. p. 169–179.

CARBERY, C. M.; WOODS, R.; MARSHALL, A. H. A bayesian network based learning system for modelling faults in large-scale manufacturing. In: IEEE INTERNATIONAL CONFERENCE ON INDUSTRIAL TECHNOLOGY (ICIT), 2018., 2018. **Anais. . .** p. 1357–1362.

CERQUITELLI, T. et al. A fog computing approach for predictive maintenance. In: INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION SYSTEMS ENGINEERING, 2019. **Anais. . .** p. 139–147.

CHUKWUEKWE, D. O.; GLESNES, T.; SCHJØLBERG, P. Condition monitoring for predictive maintenance–towards systems prognosis within the industrial internet of things. , 2016.

CORMEN, T. H. et al. **Introduction to algorithms**.

COSTA, R. et al. Semantic enrichment of product data supported by machine learning techniques. In: INTERNATIONAL CONFERENCE ON ENGINEERING, TECHNOLOGY AND INNOVATION (ICE/ITMC), 2017., 2017. **Anais. . .** p. 1472–1479.

CRESPO MÁRQUEZ, A.; FUENTE CARMONA, A. de la; ANTOMARIONI, S. A process to implement an artificial neural network and association rules techniques to improve asset performance and energy efficiency. **Energies**, v. 12, n. 18, p. 3454, 2019.

FERNáNDEZ-LóPEZ, M.; GOMEZ-PEREZ, A.; JURISTO, N. Methontology: from ontological art towards ontological engineering. **Engineering Workshop on Ontological Engineering (AAAI97)**, 03 1997.

FERREIRA, L. L. et al. A pilot for proactive maintenance in industry 4.0. In: IEEE 13TH INTERNATIONAL WORKSHOP ON FACTORY COMMUNICATION SYSTEMS (WFCS), 2017., 2017. **Anais. . .** p. 1–9.

GAO, R. et al. Cloud-enabled prognosis for manufacturing. **CIRP annals**, v. 64, n. 2, p. 749–772, 2015.

GARDNER, M. W.; DORLING, S. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. **Atmospheric environment**, v. 32, n. 14-15, p. 2627–2636, 1998.

GATICA, C. P. et al. An industrial analytics approach to predictive maintenance for machinery applications. In: IEEE 21ST INTERNATIONAL CONFERENCE ON EMERGING TECHNOLOGIES AND FACTORY AUTOMATION (ETFA), 2016., 2016. **Anais. . .** p. 1–4.

GE, Z. et al. Data mining and analytics in the process industry: the role of machine learning. **IEEE Access**, v. 5, p. 20590–20616, 2017.

GERS, F. A.; SCHMIDHUBER, J.; CUMMINS, F. A. Learning to forget: continual prediction with lstm. **Neural Computation**, v. 12, p. 2451–2471, 2000.

GLAWAR, R. et al. Conceptual design of an integrated autonomous production control model in association with a prescriptive maintenance model (prima). **Procedia CIRP**, v. 80, p. 482–487, 2019.

GOLIGHTLY, D.; KEFALIDOU, G.; SHARPLES, S. A cross-sector analysis of human and organisational factors in the deployment of data-driven predictive maintenance. **Information Systems and e-Business Management**, v. 16, n. 3, p. 627–648, 2018.

GÓMEZ-PÉREZ, A. Ontological engineering: a state of the art. **Expert Update: Knowledge Based Systems and Applied Artificial Intelligence**, v. 2, n. 3, p. 33–43, 1999.

DOMINGUE, J.; FENSEL, D.; HENDLER, J. A. (Ed.). Ontologies and the semantic web. In: _____. **Handbook of semantic web technologies**. p. 507–579.

GRUBER, T. R. Toward principles for the design of ontologies used for knowledge sharing? **International journal of human-computer studies**, v. 43, n. 5-6, p. 907–928, 1995.

GRUNINGER, M.; FOX, M. S. Methodology for the design and evaluation of ontologies. In: 2016 , 1995. **Anais. . .** .

GUO, L. et al. A recurrent neural network based health indicator for remaining useful life prediction of bearings. **Neurocomputing**, v. 240, p. 98–109, 2017.

HAMDANI; WARDOYO, R. The complexity calculation for group decision making using topsis algorithm. In: AIP CONFERENCE PROCEEDINGS, 2016. **Anais. . .** v. 1755, n. 1, p. 070007.

HASHEMIAN, H. M. State-of-the-art predictive maintenance techniques. **IEEE Transactions on Instrumentation and measurement**, v. 60, n. 1, p. 226–236, 2010.

HEGEDŰS, C.; VARGA, P.; MOLDOVÁN, I. The mantis architecture for proactive maintenance. In: INTERNATIONAL CONFERENCE ON CONTROL, DECISION AND INFORMATION TECHNOLOGIES (CODIT), 2018., 2018. **Anais. . .** p. 719–724.

HORROCKS, I. Ontologies and the semantic web. **Commun. ACM**, v. 51, n. 12, p. 58–67, 2008.

ISSAM, M.; EL MAJD, B. A.; EL GHAZI, H. A new architecture of collaborative vehicles for monitoring fleet health in real-time. In: IEEE INTERNATIONAL CONFERENCE ON TECHNOLOGY MANAGEMENT, OPERATIONS AND DECISIONS (ICTMOD), 2018., 2018. **Anais. . .** p. 309–313.

JAZDI, N. Cyber physical systems in the context of industry 4.0. In: IEEE INTERNATIONAL CONFERENCE ON AUTOMATION, QUALITY AND TESTING, ROBOTICS, 2014., 2014. **Anais. . .** p. 1–4.

KAUR, K. et al. Towards an open-standards based framework for achieving condition-based predictive maintenance. In: INTERNATIONAL CONFERENCE ON THE INTERNET OF THINGS, 8., 2018. **Proceedings. . .** p. 16.

KIANGALA, K. S.; WANG, Z. Initiating predictive maintenance for a conveyor motor in a bottling plant using industry 4.0 concepts. **The International Journal of Advanced Manufacturing Technology**, v. 97, n. 9-12, p. 3251–3271, 2018.

KITCHENHAM, B. Procedures for performing systematic reviews. **Keele, UK, Keele University**, v. 33, n. 2004, p. 1–26, 2004.

KITCHENHAM, B. et al. Systematic literature reviews in software engineering–a tertiary study. **Information and software technology**, v. 52, n. 8, p. 792–805, 2010.

LASI, H. et al. Industry 4.0. **Business & information systems engineering**, v. 6, n. 4, p. 239–242, 2014.

LEE, E. A.; SESHIA, S. A. **Introduction to embedded systems**: a cyber-physical systems approach.

LEE, J.; BAGHERI, B.; KAO, H.-A. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. **SME Manufacturing Letters**, v. 3, 12 2014.

LEE, J.; KAO, H.-A.; YANG, S. Service innovation and smart analytics for industry 4.0 and big data environment. **Procedia Cirp**, v. 16, p. 3–8, 2014.

LI, Z.; WANG, Y.; WANG, K.-S. Intelligent predictive maintenance for fault diagnosis and prognosis in machine centers: industry 4.0 scenario. **Advances in Manufacturing**, v. 5, n. 4, p. 377–387, 2017.

LIAW, A.; WIENER, M. et al. Classification and regression by randomforest. **R news**, v. 2, n. 3, p. 18–22, 2002.

LIU, Z. et al. Industrial ai enabled prognostics for high-speed railway systems. In: IEEE INTERNATIONAL CONFERENCE ON PROGNOSTICS AND HEALTH MANAGEMENT (ICPHM), 2018., 2018. **Anais. . .** p. 1–8.

MALEK, M. Predictive analytics: a shortcut to dependable computing. In: INTERNATIONAL WORKSHOP ON SOFTWARE ENGINEERING FOR RESILIENT SYSTEMS, 2017. **Anais. . .** p. 3–17.

MANNOR, S.; TSITSIKLIS, J. N. The sample complexity of exploration in the multi-armed bandit problem. **Journal of Machine Learning Research**, v. 5, n. Jun, p. 623–648, 2004.

MAY, G. et al. Predictive maintenance platform based on integrated strategies for increased operating life of factories. In: IFIP INTERNATIONAL CONFERENCE ON ADVANCES IN PRODUCTION MANAGEMENT SYSTEMS, 2018. **Anais. . .** p. 279–287.

MOBLEY, R. K. **An introduction to predictive maintenance**.

MONOSTORI, L. Cyber-physical production systems: roots, expectations and r&d challenges. **Procedia Cirp**, v. 17, p. 9–13, 2014.

MOYNE, J.; ISKANDAR, J. Big data analytics for smart manufacturing: case studies in semiconductor manufacturing. **Processes**, v. 5, n. 3, p. 39, 2017.

MUSEN, M. A. et al. The protégé project: a look back and a look forward. **AI matters**, v. 1, n. 4, p. 4, 2015.

NOY, N. F.; MCGUINNESS, D. L. et al. **Ontology development 101**: a guide to creating your first ontology.

NUÑEZ, D. L.; BORSATO, M. Ontoprog: an ontology-based model for implementing prognostics health management in mechanical machines. **Advanced Engineering Informatics**, v. 38, p. 746–759, 2018.

PERES, R. S. et al. Idarts–towards intelligent data analysis and real-time supervision for industry 4.0. **Computers in Industry**, v. 101, p. 138–146, 2018.

QIU, H. et al. Bearing data set. In: 2019. **Anais...**. data retrieved from The Prognostics Data Repository, `https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/`.

RIVAS, A. et al. A predictive maintenance model using recurrent neural networks. In: INTERNATIONAL WORKSHOP ON SOFT COMPUTING MODELS IN INDUSTRIAL AND ENVIRONMENTAL APPLICATIONS, 2019. **Anais...**. p. 261–270.

ROMEO, L. et al. Machine learning-based design support system for the prediction of heterogeneous machine parameters in industry 4.0. **Expert Systems with Applications**, v. 140, p. 112869, 2020.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. **nature**, v. 323, n. 6088, p. 533–536, 1986.

SALA, D. A. et al. Multivariate time series for data-driven endpoint prediction in the basic oxygen furnace. In: IEEE INTERNATIONAL CONFERENCE ON MACHINE LEARNING AND APPLICATIONS (ICMLA), 2018., 2018. **Anais...**. p. 1419–1426.

SALONEN, A.; TABIKH, M. Downtime costing—attitudes in swedish manufacturing industry. In: WORLD CONGRESS ON ENGINEERING ASSET MANAGEMENT (WCEAM 2015), 10., 2016. **Proceedings...**. p. 539–544.

SARAZIN, A. et al. Toward information system architecture to support predictive maintenance approach. In: **Enterprise interoperability viii**. p. 297–306.

SCHMIDT, B.; WANG, L. Predictive maintenance of machine tool linear axes: a case from manufacturing industry. **Procedia Manufacturing**, v. 17, p. 118–125, 2018.

SCHMIDT, B.; WANG, L. Cloud-enhanced predictive maintenance. **The International Journal of Advanced Manufacturing Technology**, v. 99, n. 1-4, p. 5–13, 2018.

SCHMIDT, B.; WANG, L. Predictive maintenance of machine tool linear axes: a case from manufacturing industry. **Procedia Manufacturing**, v. 17, p. 118–125, 2018.

SCHMIDT, B.; WANG, L.; GALAR, D. Semantic framework for predictive maintenance in a cloud environment. **Procedia CIRP**, v. 62, p. 583–588, 2017.

SELCUK, S. Predictive maintenance, its implementation and latest trends. **Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture**, v. 231, n. 9, p. 1670–1679, 2017.

STOJANOVIC, L.; STOJANOVIC, N. Premium: big data platform for enabling self-healing manufacturing. In: INTERNATIONAL CONFERENCE ON ENGINEERING, TECHNOLOGY AND INNOVATION (ICE/ITMC), 2017., 2017. **Anais...**. p. 1501–1508.

STRAUSS, P. et al. Enabling of predictive maintenance in the brownfield through low-cost sensors, an iiot-architecture and machine learning. In: IEEE INTERNATIONAL CONFERENCE ON BIG DATA (BIG DATA), 2018., 2018. **Anais...**. p. 1474–1483.

SYAFRUDIN, M. et al. Performance analysis of iot-based sensor, big data processing, and machine learning model for real-time monitoring system in automotive manufacturing. **Sensors**, v. 18, n. 9, p. 2946, 2018.

TALAMO, C.; PAGANIN, G.; ROTA, F. Industry 4.0 for failure information management within proactive maintenance. In: IOP CONFERENCE SERIES: EARTH AND ENVIRONMENTAL SCIENCE, 2019. **Anais...** v. 296, n. 1, p. 012055.

TONG, H. Threshold models in time series analysis—30 years on. **Statistics and its Interface**, v. 4, n. 2, p. 107–118, 2011.

USCHOLD, M.; KING, J. M. L. Towards a methodology for building ontologies. In: 1995. **Anais...**.

VERMOREL, J.; MOHRI, M. Multi-armed bandit algorithms and empirical evaluation. In: EUROPEAN CONFERENCE ON MACHINE LEARNING, 2005. **Anais...** p. 437–448.

WAN, J. et al. Artificial intelligence for cloud-assisted smart factory. **IEEE Access**, v. 6, p. 55419–55430, 2018.

WANG, H.; RAJ, B. On the origin of deep learning. **arXiv preprint arXiv:1702.07800**, 2017.

WANG, K.; WANG, Y. How ai affects the future predictive maintenance: a primer of deep learning. In: INTERNATIONAL WORKSHOP OF ADVANCED MANUFACTURING AND AUTOMATION, 2017. **Anais...** p. 1–9.

WATKINS, C. J. C. H. Learning from delayed rewards. , 1989.

XU, L. D.; XU, E. L.; LI, L. Industry 4.0: state of the art and future trends. **International Journal of Production Research**, v. 56, n. 8, p. 2941–2962, 2018.

XU, Y. et al. A digital-twin-assisted fault diagnosis using deep transfer learning. **IEEE Access**, v. 7, p. 19990–19999, 2019.

YAN, J. et al. Industrial big data in an industry 4.0 environment: challenges, schemes, and applications for predictive maintenance. **IEEE Access**, v. 5, p. 23484–23491, 2017.

YI, X. et al. Building a network highway for big data: architecture and challenges. **IEEE Network**, v. 28, n. 4, p. 5–13, 2014.

YOON, K.; HWANG, C. L. Topsis (technique for order preference by similarity to ideal solution)–a multiple attribute decision making, w: multiple attribute decision making–methods and applications, a state-of-the-at survey. **a state-of-the-at survey**, p. 128–140, 1981.

YOON, K. P.; HWANG, C.-L. **Multiple attribute decision making**: an introduction. v. 104.

ZENISEK, J. et al. Streaming synthetic time series for simulated condition monitoring. **IFAC-PapersOnLine**, v. 51, n. 11, p. 643–648, 2018.

ZHANG, G. P. Time series forecasting using a hybrid arima and neural network model. **Neurocomputing**, v. 50, p. 159–175, 2003.

ZHANG, W.; YANG, D.; WANG, H. Data-driven methods for predictive maintenance of industrial equipment: a survey. **IEEE Systems Journal**, 2019.

ZHANG, X. et al. A reference framework and overall planning of industrial artificial intelligence (i-ai) for new application scenarios. **The International Journal of Advanced Manufacturing Technology**, v. 101, n. 9-12, p. 2367–2389, 2019.

ZHOU, C.; THAM, C.-K. Graphel: a graph-based ensemble learning method for distributed diagnostics and prognostics in the industrial internet of things. In: IEEE 24TH INTERNATIONAL CONFERENCE ON PARALLEL AND DISTRIBUTED SYSTEMS (ICPADS), 2018., 2018. **Anais. . .** p. 903–909.