UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE TECNOLOGIA E GEOCIÊNCIAS
DEPARTAMENTO DE ENGENHARIA DE PRODUÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE PRODUÇÃO

HANSER STEVEN JIMÉNEZ GONZÁLEZ

A CONTRIBUTION TO MACHINE LEARNING APPLICATIONS IN LOGISTICS AND MAINTENANCE PROBLEMS

Recife
2021

HANSER STEVEN JIMÉNEZ GONZÁLEZ

A CONTRIBUTION TO MACHINE LEARNING APPLICATIONS IN LOGISTICS AND MAINTENANCE PROBLEMS

Thesis presented to the graduate program in Production Engineering of the Federal University of Pernambuco, Technology and Geoscience Center as a partial requirement for obtaining a doctorate in production engineering.
Area of concentration: Management Engineering

Supervisor: Prof. Cristiano Alexandre Virgínio Cavalcante, Ph.D.

Co-Supervisor: Prof. Phuc Do, Ph.D.

Recife

2021

Catalogação na fonte
Bibliotecária Margareth Malta, CRB-4 / 1198

J61c Jiménez González, Hanser Steven.
A contribution to machine learning applications in logistics and maintenance problems / Hanser Steven Jiménez González. - 2021.
97 folhas, il., gráfs., tabs.

Orientador: Prof. Dr. Cristiano Alexandre Virgínio Cavalcante.
Coorientador: Prof. Dr. Phuc Do.

Tese (Doutorado) – Universidade Federal de Pernambuco. CTG. Programa de Pós-Graduação em Engenharia de Produção, 2021.
Inclui Referências e Apêndice.
Texto em inglês.

1. Engenharia de Produção. 2. Aprendizagem profunda. 3. Racionamento de estoque. 4. *Dropshipping*. 5. Aprendizagem profunda por reforço. 6. Sistemas de múltiplos componentes. 7. Manutenção imperfeita. I. Cavalcante, Cristiano Alexandre Virgínio (Orientador). II. Do, Phuc (Coorientador). III. Título.

UFPE

658.5 CDD (22. ed.) BCTG/2022-15

HANSER STEVEN JIMÉNEZ GONZÁLEZ

A CONTRIBUTION TO MACHINE LEARNING APPLICATIONS IN LOGISTICS AND MAINTENANCE PROBLEMS

Thesis presented to the graduate program in Production Engineering of the Federal University of Pernambuco, Technology and Geoscience Center as a partial requirement for obtaining a doctorate in production engineering.
Area of concentration: Management Engineering.

Approved in: 22/12/2021.

EXAMINATION BOARD

_____
Prof. Dr. Cristiano Alexandre Virgínio Cavalcante (Supervisor)
Universidade Federal de Pernambuco

_____
Prof. Phuc Do (Co-Supervisor)
Research Center for Automatic Control, Université de Lorrain

_____
Prof. Dra. Ana Paula Cabral Seixas Costa (Internal Examiner)
Federal University of Pernambuco

_____
Prof. Dra. Danielle Costa Morais (Internal Examiner)
Federal University of Pernambuco

_____
Prof. Dr. Philip Anthony Scarf (External Examiner)
Department of Mathematics of Cardiff University

_____
Prof. Dr. Hai Canh Vu (External Examiner)
Compiègne Univesity of Technology

ABSTRACT

As the time goes by, organizations acknowledge more and more the role of business support functions for the achievement of competitiveness and a sustainable performance. Considering that, it is important to propose novel mathematical models that enable the improvement of these functions. In the recent years, ML-based models have gained popularity in areas such as robotics, natural language processing, manufacturing, logistic and maintenance management. They have proven to be efficient in these complex domains in which the relation between some variables is sometimes unknown or in which the problem dimensionality and the solution space are high. Accordingly, in this thesis, we propose a maintenance and a logistic model based upon Machine Learning technics (ML) that have the capacity of dealing with the complexity of the problems approached when some real-life characteristics are taken into account. The first proposed model is based upon Deep Learning and aims to classify e-commerce orders in dropshipping systems as soon as they are placed on the internet. The model fulfils the gap in the literature in which models force e-taler to cumulate batches of orders before engaging in any order classification and inventory rationing. The second model is a Condition-based maintenance policy for multi-component systems based upon Deep Reinforcement Learning and Goal Programming. The model fulfills a gap in the literature in which real industrial system factors such as multiple degradation states, imperfect maintenance and multiple conflicting criteria are not considered. In order to validate the efficacy of each model, numerical experiments and sensitivity analyses were conducted using simulation. Results showed that the proposed models enable the improvement of key indicator performances such as order fulfilment rate, total e-tailer’s profit, maintenance cost rate and average system’s reliability, in different scenarios.

Keywords: deep learning; inventory rationing; dropshipping; reinforcement learning; multi- component systems; imperfect maintenance.

RESUMO

Com o passar do tempo, as organizações reconhecem cada vez mais o papel das funções de suporte no alcance da competitividade e de um desempenho sustentável. Diante disso, é importante propor novos modelos matemáticos que possibilitem o aprimoramento dessas funções. Nos últimos anos, os modelos baseados em aprendizagem de máquina (ML) têm ganhado popularidade em diversas áreas tais como a robótica, o processamento de linguagem natural, a manufatura, a logística e o gerenciamento da manutenção. Esses modelos têm se mostrado eficientes nesses domínios complexos em que a relação entre algumas variáveis é desconhecida ou em que a dimensionalidade do problema e o espaço de soluções são grandes. Nesse sentido, esta tese propõe um modelo de logística e outro de manutenção baseados em aprendizado de máquina. Estes modelos têm a capacidade de lidar com a complexidade dos problemas abordados quando algumas características realistas são consideradas. O primeiro modelo proposto é baseado em aprendizagem profundo e visa classificar os pedidos de e-commerce em sistemas de *dropshipping* imediatamente após o recebimento no sitio web. Este modelo preenche uma lacuna da literatura em que os modelos forçam os varejistas a acumular lotes de pedidos antes de classificá-los ou de fazer a alocação do estoque. O segundo modelo é uma política de manutenção baseada na condição para sistemas de múltiplos componentes, baseado no aprendizado profundo por reforço e na programação por metas. O modelo preenche uma lacuna na literatura em que alguns fatores de sistemas industriais reais, tais como múltiplos estados de degradação, manutenção imperfeita, e critérios múltiplos e conflitantes, não são considerados. Para validar a eficácia de cada modelo, foram conduzidos experimentos numéricos e analises de sensibilidade usando simulação. Os resultados mostram que os modelos propostos possibilitam a melhoria do desempenho de indicadores-chave, tais como a taxa de atendimento de pedidos, o lucro total, a taxa de custo de manutenção e a confiabilidade média do sistema, em diferentes cenários.

Palavras-chave: aprendizagem profunda; racionamento de estoque; *dropshipping*; aprendizagem profunda por reforço; sistemas de múltiplos

componentes; manutenção imperfeita.

LIST OF FIGURES

LIST DE TABLES

TABLE OF CONTENTS

1 INTRODUCTION

As the time goes by, organizations acknowledge more and more the role of optimizing business support functions on the achievement of a sustainable performance and competitiveness.

Two key business support functions are maintenance and inventory, which are responsible for a large part of the operating cost of an organization. According to Bevilacqua, & Braglia (2000) maintenance activities can represent between 10 and 70 percent of the production cost in a company, while the cost of spare parts consumed in maintenance activities accounts for almost 2.5% of its annual budget (Gallagher, Mitchke & Rogers, 2005). On the other hand, the stock-outs incurred in the e-commerce market due to an inadequate inventory management can result in near 25.4% of incomplete orders and 12% of delayed orders (Enos, 2010; Kim & Lnnon, 2011). Considering that, it is important to constantly invest in novel strategies and solutions that allow the optimization of these functions.

With the rapid incorporation of new technologies tied to Industry 4.0 and electronic services under firms, business support functions are supposed to change dramatically (Cagle *et al.*, 2020). Therefore, it is important that the novel models are aligned with new challenges and opportunities that can change the way in which industrial/services operations are carried out, in which decisions are taken, and the way the information is used in light of the adoption of Industry 4.0 and the spread of electronic services.

According to Franciosi et al. (2020), the adoption of key technology of I4.0 such as Augmented Reality (AR), Additive Manufacturing (AM), Big Data & Analytics (BD&A), Cloud Computing (CC), Industrial internet of Things (IoT), Horizontal/Vertical Integration (H/VI), Autonomous/Collaborative Industrial Robots (A/C-IR), Simulation (S), and Cyber-Security (CS), can affect both positively and negatively the effectiveness of the maintenance functions. The effects can be perceived in areas such as the access to maintenance tools, support equipment, and equipment performance data, the availability of spare parts, the provision of safety conditions for the maintenance environment, maintenance budget, maintenance management, capability of foreseeing and preventing potential critical equipment´s states, and the quality of maintenance actions. The integration and transparency of the key

technology and paradigms of I4.0 can also lead to changes in order fulfilment management (Fatorachian & Kazemi, 2021). For example, cloud technologies, Artificial intelligence (AI), Big Data Analytics (BDA), and the integration of RFID tags with inventory management systems can enhance real-time information access and analysis, collaboration, responsiveness, and product and material visibility throughout the supply chain. Particularly, having an advanced visibility of inventory levels can enable accurate inventory planning and control, which in turn enable a reduction in the cost of stock outs and a consequent customer satisfaction improvement (Quesada *et al.*, 2008; Tan *et al*., 2017).

Considering the aforementioned discussion, this thesis investigated one logistic problem and one maintenance problem that are very common in real and complex industrial systems. The proposed models take advantage of the increasing in data availability and the emergence of Machine learning technics, which allow high data-driven decision-making, computing power, efficient data pre-processing and the representation of high dimensional problems, in order to provide promising solutions.

The first problem approached in this thesis is the occurrence of frequent stock outs in the business-to-consumer market (B2C), which leads e-tailers to loss profit and service-level opportunities. Particularly, stock-outs are caused by two main factors: the difficulty of forecasting demand boosted by the widely adoption of electronic services, and the difficulty of synchronizing inventory replenishment for all the SKUs (stock keeping units) offered. The first one is challenging due to factors such as high frequency small orders, timely delivery requirements, and rapid habit consumer changes. The second one is challenging due the necessity of keeping in the inventory an ever-increasing number of types of products, and the difficulty of planning orders beforehand caused by the high demand variability in e-commerce. Given the aforementioned problems, two of the ways of providing service level and productivity to e-tailers are the collaboration between players and the improvement in the way the on-hand inventory is managed.

One of the strategies that e-tailers use to fight stock-outs and improve the inventory rationing is called dropshipping. In this strategy, the e-tailer outsource the picking, packaging and delivering of orders that it cannot fulfill, sending them to other players such as the wholesaler or manufacturer. By doing so, the e-tailer can

generate profit opportunities, save storage space and save inventory holding costs, while share the revenue with the other players(s) (Khouja & Stylianou, 2009; Chen et al., 2018). However, in order to implement a dropshipping strategy/system, it is necessary to develop rules to decide how to split the demand between the e-tailer and the other player(s). Such a decision is not straightforward, since the way the orders are split can influence the revenue of each player, the number of orders that can be fulfilled with the available inventory, and the risk of order delays (Jiménez et al., 2019). Due to that, there are many models in the literature consisting on different splitting rules that aim to provide margin and service-oriented opportunities to e-tailers. They take into account different factors such as the order arrival time, the type of client, the contributing margin of SKUs that compose the orders, and the available inventory of the demanded SKUs.

In this thesis, we propose an algorithm for efficiently splitting the demand and rationing the e-tailer's inventory in a dropshipping system. However, our model intends to overcome the limitations of other inventory rationing models in the literature, in which the e-tailer is forced to cumulate batches of orders for a complete period before taking any decision on how to split the demand. On the one hand, e-tailers can benefit from cumulating demand because they can gather more information about the inventory position, which in turns can prioritize orders that increase inventory usage and profits, and outsource the other orders (Mahar & Wright, 2009). On the other hand, when orders are cumulated, there is a delay in the decision making. The consequence of that is that the e-tailer should decide how to allocate the on-hand inventory and to split orders between players in the day after the arrival of orders. In this sense, the proposed model allows both advancing the inventory rationing decision making and the improvement in productivity and service-level performance by classifying orders in near real time based on characteristics such as the demand for SKUs, the inventory availability of SKUs, and the potential order margin.

The second problem approached in this thesis is the lack of effective maintenance policies that are well aligned with the characteristics of complex real industrial systems, which are composed of multiple components, which degrade visiting multiple states, and that can be recovered to better degraded states through imperfect maintenance actions. Besides, maintenance policies lack current concern

factors such as plant/worker safety, environmental impact, and social impact that are beyond the economic gains or for which economic impacts are difficult to be measured.

The fact that real industrial systems are composed of many components and subsystems supposes a challenge for maintenance optimization, since the systems develop different kind of dependencies, such as structural, economic, stochastic and resource ones, that grow among the many components and subsystems that are intertwined to each other (Van Horenbeek and Pintelon, 2013; Kazer et al., 2017). These complex dependencies between components add a dynamic character to the degradation and the aging of the system which difficult the prioritization of components and the selection of effective maintenance action sets for avoiding failures (Haugen et al., 2018). Therefore, maximizing the efficiency and profitability of assets in the aforementioned systems requires developing effective decision-making models capable of coping with the uncertainties on the system degradation processes, caused by the component dependence or by any other source of uncertainties such as operational conditions and external environment (Wang X et al., 2014).

On the other hand, tracking the health of the systems by monitoring intermediate degradation states, and allowing imperfect maintenance via condition-based maintenance on them is important. These practices allow critical states be identified in order to avoid failures, and to give place to maintenance opportunities in order to increase the efficiency of maintenance policies. The consideration of these factors allow managers to cope with the dynamic character of the system degradation process allowing the reduction of unnecessary actions and gains in resource usage and system availability performance (Marseguerra et al., 2002; Cheng et al., 2018). However, it supposes a challenge for maintenance optimization since the dimensionality of the problem increases as the number of components, the number of intermediate states tracked, and the number of imperfect actions allowed in each component increase. As a consequence, classical maintenance policies can become inefficient in terms of the solutions provided or in terms of computing effort (Barde et al., 2019).

Finally, the maintenance policies might consider the optimization of other criteria that are difficult to be measured in economic terms, rather than just the optimization of the cost criteria composed by parcels such as profit, breakdown cost, inventory holding costs and lost sales cost. Factors such as the environmental and social impact, the worker/plant safety, and the corporative image damage, can also be used to assess the efficiency of maintenance policies (Santos et al., 2021; Jones et al., 2009; Okoh, 2015; Marseguerra et al., 2002). In these cases, the maintenance optimization has a multi-criteria character, in which maintenance criteria have different dimensions and can be conflicting (*e.g.*, maintenance cost and worker safety), and thus it is necessary to consider the preference structure of the maintenance manager or the maintenance decision board- in order to guide the decision-making.

Considering the aforementioned context, the second model presented in this thesis proposes an inspections CBM model based on Deep reinforcement learning and Goal programming. The model suited for maintenance decision-making in multi-state multi-component systems allows imperfect maintenance and is capable of optimizing multiple objectives. The proposed model is able to consider the preferences of the decision maker and offers a means of extending the solution to systems with a different number of components and/or of degradation states.

1.1 OBJECTIVES

1.1.1 Main Objective

To develop machine learning based algorithms to improve the efficiency of e-commerce inventory rationing and maintenance policies in multi-component systems.

1.1.2 Specífic Objectives

- To determine the factors that affect inventory management in B2C;
- To determine a suitable indicator for measuring the importance of an order for an e-tailer;
- To develop an algorithm for order classification and inventory rationing in dropshipping systems;
- To study the conditions that affect maintenance policy efficiency in multi-state multi-component systems with imperfect maintenance and conflicting criteria;

- To develop a CBM policy for multi-component systems that can control the factors that affect system degradation;
- To validate proposed models via simulation.

1.2 MOTIVATION

The motivation of this thesis is the necessity of optimizing logistic and maintenance functions in e-commerce inventory rationing and multi-component system maintenance, respectively. In order to do that it is necessary to take into account some real-life characteristics of these problems that the models of literature are lacking, which would allow a better efficiency of the organization's strategies. This thesis is also motivated by the capacity of ML algorithms to cope with the complexity of these decisions when the aforementioned real life characteristics are considered in the problem formulation. Specifically, the relation between some variables might be unknown, the state-action space might be too large, or the transition state and the effect of solutions might be stochastic making the search of optimal solutions difficult. Particularly, the inventory rationing of e-tailers in partial dropshipping systems can be approached by a machine learning algorithm called deep neural networks (DNN). DNNs can benefit from the availability of records of fulfilled orders in order to classify the orders between the e-tailer and the other (s) player(s). In that way, the DNN would be able to mapping the relation between the characteristics of the order and the inventory, such as the demand of SKUs, SKUs availability, and SKUs margin to guide the decision making, which can also allow making the decision making in near real time.

The maintenance of multi-state, multi-component systems with imperfect actions and the inclusion of non-economic performance criteria can be approached by a ML algorithm called double deep reinforcement learning (DDQN), which can deal with the high state-action space that can be produced by the multiple states that the system components can visit and the multiple actions that can be taken in the same component, and in a set of components at the same time. An adequate representation of the state-action space and the concept of having an agent that can learn over time from experiences have the potential not just to deal with large state-action spaces, but can be useful for optimizing the

long-term performance of the system dealing with the stochastic character of the system degradation process that can be caused by the component dependency and the variability of external and operating conditions. Finally, the usage of DDQN can be complemented with a reward modeling based on goal programming, which can provide the DDQN algorithm with the capacity of integrating the preferences of the decision-maker on economic criteria such as downtime, inventory holding cost, maintenance cost, and non-economic criteria such as environmental and social impact, and worker/plant safety. This feature allows the DQN algorithm to assess maintenance policies through multiple objectives that are expressed in different dimensions and that can be conflicting.

1.3 RESEARCH METHOD

This study can be considered as an applied research, in which the modelling process itself is used as the research method. The modelling process is particular to each problem that is approached in this study, namely, the inventory rationing problem for e-tailer's stock that adopt dropshipping systems, and the maintenance optimization of multi-state multi-component systems under imperfect maintenance and conflicting criteria. The modelling process aims to uncover ways of including the factors that the body of the literature lacks in the existing studies regarding the aforementioned context, providing solutions that are suitable to real e-tailer's operation and industrial system maintenance practices. In order to achieve that, simulation is used as the work method, which allows the analysis of the data and the assessment of the solutions provided by the models proposed.

1.4 OUTLINE OF THE THESIS

This thesis was divided into five chapters: the first chapter, which was already introduced is a preface of the importance of solving the maintenance and logistics problems approached in this thesis. It also grounds the objectives of the study regarding the problems approached and the methodologies used to conduct the study. The second chapter corresponds to the theoretical and empirical advances, in which key concepts for the understanding of the problems are presented. The chapter also presents an up to day literature review on the models developed that aimed to cope with the uncertainties in the e-commerce demand

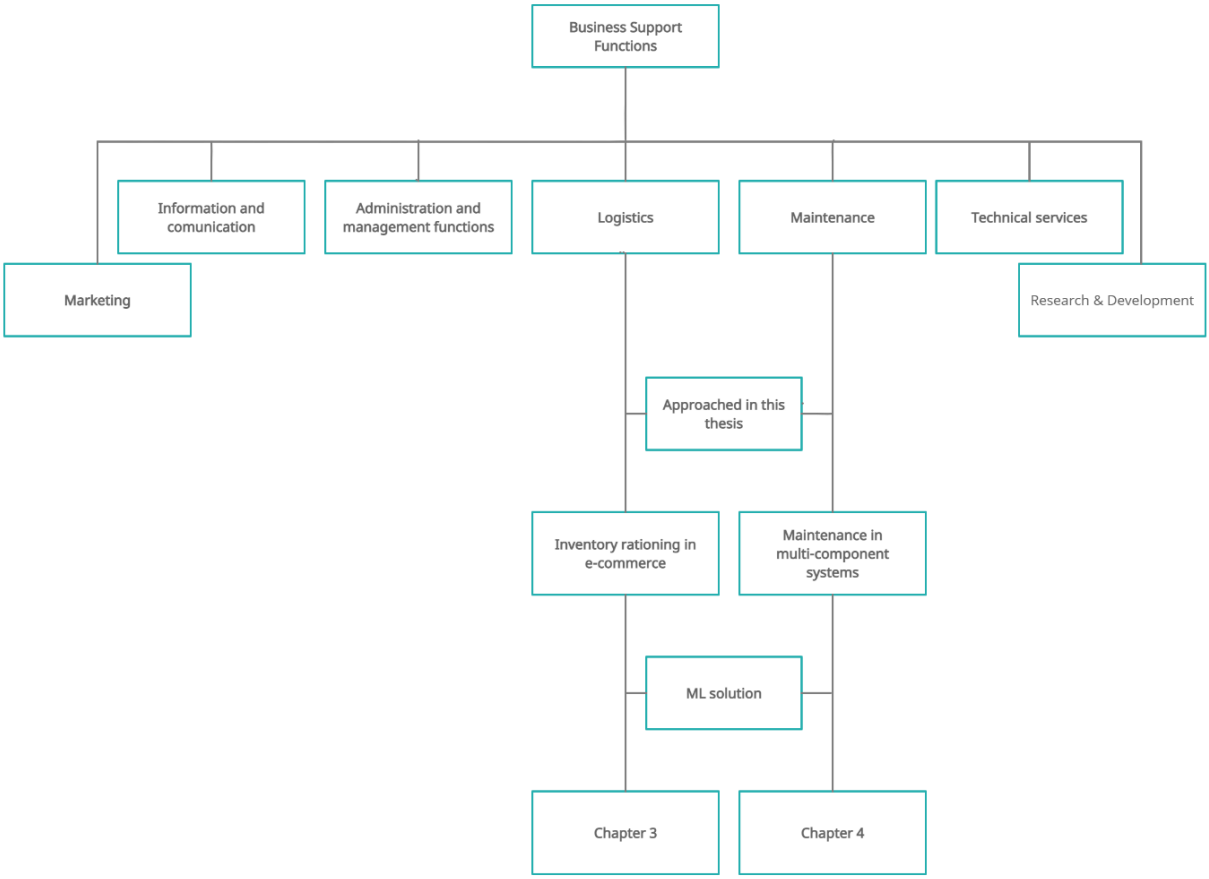
and in the degradation of multi-component systems, are presented. At the end of the aforementioned section, the gaps in the literature and the necessity of providing solutions that are more aligned with the real constraints and behavior of the systems approached, remain clear.

In section 3, a model for inventory rationing in dropshipping systems is proposed. In this section, the necessity of advancing the decision making and improving the inventory usage in order to provide better profit and service-level opportunities to e-tailers, is explained in details. A detailed formalization and justification of the use of a deep neural network classifier coupled to an existing inventory rationing algorithm is also given. Additionally, a numerical example with a sensitivity analysis is used to show how e-tailers with different sizes *i.e.*, with different types of SKUs, and with different demands *i.e.*, number of daily arrival orders, can benefit from the proposed model. The benefits include an advance in the decision-making, a smarter demand splitting between players, and an increasing in the revenues and the order fulfilment rate.

In section 4, a model for the optimization of maintenance of multi-component systems is proposed. It considers factors such as multi-state components, and the possibility of performing imperfect maintenance and assessing maintenance policy performance with multiple objectives (economic or not), that allow a more realistic modelling and then better solutions. The effectiveness of the model and its capacity of dealing with maintenance in complex systems is proven through a comparison with benchmark policies. In such comparison, the proposed model, which is based on reinforcement learning and Goal programming, offers a significant improvement over the long-term average system reliability and cost rate.

Finally, in section 5, the conclusions of the thesis are drawn highlighting the effectiveness of the machine learning solutions provided. Directions on future lines of research are also given. Figure 1 shows how the models proposed in chapters 3 and 4 are related.

Figure 1 - Link between proposed models



Source: The Author (2021).

2 THEORETICAL AND EMPIRICAL ADVANCES

The aim of this chapter is to present a review of basic concepts related to the logistics and maintenance's problems approached and the solutions provided during the development of this work, as well as the evolution of the models proposed in the literature. To this end, the main factors that affect maintenance efficiency in multi-component systems as well as the factors that influence B2C inventory rationing are presented. Additionally, works that contributed to the development of multi-component system's maintenance and B2C inventory rationing proposed so far were studied in order to position the proposed solution in the literature.

2.1 DEMAND VARIABILITY IN B2C MARKET

In the recent years, different e-commerce business models have grown very sharply as a consequence of the increase in the efficiency and safeness of internet shopping. The business models go from manufacturers that buy their materials from online wholesalers to individuals that buy their groceries through an app on their cell phones and from home.

One important e-commerce business model is called business-to-consumer market. In this model, the final user of the product is the buyer (a person from any location), and online retailers (e-tailers) are the sellers. The transactions between these parties occurs through the internet, and without intermediaries. The B2C market is characterized by the dynamic character of the demand, which is caused by a mix of factors that are also characteristic of that specific business model. Such factors include, but are not limited to the ever-changing consuming patterns, the growing customer exigencies for shorter delivery windows, and frequent small-size orders. The demand variability caused by the aforementioned factors affect the efficiency of the inventory planning and replenishment operation of e-tailers, which become infective and tend to worsen with the necessity of having an ever-increasing number of SKUs to be offered and kept in the warehouse, and the stressed inventory policies adopted with the aim to reduce the holding cost. These factors difficult the synchronization of inventory replenishment policies for the SKUs kept in the inventory, and then lead e-tailers to face stock outs very often.

When the e-tailers face stock outs, the postponement of orders is increased, as well as the logistic cost such as the picking, packing and transport of orders that have to be delivered in more than one consignment. The postponement of orders problems encourages e-tailers to adopt strategies such as offering discounts or outsourcing the fulfilment of orders in order to influence the tendency of customers buy another player. Finally, with the inefficient usage of the inventory, the cost that is incurred, and the amount of orders that cannot be delivered on-time or that cannot even be delivered because of lack of inventory, e-tailers lose profit and service-level opportunities.

2.2 DROPSHIPPING SYSTEM

Dropshipping is a collaborative system used for fighting stock outs in e-commerce, which consist of an e-tailer that sell products through the internet, and that outsource the picking, packing, and delivering of orders that it cannot attend due to stock outs or to a given interest on other orders/clients (Khouja & Stylianou, 2009; Chen et al.,2018). In this transaction, the outsourced players such as wholesalers, manufacturers or other e-tailers, share the revenues of the orders with the main e-tailer or seller. When the e-tailer has his own inventory and just outsource part of the demand, the system is called partial dropshipping, which is the system used in this thesis. On the other hand, when the e-tailer does not hold any inventory and does not have an in-house order fulfilment operation, the system is called total dropshipping.

The benefit of using this system, is that the e-tailer can save inventory holding space and cost, it can attend a bigger demand and then improve its revenues and its service-level oriented performance (Jimenez et al., 2019; Jiménez et al., 2020). On the other hand, the drawbacks of using dropshipping are that the players that agree to fulfill the outsourced orders, rarely assume the risk of order delay, so if order are not delivered or are delayed, clients may cancel the orders, leave negative reviews that can affect the e-tailer’s current and future business (Kim and Lennon, 2011). Additionally, in dropshipping systems in which the contractual relationship between the players is not clear or not trustful, the players can use customer information on their own behalf, or even start selling the same products on its own (Park, 2016).

Finally, the employment of partial dropshipping system implies the necessity of having a strategy for splitting the demand between the e-tailer and the other(s) player(s), since this decision can affect the revenues and the effectiveness of inventory usage of the e-tailers (Cheng 2016). The revenues can be affected because outsourcing the order fulfilment of any specific order, implies that the order will produce a smaller profit than when fulfilled with the on-hand inventory, so order's profit should be compared carefully. On the other hand, the efficiency of usage of the on-hand inventory can be affected since some orders can consume a large part of the inventory in relation to orders that can generate a higher margin and consume the inventory at the same level (Jimenez et al., 2019; Jimenez et al., 2020). Considering that, it is necessary to take into account the inventory position and the demand for SKUs of each order. As follows, a comprehensive review of the literature on inventory rationing models is given, highlighting their contribution to the topic and their gaps.

2.3 LITERATURE REVIEW ON INVENTORY RATIONING MODELS FOR E-COMMERCE

The problem of stock-outs in the supply chain has been addressed in the literature by different authors. The proposed models are based on two main types of strategies, namely stock control and inventory rationing policies. The earlier included tactical level decisions such as lot size, stockpile, and period length decisions that allows e-tailers a better planning for acquiring SKUs in bigger quantities and quickly according to the needs of the fulfilment process (Shekarabi et al., 2018; Gharaei et al., 2019a). Some authors have extended these type of models by integrating strategic-level decisions such as warehouse location, routing strategy, and warehouse capacity (Rim & Park, 2008; Diabat & Deskoores, 2016; Rabbani et al., 2018; Hamdan & Diabat, 2019). The joint optimization of these decisions not just enable improving the fulfillment efficiency of the supply chain actors but offering shorter lead times to clients with the aim to build loyalty and fight in some extent the demand variability. Other authors have also integrated other current organizational factors that customers are sensitive to such as social responsibility, environmental print, product quality and price. These models proposed channel coordination and profit division approaches that not just

aim to building loyalty in order to stabilize the demand to some extent, but also optimizing revenues of the different players of the supply chain that are involved in the aforementioned tactical and strategic-level decisions (Modak et.al, 2014, Modak et.al, 2016a, Modak et.al, 2016b, Modak et.al, 2016c, Modak et.al, 2016d, Modak et.al, 2018; Gharaei et al., 2019b; Gharaei et al., 2019c; Kazemi et al., 2018).

According to Jimenez *et al*. (2019), the aforementioned type of models is very generic for dealing with stock-outs in e-commerce, since although they do deal with shortage in a more holistic way including tactical level and strategic-level decisions that allow a more efficient design of the multi-echelon supply chain, they generally consider shortage in just one or two types of product. In real warehouses of e-tailers, hundreds or thousands of types of products should be maintained simultaneously. Therefore, the synchronization of continuous stock control policies for all the products turns very challenging and need to be supported by another strategy that allow a better rationing/splitting of on-hand SKUs. This strategy should be capable of supporting the order planning problem, answering questions such as which orders to attend with the on-hand inventory and in what sequence, and given the case, which orders to outsource (Jimenez *et al*., 2019).

The second type of strategies mentioned at the beginning of this subsection that allow dealing with stock-outs are called inventory rationing-based models. These models are based on different allocation rules that enable splitting the on-hand inventory between as many orders as possible. These rules include: sending just a part of the order delaying the missing SKU´s delivery; postponing the delivery of the whole order when at least one item is missing (Rim & Park, 2008); and compensating order delays with dynamic price discounts (Ding et al., 2006). These strategies are valuable in the sense that they allow extending the fulfillment period which allow e-tailers fulfilling more orders despite stock outs. Others even allow avoiding the delivery of orders in more than one consignment which bring additional shipping cost. However, the postponement of complete orders is still a problem that cause customer dissatisfaction, some of who are not willing to accept delays even with price discounts. Finally, even if customers

accept receiving their order with delay, the credibility of the e-tayler and the loyalty of customers get compromised.

Other strategies have been proposed in order to diminish the order postponement when the on-hand inventory is rationed. One strategy that stands out by its capacity of creating add value from collaboration is called dropshipping. It consists in the e-tailer sharing orders with other players such as e-tailers, wholesalers, and manufacturers, when it faces stock-outs or when it wants to dedicate his stock to specific clients, segments or orders that it perceives as more advantageous (Khouja & Stylianou, 2009; Chen et al., 2018; Jiménez et al., 2019; Ayanso et al., 2006). As an example, Jimenez *et al*. (2019) and Jimenez *et al*. (2020) have developed partial drop-shipping models that allow e-tailer prioritizing the on-hand inventory for fulfilling orders with the higher criticality performance, i.e., the best relation between margin and inventory consumption. The rest of the orders are shared with the supplier which include orders with low criticality and orders that have missing SKUs when faced with the on-hand inventory. These models have demonstrated to increase profit and service-oriented opportunities for e-tailers.

2.4 MODELLING AND MAINTENANCE IN MULTICOMPONENT SYSTEMS

The development of adequate maintenance strategies can lead industrial systems to high standards of efficiency, increasing the availability and the reliability of the equipment, and the productivity and profitability of the operations (Ding and Kamaruddin, 2015), (Dekker, 1996).

On real industrial systems, the degradation and reliability of the whole system are dominated by the interaction of the different components that compose it. These components can develop different type of dependencies such as economic, structural, and/or stochastic ones. Economic dependency can arise when the way the system can be maintained depends on how beneficial is in economic terms to maintain certain components together. It generally accounts for savings in setup activities. The structural dependency can arise when components are connected physically composing a structure, so if one component is broken the structure may not work and then, they should be maintained simultaneously. Finally, stochastic dependency arises when the

degradation of one component could affect the degradation of others. The dynamic character of the system degradation can also be boosted by the increasing in the number of components since the dependencies already mentioned strengthen. Therefore, the arrival of unforeseeable failures of the system get potentiated.

Considering the aforementioned discussion, it is vital to develop maintenance models that consider the uncertainties of the degradation of multi-component systems in order to define a suitable maintenance plan that allows the optimization of the long-term performance of the system (Nicolai and Dekker, 2008; (Horenbeek, Van, Buré, et al., 2013).

One effective way of dealing with the dynamic character of the system degradation process is called condition based maintenance (CBM). It consists of tracking the health of the system and defining preventive thresholds for its components, that is, defining critical degradation states for each component that trigger dynamic maintenance actions. This approach allows unnecessary maintenance actions to be reduced which in turn allow making savings on maintenance resources and improving the system availability. One may distinguish between two types of CBM approaches, namely continuous-based CBM and inspection-based CBM. In the former, the system health tracking can be supported by sensors in order to identify the state of the components continuously. In the later, the health of the system/components is checked at every fixed time-window via inspections. It is commonly used in systems in which sensors are very expensive, in which other functions, such as quality, force inspections at fixed windows, or in which operating conditions such as temperature or pressure make unfeasible obtaining data via sensors (Marseguerra et al., 2002; Cheng et al., 2018). A detailed review on models, algorithms and technologies for diagnostics and prognostics using CBM can be found in Jardine et al. (2006).

Among the many models that compose the corpus of the CBM's literature, some authors have recently approached the optimization of CBM for industrial systems using cutting-edge machine learning technics. Reinforcement learning for example, is an adaptive learning type of algorithms that is very successful at

tackling complex problems in dynamic environments such as robotics for industrial automation, manufacturing, business planning, natural language processing, self-driving transport, and healthcare…Specifically, it has the capacity of acquiring experiences through interactions with an environment without any specific instruction, and learning the immediate and the future impact of actions over the performance of the system in order to define efficient courses of actions (Kaelbling et al., 1996).

RL algorithms stand out over wide used approaches such as dynamic programming due to a variety of reasons. For example, RL algorithms have the capacity of dealing with incomplete information such as in problems in which the one-step transition probabilities of states are unknown. Another reason is their capacity of dealing with large state spaces and actions when combined with deep learning algorithms which demand higher memory space and computing times (Barde et al., 2019). The later seems to be very important in the context of multi-component systems with high number of components, in which setting thresholds for each component is not just impractical but might be suboptimal (Liu et al., 2020). Therefore, RL can allows more efficient CBM policies in which the search of solutions in the state-action space is more efficient. These models can establish dynamic maintenance actions that do not follow a specific rule for individual components but rather for a combination of them. As follows, a comprehensive literature review of RL based models for CBM optimization, as well as their contributions and gaps are presented.

2.5 LITERATURE REVIEW ON RL-BASED CONDITION BASED MAINTENANCE

A plenty of RL models has been developed to solve a variety of realistic control and decision-making issues in the presence of uncertainty. Some of these authors have developed models for optimizing preventive and grouping maintenance policies in multi-component systems. These models are generally base on top of RL algorithms such as SARSA(λ), Q-learning and actor-critic (Huang et al., 2020; Rochetta et al., 2019). However, despite of their capacity of dealing with the uncertainties of the degradation of multicomponent system, authors such as Rocchetta et al. (2019) have stablished some disadvantages that

should be addressed. The main drawbacks of the aforementioned methods lie in usage of a demanding memory intensive representation of states and actions, which make it hard to scale them to real high-dimensional problems. In these contexts, which are characterized by large state and actions spaces, the aforementioned RL algorithms can produce unfeasible computational times for finding good solutions. In the context of maintenance, we can suppose the difficulty of employing the aforementioned RL methods for optimizing maintenance in multi-component systems, in multi-state systems in which intermedium degradation states are tracked for exploring maintenance opportunities during inspections, or in both.

A few authors have proposed RL algorithms that overcome the difficulties of the aforementioned algorithms (Kuhnle et al. 2019; Wang et al., 2016; Huang et al, 2020; Wang X et al., 2014; Andriotis et al., 2019). These have been adapted for approaching maintenance systems with multiple degradation states, or for guaranteeing certain levels of production that should be satisfied in system with productivity degradation. The authors have used a family of algorithms such as Deep Reinforcement Learning (DQN) due to its capacity of representing the state and actions spaces in a non-tabular way and to speed up the optimization process and the q-values convergence using deep neural networks (DNN) as function approximators. Some of these authors have also used a variant of DQN called Double Deep Reinforcement Learning (DDQN) for bringing a higher capacity of dealing with high dimensional problems. The design includes a second DNN that provide more stability to the minimization of the loss of the q-value function.

Despite the benefits of using the aforementioned RL algorithms, it is important to note that their applications were made approaching the systems whether as a simple component, as having independent subsystems, or as production lines, in which buffers should be set among them in order to delay the propagation of machine stoppage. Therefore, an application and adaptation of these algorithms, specially DDQN ones, in real industrial systems composed not just by multiple degradation states of single components but by multiple component that interact between them boosting the uncertainties of the system degradation, is an

important task and a potential opportunity for improving the efficiency of maintenance policies in these systems.

Other papers have used RL-based models for integrating maintenance with other decisions such as production control and quality control (Rocchetta et al 2019; Paraschos et al., 2020; Zhang & Si, 2020). These papers have shown exciting results when compared to other existing condition-based and grouping models. They yield decreased computing times, can be scaled to high dimensional problems in theory, and allow the integration of various decisions across the industry offering holistic solutions to industrial problems. Despite that, these models present some limitations regarding the representation of degradation states and the complexity of the systems, for example: Rocchetta et al. (2019) have developed a model based on RL and an ensemble of DNN's for the operation and maintenance optimization of power grids. The authors considered both the multi-state and multi-component characteristics of these systems. However, the maintenance action set is relatively small and the computing time is still an open issue; Paraschos et al. (2020) have developed a tabular RL algorithm modelled after the Q-learning algorithm and called R-learning. The model allows the integration of production, maintenance, and quality decisions for systems with multiple degradation states. However, the system to be maintained is approached as a single unit and thus, maintenance actions are defined over a single component (the system). Therefore, the representation of a more realistic system with multiple component that degrade simultaneously is not approached. Finally, Zhang & Si (2020) developed a customized DQN algorithm for approaching a multi-state multi-component system with dependent stochastic and economic competing risks. The algorithm maps directly the system state to the action space without the need of setting maintenance thresholds. Despite that, the model does not consider imperfect maintenance actions. This type of maintenance actions represents an opportunity in multi-component systems that can allow managers to exploit the health information regarding the intermedium degradation state of the system's component. Taking this information into account for driving maintenance decisions can help managers balancing the maintenance cost of the policies and the availability of the system. When these actions are performed opportunely, they allow taking advantage of the economic dependency

of components and the opportunities for recovering some components to better degraded states when others are being maintained.

Finally, Liu et al. (2020) and Andriotis et al. (2019) developed opportunistic maintenance optimization models based upon RL that deals with multistate multi-component characteristics of complex systems, imperfect maintenance, limited maintenance resources, and partially and fully observable multi-component engineering systems. The algorithm based upon an actor-critic algorithm works as a DQN adapted algorithm which allow scaling solutions to high dimensional problems and to provide competent near-optimal solutions to otherwise intractable learning problems.

Although the aforementioned RL-based models have the capacity of coping with the uncertainties in the degradation of either multi-state, multi-component systems, or both, they present the same drawback; maintenance policies are assessed only in economic terms. In these models, the efficiency of maintenance policies is measured by factors such as maintenance cost, inventory holding cost, defective product cost, downtime cost, and any other criterion that can be expressed in monetary terms. Therefore, these models are not suited for multi-objective optimization.

The optimization of maintenance in multi-component systems using multiple objectives is justified by the increasing importance grounded in the sustainability of manufacturing processes and the adoption of Industry 4.0 (I4.0). Specifically, organizations are getting increasingly concerned about the social and environmental impacts of maintenance activities as a way of creating value so that products can be processed not just optimizing the cost effectiveness of maintenance actions and the system availability, but minimizing environmental impacts, usage of energy and natural resources, and safety risks for safe employees, community and consumers (Jasiulewicz-Kaczmarek *et al.*, 2021). These environmental and social factors are generally measured with subjective measurements or are difficult to be measured in economic terms, which demand a multi-objective optimization when considered together with economic factors. Additionally, these economic and non-economic criteria can be conflicting at times. One example is the conflict between the maintenance cost and the safety

of workers. Thus, we can try to minimize the cost of maintenance actions by performing preventive actions and avoiding costly corrective actions. However, after a certain point, if one wants to increase the safety conditions of worker in a specific operation context, we might have to sacrifice some cost performance by increasing the amount/type of preventive/proactive actions in the system beyond its optimum limit or combination. This situation would demand the consideration of the preferences of the maintenance board over the criteria.

Another sign showing the need of multi-objective maintenance optimization is related with the progressive adoption of I4.0 paradigm. I4.0 is moving quickly to take advantage of innovations in industrial IoT, AI and machine learning (ML), edge computing and wireless communications, encouraging the integration of information, operation, and decision-making across the different areas of the organization. As a consequence, the joint optimization of maintenance and other decisions such as quality, production, and inventory control, each of one with its own performance criteria, makes itself more necessary and possible than ever (Paraschos et al., 2020; Lindström et al., 2017; Cheng et al., 2018; Bahria et al., 2018). To this complexity, we can also add the integration of key technology of I4.0, such as Augmented Reality (AR), Additive Manufacturing (AM), Big Data & Analytics (BD&A), Cloud Computing (CC), Industrial internet of Things (IoT), Horizontal/Vertical Integration (H/VI), Autonomous/Collaborative Industrial Robots (A/C-IR), Simulation (S), and Cyber-Security (CS), to production lines and maintenance teams. Despite the fact that this integration aims to facilitate maintenance efficiency, it can require the joint optimization of new performance criteria to guarantee the efficiency of the key technology and then contribute to the efficiency of the whole system. According to Franciosi et al. (2020) the integration of the key I4.0 technology with maintenance will deeply affect maintenance processes in terms of management, realization, and support processes, to make them smarter and more sustainable. Table 1 presents a summary of the maintenance process that can be affected by each key I4.0 technology.

According to Franciosi et al. (2020), all the key I4.0 technology, that affect the maintenance processes presented in Table 1, will have both positive and negative impacts over the different dimensions of sustainable manufacturing

systems, namely, economic, environmental, and social dimensions. Therefore, they claim the necessity of developing multi-criteria/multiobjective assessment models to measure the effect of the integration of these technology with maintenance.

Table 1 - Impact of key I4.0 technology in maintenance processes

Maintenance process	I4.0 technology								
	AR	AM	BD&A	CC	IoT	H/VI	A/C-IR	S	CS
Maintenance management	X		X	X	X	X	X	X	X
Execution of Preventive and corrective actions	X						X		
Maintenance budget	X	X	X	X	X	X	X	X	X
Operational documentation delivering	X						X		
Data management	X		X	X	X		X	X	X
Health and safety guarantee to individuals and maintenance environment preservation	X						X		
Maintenance requirement delivering during items design and modification							X		
Result improving	X		X	X	X		X		
provision of Internal human resources	X			X	X		X	X	X
Provision of external maintenance services			X			X			
Spare part delivering		X				X	X		
Delivering of tools, support equipment & information systems	X		X	X	X		X		

Source: Adapted from Franciosi *et al*. (2020).

2.6 FINAL REMARKS OF THE CHAPTER

This chapter presented the contributions of inventory rationing models and RL-based CBM models presented in the existing literature, as well as the gaps that can give place to the design of novel models that fit the characteristics and needs of real industrial systems. The review of inventory rationing models makes clear the necessity of novel models that allow e-tailers to improve the

effectiveness of inventory usage and order splitting in the light of the adoption of dropshipping fulfilment as a collaborative way of providing e-tailers with more profit and service-oriented opportunities. Novel methodologies can also explore ways of overcoming the limitation of existing models that force e-tailer to cumulate batches of orders for entire periods in order to better use the on-hand inventory. On the other hand, on the literature review on RL-based CBM models, the necessity of maintenance policies that consider the characteristics of the degradation of real multi-component systems was elucidated. The novel policies might deal with the uncertainties resultant of the simultaneous degradation of many components, of the variability of operating conditions, and any other source of uncertainty that affect the way in which maintenance actions should be conducted to guarantee an optimum system availability and maintenance cost. These policies might also consider the possibility of performing imperfect maintenance in order to take advantage of maintenance opportunities given during inspections, in which degraded components can be recovered to better degraded states, enabling strategic intermediate system states that can help balancing the availability and the cost performance. Finally, the models can enable the optimization of policies based on multiple criteria that can be useful for assessing the policies considering non-economic current concerns such as environmental impact, worker/plant safety, or any other non-economic criteria that can be considered in order to integrate the optimization goals of other functions such as in production, inventory, and quality control.

As follows, we present a classification algorithm that allows e-tailers that use a dropshipping system to ration their on-hand inventory and to split the demand between them and other players in order to better face stock outs.

3 AN ONLINE DEEP NEURAL NETWORK-BASED MODEL FOR ORDER CLASSIFICATION AND INVENTORY RATIONING IN BUSINESS-TO-CONSUMER E-TAILERS

In the business-to-consumer market, e-tailers often have to deal with stock-outs due to the high variability of the demand and the consequent difficulty of synchronizing replenishment policies of SKUs. In this context, an efficient strategy for rationing the on-hand inventory is vital for offering more margin and service-oriented opportunities to e-tailers. This strategy implies an adequate splitting of the demand among fulfilment players when the e-tailer adopt a dropshipping system. In this chapter, we proposed a dropshipping inventory rationing model based upon deep neural networks. The model is able to overcome the limitation of existing models of forcing the e-tailers to accumulate orders before engaging in any demand splitting, in order to gather more information about the inventory positions of SKUs. The proposed model classifies incoming orders in near real time supporting the decision of which player should be designated to attend them, thus enabling the decision to be anticipated in a complete operating shift. Moreover, since the order classification decision is taken in advance, the model has the potential of improving the performance of subsequent activities such as picking, packing, delivering and inventory replenishment.

3.1 THE PROBLEM OF SPLITTING THE DEMAND IN A DROPSHIPPING SYSTEM

Suppose that there is an e-tailer that uses a partial dropshipping system as a fulfillment system when it has a limited on-hand inventory. In that case, it has to decide which orders should be fulfilled by himself using his internal inventory, and which ones should be attended by another player such as a manufacturer, a wholesaler, or other e-tailer. To make this decision, he cumulates orders that have been entered on the internet in a batch during an operational shift. After that he/she will be able to priories the cumulated orders according to some rule of thumb that will help him/her to split the demand between the players. Particularly in this thesis, a dropshipping system consisting of two players: one e-tailer and its supplier, is considered.

A very simple and widely used rule of rationing the inventory consists of meeting orders that arrive first following the first come-first-served rule (FCFS). In that case, the e-tailer prioritizes orders according to service-oriented goals, allocating orders to himself until all the on-hand inventory is depleted, while transferring to other player (s) the remaining orders. However, if the e-tailer want to use a more sophisticated strategy that help him to optimize not just his service level opportunities, but also his profit opportunities, he needs to develop a rule that also considers characteristics of the orders such as how the demand for SKUs of the orders deplete the on-hand inventory and the contribution margin of the demanded SKUs (Jiménez et al., 2019; Jiménez et al., 2020). The conception of such a rule is not that straightforward since on one hand, there are orders composed of SKUs that have larger combined contribution margins, which enable the e-tailer to increase his revenues. On the other hand, there are orders that demand a high quantity of SKUs which happen to have a low inventory availability, making it difficult for the e-tailer to manage the inventory to attend more orders. Therefore, such a strategy should enable e-tailers to chase order whose composition in terms of SKUs, leads to both high profit and low inventory consumption.

As shown in the literature review, there are some rationing policies that consider the aforementioned characteristics of the orders when splitting the demand between players. Although these models allow e-tailers to increase their revenues and to increase the number of orders fulfilled in each operation shift, they share a common drawback of other rationing models. Specifically, the e-tailers are instructed to cumulate batches of orders during complete shifts before performing any inventory allocation or demand splitting between players. This rule is practiced with the aims of gathering more information about the inventory position of the SKUs.

The consequence of the aforementioned drawback of existing rationing models, is that the decision-making over which player should attend each order is not made in real time, and then players lose a full shift of operation before starting to fulfill orders. Therefore, they lose profit and service-level opportunities that can be extended to several subsequent activities such as picking, packing, and delivering, which can benefit for obtaining the splitting information in advance to

be better planned. Having that information in advance can also favor some decisions that influence stock-outs such as inventory replenishment, which can also be better panned. Finally, when orders can be quickly picked, packaged and delivered, the inventory holding costs, specifically the obsolescence cost, can be reduced (Ozer & Wei, 2004; Wang & Toktay, 2008). The earlier is particularly true for food which comprises perishable items that have a short shelf life and lose quality exponentially according to the storage time.

Considering the aforementioned evidence, and the fact that in the context of the supply chain management, the speed at which transactions and operations are carried out is key determinant of efficiency (Morgan, 2004), it would be ideal to take the decision over each order in near real time, *i,e.*, as soon as each order enters the system.

3.2 METHODOLOGY

In order to decide whether the e-tailer or the supplier should fulfill a specific order as soon as it is entered on the internet, a classification procedure must be drawn up. It should create order profiles and compare them with the orders that enter the system in order to classify them. This classification should be done based on the order characteristics, the available inventory and e-tailer's productivity and service level goals. This procedure must act online, without the need to compare orders that are being entered, or to cumulate them in a batch for further analysis.

In this paper, the proposed procedure is based on a deep learning (DL) approach coupled with a known inventory rationing policy. It is called “inventory rationing based on deep learning” or IR-DNN. Deep learning models are based on deep neural networks (DNN) that consist of multiple processing layers. These structures can learn representations of large data with multiple levels of abstraction and perform functional input/output mappings in multivariate and non-linear problems (Lecun et al., 2015). This is possible given three main characteristics: parallel character, multi-parametric character, and high computing speed (Bre et al., 2018; Nguyen & Medjaher, 2019; Basheer & Hajmeer, 2000).

DNNs are considered powerful tools when mathematical models are not available or when the models are difficult to compute analytically. Therefore, the first reason for using DNNs in this study is because they can mimic and replace existing inventory rationing models enabling order fulfilment decision-making be made in advance. This is necessary, since, although the existing inventory rationing models consider the characteristics of the orders in order to prioritize them according to their importance, they rely on being able to cumulate orders before taking any decision on who should attend each order, which delay the decision-making. The second reason for using DNNs is that they can deal with high dimensional problems and large datasets, which seems to be inconvenient for traditional machine learning algorithms that are commonly used for binary classification such as Decision trees, Random forest, Bayesian networks, Support vector machines, and Logistic regression. Regarding the dimensionality of the problem, in the B2C order classification problem, the number of input variables that can be used to decide how to attend the demand increases exponentially as the variety of products offered by the e-tailer increases. These variables include the demand, the available inventory and the margin of each item. Therefore, DNNs have the potential of yielding a good performance irrespective of the size of the e-tailer's business. Regarding the dataset size, the performance of DNNs stagnates as a slower pace as the amounts of data increases regarding the aforementioned models (Lecun et al., 2015; Ng, 2016), which indicates that DNNs can benefit from large datasets that can be obtained when the number of e-commerce orders placed and SKUs offered increase.

A third interesting characteristic of DNNs is the size of the model; while in DNNs the input nodes, hidden layers, and output nodes are fixed, in other widely used algorithms such as SVM, the number of support vector lines could reach the number of instances in the worst case. Finally, in contrast to the aforementioned models, DNNs may not require extensive human interaction, and then, less work on feature engineering is needed (Khan & Yairi, 2018).

To evaluate the proposed IR-DNN, a simulation experiment was carried out. The parameters and conditions related to the demand, the fulfillment operation and the software packages used to construct the IR-DNN model are

specified in the simulation section. As follows, the way in which DNN and the existing inventory rationing model work together is shown.

3.3 IR-DNN MODEL

A DNN is used to capture two characteristics of the orders: the demand of the SKUs and their available inventory, and to map them to the productivity and service-oriented performance of the dropshipping system players. The DNN does this input/output mapping by using historical information about orders that were fulfilled “appropriately” in the past. The DNN uses this historical information in a training step in order to understand what the characteristics are that an order should have in order to be fulfilled by one of the players (supervised learning), *i.e.*, the e-tailer or the supplier(s). Once the ANN has been trained, it can be used for classifying incoming orders into two categories: e-tailer and supplier. This classification could be made immediately without the need to cumulate orders for further analysis and without using a pairwise comparison approach.

In this paper, the concept developed by Jimenez *et al.* (2019) was adopted to define the “appropriate” criteria for classifying orders. The authors established that the orders that should be fulfilled by the e-tailer are the orders that generate the greatest margin and that, at the same time, have the least adverse impact on the availability of inventory. The ratio of these two criteria is called "dynamic criticality" and was used by the authors to classify batches of orders, demonstrating that it is possible to offer more revenue opportunities while using the same inventory to attend to more customers.

The IR-DNN model proposed in this chapter uses the criterion “dynamic criticality” algorithm developed by Jimenez *et al*. (2019) and later extended by Jimenez *et al*. (2020) in order to simulate the allocation of orders for historical data of the e-tailer's operation and then complete the dataset that the DNN needs in order to be trained. By doing so, the characteristics of the order such as the demand for items, inventory availability and margin are captured by the ANN, which relates them to the player that was chosen to fulfill them in historical orders and creates an order profile for each player. These profiles enable new orders to be classified individually as they are placed. In terms of the vocabulary used in machine learning, an adaptation of the model of Jimenez *et al*. (2019) is used as

a base model to label the historical data records. Table 2 shows the definitions and notations used in the model.

Table 2— Notations and definitions used in the model

Notation	Definition
O_k	Placed order
J	Stock Keeping Unit offered
Ci	Criticality of order O_k
E_i	Level of inventory commitment for the order O_k
m_i	Profit on order O_k
e_j	Level of inventory commitment for item j
d_{ij}	Demand for item j in order O_k
q_j	Existing inventory of item j
OFR	Percentage of orders fulfilled by the e-tailer in a period
F	Number of orders fulfilled by the e-tailer during a period
N	Number of orders that arrive during a period
M_t	Profit obtained from the fulfillment of all the orders
M_{int}	Profit obtained from orders fulfilled using the internal stock
M_{drop}	Profit obtained from orders fulfilled using dropshipping
$m_{i.int}$	Unit margin of item j when attended to by internal stock
$m_{i.drop}$	Unit margin of item j when attended to via dropshipping
FIFO	First-in-first-out, or inventory allocation rule where the first order to enter is the first to be attended to
Dynamic Criticality	Inventory allocation rule based on the criticality index that uses perfect information on demand and inventory positions.
D_k	Input demand vector of all items in the order O_k
Q_k	Input available inventory vector of all items in the order O_k
G_k	Input margin vector of all items in the order O_k when attended to by the e-tailer
S_k	Input margin vector of all items in the order O_k when attended to by the supplier
A_k	Label or suitable fulfillment actor for attending to order O_k represented by a binary label (0,1)
MLF-NNs	Multilayer feed-forward neural networks
TP	True positive rate
TN	True negative rate
FP	False positive rate
FN	False negative rate

Source: The Author (2021).

3.3.1 Dynamic criticality algorithm

According to Jimenez *et al*. (2019; 2020), the e-tailer must prioritize his available inventory to fulfill orders that have the least adverse impact on the availability of inventory and provide the highest margin.

Orders with these characteristics must be fulfilled by the e-tailer because they contribute in an important way to the total margin, *i.e.*, besides being profitable by the nature of their items (more profitable), the dropshipping cost can be avoided which consequently improves the margin of the orders. These orders also enable e-tailers to optimize the available inventory and serve more customers. This is important because when the e-tailer fulfills a higher percentage of orders in relation to the total number of customers, the service level is increased, he gains more control over the picking, packaging and delivery of orders while the risk of the other player being delayed is reduced. This last factor is relevant in some e-commerce environments where the contractual relationship between players and the e-tailer is not well defined or is not trust-based. In this context, it can happen that the e-tailer becomes concerned about the possibility of the other player misappropriating information about customers for its exclusive benefit (De Gusmão et al., 2018). Consequently, the e-tailer might prefer to serve the maximum number of customers possible with the available inventory (Park, 2016).

Another feature of the policy developed by Jimenez *et al*. (2019) is that incomplete orders are not allowed. This means that orders cannot be shipped in multiple consignments and that all items on the same order are served by the same player. This element enables the additional shipping cost to be avoided and the order to be postponed. The result of this policy is a more productive and service-oriented operation, since the e-tailer can generate more margin and fulfill more orders.

To formalize the aforementioned concepts, Jimenez *et al*. (2019) brought together two elements: the level E_i in which each order depletes the inventory, and the margin p_i of the order. These elements form a criterion called criticality *Ci* given by Equation 1, which lets the orders be ranked according to their

importance for the e-tailer and this serves as a criterion for splitting the demand between the e-tailer and suppliers.

$$C_i = E_i / p_i \tag{1}$$

The inventory commitment E_i for a particular order indicates how negative, it could be attending to this order given its size and given the inventory availability. E_i is defined by Equation 2:

$$E_i = \sum_{j=1}^{J} \left(e_j \times d_{i,j} \right) \tag{2}$$

As the orders consist of different SKUs, the inventory commitment e_j must be calculated for each SKU individually according to its availability and the demand for it during an operational shift. While E_i refers to the impact of a specific order on the inventory, e_j refers to the inventory sensitivity of a specific SKU that is offered by the e-tailer. e_j is presented in Equation 3:

$$e_j = \frac{\sum_{i=1}^{I} d_{i,j}}{q_j} \tag{3}$$

The procedure of splitting orders between the fulfillment actors considering the aforementioned criticality concept is presented as follows:

1. Organize the orders in ascending order according to the criticality index;
2. Allocate the on-hand inventory to the first order in the sequence;
3. Update the inventory availability of SKUs and the list of remaining orders/demand;
4. Recalculate the criticality for the remaining orders;
5. Repeat from step 1;
6. If the quantity ordered for a specific SKU within a specific order is greater than the available inventory, allocate the whole order to a supplier;
7. If step 6 is done, recalculate the criticality for the remaining orders;

8. When all orders are assigned to one of the fulfillment actors, organize orders in each group (fulfillment actors) according to the initial order in which they were entered into the system;
9. Close the process.

Step 8 aims to increase the service-oriented performance of the fulfillment operation and ensure that if there is some delay in successive operations such as picking, packing and delivery for any of the fulfillment actors, it occurs in orders that arrive last. For more details about the criticality policy, please read Jimenez *et al.* (2019; 2020).

3.3.2 Input data

The aforementioned procedure was used to generate the instances that the DNN needs in order to be trained and validated. Each instance has data that contain the characteristics of the order such as the demand for SKUs, the inventory available in SKUs, the potential order margin when attended to each fulfillment actor, and a label that identifies the suitable fulfillment player.

The dataset was obtained by planning the orders of the warehouse e-tailer for several days based on the historical records of orders. In each period, the inventory initially available is checked and allocated to a batch of accumulated orders from the previous day using the model of Jimenez *et al*. (2019). Since this model is dynamic and the available inventory of items changes whenever an order is allocated to a fulfillment actor from a batch, the partial inventory positions were recorded during each step of the algorithm. The inventory position represents the remaining inventory after an order is allocated, but it also represents the available inventory of each item for the next order to be dealt with, which is checked by the warehouse management system (WMS) whenever an order is placed on the internet. Saving the partial inventory positions, thus enables the available inventory for each order to be tracked whichever the period, and this is used by the DNN as another feature for mapping the suitable fulfillment player. In the case of new orders, this information is retrieved from the WMS and the e-commerce website that makes the sale.

In summary, the dataset comprises a massive amount of orders from different periods, from which the demand, the available inventory, the potential margin of each SKU for both players, and the suitable fulfillment actor for each order, can be easily retrieved. The DNN should be able to use this information in order to understand what characteristics the orders are supposed to have in order to be classified as e-tailer or supplier without having to cumulate orders in a batch for further analysis.

Before feeding the DNN with historical information and training it, the dataset must be adjusted and completed. This is necessary since DNNs work with fixed inputs (features) and outputs (labels) and cannot deal with the unexpected and variable demand of SKUs in each order. Therefore, the first layer of the DNN consists of one node per each characteristic of the SKU, for each SKU offered by the e-tailer. In the case of the neurons that represent the demand for SKUs, the ones that are most in demand receive the amount demanded, while the ones that are not demanded receive a value of zero in the dataset also representing inputs for the ANN. In the case of the inventory, the availability of SKUs for the items that were not demanded is also informed in the dataset representing inputs. This configuration sets the inputs and outputs of the ANN to be fixed.

Formally, it is assumed a dataset $(D_k, Q_k, G_k, S_k, A_k)$ with $k = 1,2, \dots m$ instances. The inputs of the DNN are presented as follows:

- $D_k = (d_1(O_k), d_2(O_k), \dots, d_n(O_k))$ is the array of demand for all the SKUs that comprise the order O_k, zeros being allocated to non-demanded SKUs;
- $I_k = (i_1(O_k), i_2(O_k), \dots, i_n(O_k))$ is the array of inventory for all the SKUs that comprise the order O_k, the remaining inventory being allocated to non-demanded SKUs.
- E_k is the contribution margin of order O_k when it is fulfilled by the e-tailer;
- S_k is the contribution margin of order O_k when it is fulfilled by the supplier.

For *n* SKUs offered by the e-tailer the ANN has a number of inputs of $(n * 2) + 2$. Output or labels of the neural network are presented as follows:

- A_k is binary and is defined as follows:

$$A_k(O_k) = \begin{cases} 0 \; if \; O_k \text{ is fulfilled by the supplier} \\ 1 \; if \; O_k \text{ is fulfilled by the e} - \text{tailer} \end{cases}$$

It is important to consider that *m* depends on the quantity of historical data that is available and on the preference of the modeler. This parameter is defined in the simulation section.

3.3.3 Architecture of the neural network

In this study feed-forward deep neural networks were used. They are multilayer perceptrons in which the information flows in a sequential way. They consist of three types of layers; the first one is an input layer, in which the nodes, also called neurons, represent the input variables of the problem; The second type is a group of hidden layers in which the neurons act, thus capturing the nonlinearity in the data; the third and last type is an output layer, in which the neurons represent the variables that it is wished to predict (Svozil et al., 1997). Each neuron of the aforementioned layers is connected to all neurons in the next layer by a weight coefficient, which reflects the importance of the particular feature/signal.

Formally, the feed-forward deep neural network with N_i inputs, N_h hidden neurons and N_o outputs can compute N_o nonlinear functions to obtain the activation value of each neuron, which consist of N_i inputs that are processed by the hidden neurons (Marini et al., 2008).

The architecture of the network is determined by a procedure called “training” in which the parameters of the model are estimated and a function is approximated. In this study it was considered that the relationship between dependent and independent variables is not known analytically, thus the training is done via supervised learning, where a regression function is approximated based on examples. This approximation implies finding the best values of the weights that minimize the loss function that represents the distance between the predicted and the measured values. The most widely used method to perform this

minimization and the used in this study is called back-propagation. This is used to perform an iterative weight update by examining a number of epochs in which a search of the error surface is made by using gradient descent for minimum error points. In each epoch, a solution is produced and then the error is propagated backwards to update the weights (Marini et al., 2008). During this process, the weights of the network are updated in accordance with equation 4:

$$\Delta w_{ji}(t) = -\eta \frac{\partial \mathrm{E}}{\partial w_{ij}} + \mu \Delta w_{ji}(t-1) \quad (4)$$

In equation 4, η represents the learning rate and μ a constant, called momentum. The former governs the step size as the algorithm updates the weights. This happens because, if the steps are too small, the algorithm converges slowly, but if the steps are too large, the algorithm is unstable, and therefore tends to oscillate and may not converge. The latter helps the algorithm to avoid becoming stuck in flat spots and is only determined by experiments.

After the training procedure, a validating set of examples is used to test if the network is able to be effective on the data that has not been used in the previous process. Since the model is built for a classification task, a particular pattern recognition process called class-modelling is performed, in which each class is analysed. In this process, it is verified whether the input vector measured on a new sample is compatible with the model of that specific class (Marini et al., 2008). Besides the regular architecture of feed-forward deep neural network, the algorithm proposed in this study includes a constraint that ensures the network avoids classifying an order as “e-tailer” when this player does not have enough inventory to meet all the SKUs demanded.

3.4 ASSESSMENT OF IR-DNN

The criteria of Accuracy and F1-score were used as measures to assess the performance of the IR-DNN. Other measures such as Total Margin and order fulfillment rate (OFR) were used as productivity and service-oriented goals, respectively. They offer managers useful and factual information about the performance of the warehouse, rather than just leaving managers to make intuitive assessments of the results. Besides, they make it feasible to compare the

IR-DNN with other rationing policies that cumulate the demand for a certain period of time, which can be evaluated using the same criteria.

3.4.1 Accuracy

In order to verify if the DNN performs well regarding the order classification task, the accuracy of the model must be measured. This is defined as the fraction of predictions that the model classified correctly. Equation 5 defines the mathematical expression for binary classification:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (5)$$

If the class called “e-tailer” is established as negative and the class called “supplier” is established as positive in the output of the DNN, then the orders correctly classified as “supplier” are true positives (TP), the ones wrongly classified as “supplier” are false positives (FP), the orders correctly classified as “e-tailer” are true negatives (TN), and the ones wrongly classified as “e-tailer” are false negatives (FN).

3.4.2 F1-Score

In cases where there is a data imbalance (*i.e.*, more instances of one class than of the other), using only accuracy as a reference can give the naive impression that the predictive power of the model is high for both classes, when in fact it may be very poor for one of them. It happens because the algorithm tends to sample and weight the classes according to the amount of data in both classes and then, it ends up classifying the majority class accurately and the minority one very poorly.

In the B2C context, the class distribution depends on the demand distribution. Therefore, splitting demand in a dropshipping system is generally considered as an unbalanced problem, since the e-tailer should fulfill most of the orders, while the supplier is asked to fulfill orders when the e-tailer faces stock-out or decides not to meet some demand. The distribution parameters of the two classes are presented in the simulation section.

In the case of this study, it is possible to evaluate the classification task by using the F1-Score. It tracks how the model deals with the class imbalance by

assessing the tradeoff between the Precision and Recall in the minority class. The first one is related to the proportion of orders that the model assigns correctly to a given class among all the orders that the model assigns to that class. On the other hand, the recall refers to the proportion of orders that the model assigns correctly to a given class among all the orders that actually belong to that class. Equations (6) to (8) present the mathematical expression for precision, Recall, and the F1-score, respectively. These equations show how to calculate the F1-score for the class “supplier” which is the minority class. However, it is also possible to recreate the equations in order to calculate metrics for the class “e-tailer” in order to check the model performance regarding this class after the optimal model has been selected. Equations (9) and (10) present the mathematical expression for Precision and Recall for the class “e-tailer”.

$$PRE_{supplier} = TP/(TP + FP) \quad (6)$$

$$REC_{supplier} = TP/(TP + FN) \quad (7)$$

$$F_1score_{supplier} = \frac{2 * (PRE_{supplier} * REC_{supplier})}{(PRE_{supplier} + REC_{supplier})} \quad (8)$$

$$PRE_{e-tailer} = TN/(TN + FN) \quad (9)$$

$$REC_{e-tailer} = TN/(TN + FP) \quad (10)$$

3.4.3 Order fulfillment rate

The order fulfillment rate (OFR) is a service-oriented indicator that expresses the proportion of customers that the e-tailer can serve with the on-hand inventory in a particular shift. The OFR is calculated as shown in Equation (11).

$$OFR = (F/N) \quad (11)$$

3.4.4 Total Margin

The total margin M_t aims to measure the productivity performance of the fulfillment operation. It is defined as the sum of the margin M_{int} obtained by fulfilling orders from the e-tailer's inventory and the margin M_{drop} obtained from the drop-shipped orders. At the same time, the margin obtained from each fulfillment player depends on the unit margin of each SKU ordered. Equations (12) to (14) show their respective mathematical expressions:

$$M_t = M_{int} * + M_{drop} \tag{12}$$

$$M_{int} = \sum_{j}^{n} \sum_{i}^{m} m_{i.int} * d_{ij} \tag{13}$$

$$M_{drop} = \sum_{j}^{n} \sum_{i}^{m} m_{i.drop} * d_{ij} \tag{14}$$

3.5 SIMULATION AND OPTIMIZATION

The performance of the IR-DNN was investigated in a numerical example. First the model was compared with a benchmark policy considering the case in which the e-tailer offers 350 different SKUs and receives a daily demand of 210 orders. Secondly, a sensitivity analysis is conducted varying the number of SKUs offered (e-tailer size) and the number of daily orders (demand). Some assumptions regarding the fulfillment operation are presented in Table 3.

Table 3—Assumptions of the numerical application

Assumption 1	The dropshipping strategy is composed of two players: one e-tailer and one supplier. The usage of more players should be investigated in other studies.
Assumption 2	The e-tailer offers first necessity items such as pharmacy, food and cleaning products which have delivery windows up to 24 h.
Assumption 3	The e-tailer's items are divided into 3 classes (A, B and C) representing the frequency of demand. The probabilities p_i are: p_A = 0.1, p_B = 0.3, p_C = 0.6 (Jiménez et al., 2019). The rising probabilities represent the long-tail phenomenon of the business-to-consumer market in which approximately 80-90% of the SKUs (B+C) are highly demanded in very low quantities and account for approximately 80% of the revenues (Hompel & Schmidt, 2006; Anderson, 2016). This

	concept is represented altogether by the parameters set in Assumptions 1 and 2. However, more product classes and different probabilities can be set.
Assumption 4	The possible demand of items for each SKU follows a triangular distribution. According to Jimenez *et al*. (2020), it is adequate when the distribution of random variables is inaccurately known. Accordingly, the pessimistic demand for SKUs is 1, the most likely demand is 2, and the maximum demand is for 6 items.
Assumption 5	The initial inventory level of each SKU for a particular shift follows a normal distribution, with 15 items as the mean and 2 as the deviation (Jiménez et al., 2019).
Assumption 6	Each operating shift starts without back orders, since these orders have been fulfilled either by the e-tailer or by dropshipping. This last operation is performed by a supplier with a high fulfillment capacity considered here as infinite for practical reasons. This high capacity is the consequence of the supplier´s additional investment in picking and packing capacity (Ayanso et al., 2006).
Assumption 7	When orders are drop-shipped, a decrease of 30% in the margin is inferred, which represents the cost of the dropshipping service (Jiménez et al., 2019).
Assumption 8	We consider that the delivery windows of the players are short enough to not consider delivery time as an important criterion for splitting demand between players.

Source: The Author (2021).

The IR-DNN model was built, trained and validated in the Python environment 3.6 version, using TensorFlow and Keras APIs. The data pre-processing, as well as the training and validation processes are presented as follows.

3.5.1 Data pre-processing

The data pre-processing included data scaling and correcting imbalances between the two classes. Data input such as demand per item, available inventory and margin where standardized *i.e.*, scaled between 0 and 1, in order to improve the power of the optimization process, namely the gradient descent, in the training process. On the other hand, the class imbalance was corrected in order to balance the sampling in the training process. This is necessary because, as the class “e-tailer” fulfills more orders naturally than the class “supplier” in each batch, the “supplier” class presents less instances in the dataset and thus is under sampled. In order to fix data imbalances, a class weight method was applied. This method is intended to assign a higher weight to errors committed in the minority

class in order to guide the training process to a more fear error minimization for the classes (Huang et al., 2013).

3.5.2 Training and validation of IR-DNN

The model was trained on a dataset that represents the players operating during a 50-day period, receiving 200 orders per day and offering 350 different SKUs, for a total of m=10000 instances. 90 % of the dataset was used for the training and testing process while the remaining 10% was used for validation. The training process used an optimizer called “Adam” which minimized the loss function and performed the DNN weight update. Adam optimizer is commonly used in deep learning applications due to four main characteristics: capacity for dealing with large data and parameters, high predictive power, high computational efficiency, and low memory usage (Kingma & Ba, 2014). It is important to note that the training was conducted after the datasets were labelled by means of the model of Jimenez *et al*. (2019), which take daily batches from the dataset and separate the order between the supplier and the e-tailer according to the dynamic critically index.

During the training process, the hyperparameters were tuned: the number of layers and the number of neurons were optimized using a cross-validation technic with 7 folds, which means that 90% of the dataset was split in 7 parts. Thus, one part was used as the test set and the remaining six parts were used as the training set. In that way, the model was trained setting the test set as each one of the possible 7 folds in order to assess the generalization capacity of the model. The number of epochs was optimized using the early stopping method, which allows the training process to stop when the validation error reaches its minimum which avoids overfitting the DNN (Prechelt, 1998).

As the DNN gives the probabilities of belonging to one class or another using a sigmoid activation function in the output layer, the threshold of the classifier must best associate each order to each class. In order to define this threshold, it was necessary to define which type of error is more important for the problem, is it FP or FN? If we commit FP, the orders that are supposed to be classified as “e-tailer” are classified as “supplier”, in which case, orders with a good size-margin relation are lost to the e-tailer so there are drawbacks both for

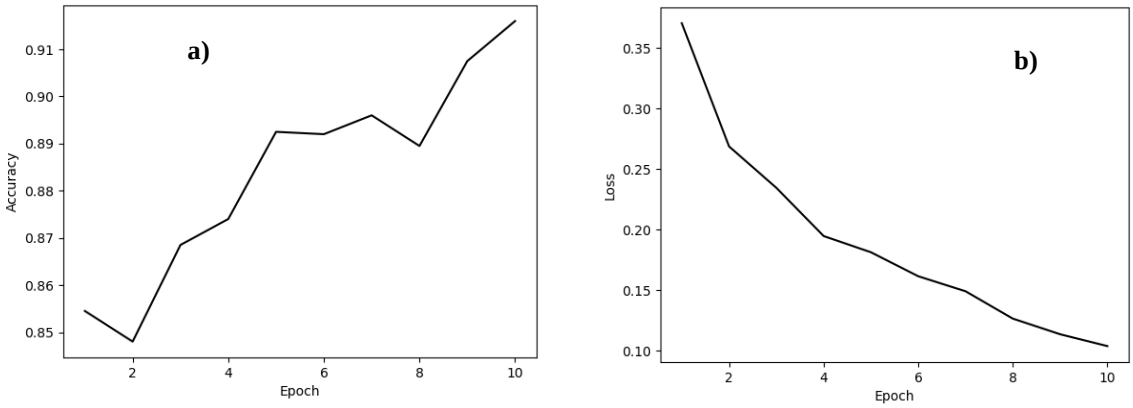
the margin and the inventory commitment. The same effect is obtained when we commit FN, because the e-tailer receives orders that have a poor size/margin relation so they take the place of orders that would be better for him. Despite the fact that the effect of the two errors is the same for the e-tailer, when an FP is committed, the supplier can fulfil the orders without being concerned about depleting the inventory because the capacity of the supplier's inventory is large. The same does not hold true for the e-tailer. When an FN occurs in an order for which the e-tailer has no inventory (or has less than the demanded amount), the e-tailer should postpone the order, which is in fact what the modeld of f Jimenez *et al*. (2019) and the proposed model try to avoid. Considering the latter, FN errors are the most undesirable, so the classification threshold was set at 0.8 meaning that just the orders that have 0.8 of probability of belonging to the class "supplier" or more, were scored as "supplier" during the training process.

After the IR-DNN model was trained and validated in terms of accuracy and F1-score, its performance in terms of the OFR and profit was compared with the FIFO rule. In this rule, the e-tailer gives priority to orders that arrive first until the inventory is exhausted for some item in the order, and then outsources the remaining orders. The comparison was made by performing the two strategies during a 30-day period with a daily batch of 200 orders that are not part of the dataset used for training and validating the IR-DNN. Results are presented as follows.

3.6 RESULTS AND DISCUSSION

After the IR-DNN had been optimized, it was found that the architecture that offers the highest F1-score corresponds to a network with 2 hidden layers, with 50 neurons in the first layer and 60 neurons in the second layer. The model was trained during 10 epochs with Dropout values of 0.3 for the first layer and 0 for the second one. Figure 2(a) and 2(b) present the training process with the best hyperparameters in terms of the accuracy and the evolution of losses.

Figure 2—Training process. a) Accuracy evolution. b) Loss evolution



Source: The Author (2021).

The optimal IR-DNN model has an F1-score of 0.641 and an accuracy of 0.912% on the test set, indicating that 91.2% of orders are classified correctly in relation to what the model of Jimenez *et al*. (2019) consider as efficient in terms of productivity and service-level performance. Although the optimal model achieves the best trade-off between precision and recall for the minority class “supplier” by optimizing the F1-score, it is possible to assess the model performance in terms of the precision and recall for both the classes “supplier” and “e-tailer” in order to verify how the model performs in each of them. These metrics are presented in Table 4.

Table 4—Results

Class	Precision	Recall
Supplier	0.615	0.669
E-tailer	0.955	0.944

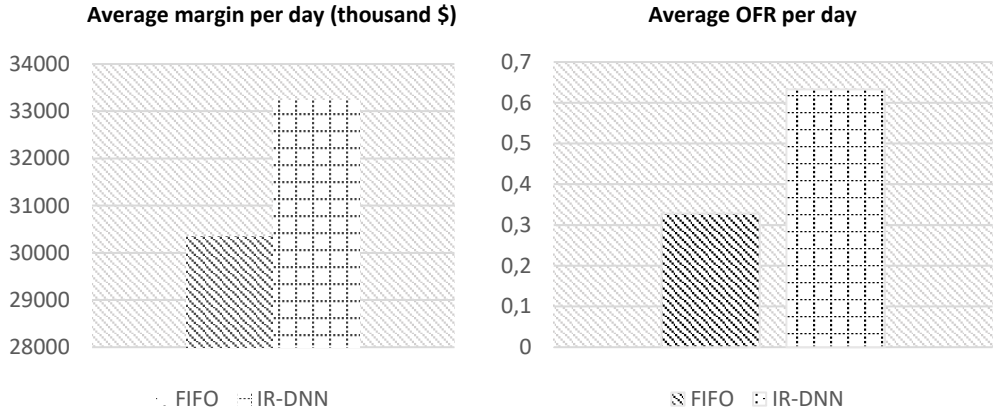
Source: The Author (2021).

From Table 4, it can be seen that the precision of the model regarding the class supplier is 0.615 which indicates that the model correctly classifies 61.5% of the orders that it considers should be fulfilled by the supplier. In that case, precision also represents the ability of not classifying as “supplier” an order that is “e-tailer”. Regarding the class e-tailer, the model correctly classifies 95.5% of the orders that it considers should be fulfilled by the e-tailer. In that case, precision represents the ability of not classifying as “e-tailer” an order that is “supplier”. When it comes to analyzing the recall of the model, it can be seen that of all the

orders that in fact should be fulfilled by the supplier, the model can classify 66.9% of them. In that case, recall represents the ability of the model to find all the orders that actually belong to the class “supplier”. In case of the e-tailer this percentage is 94.4%, and it represents the ability of the model to find all the orders that actually belong to the class “e-tailer”. From these results it is possible to claim that the model separates the two classes well.

In addition to the F1-score and accuracy performance of the IR-DNN, it is important to check the performance of the model in terms of the e-tailer's productivity and service-oriented goals, which are ultimately measures that can bring more useful information to managers. Figures 3(a) and 3(b) show the comparison between the IR-DNN model and the FIFO rule, regarding the average OFR and the average total margin for a batch of 210 orders during 30 days of operation. In the FIFO policy, the order planning was carried out at the end of the operating shift when the order batch was cumulated every day. On the other hand, in the IR-DNN model, the order planning was carried out simultaneously as the orders were entered into the system.

Figure 3—Performance of IR-DNN model in comparison with FIFO policy. a) Margin performance, b) OFR performance



Source: The Author (2021).

From Figures 3a and 3b it can be inferred that the IR-DNN model outperformed the FIFO strategy by 9.49% in relation to the margin and by 90.01% in relation to the OFR, thus allowing a more profitable and service-oriented fulfillment operation. The IR-DNN presented an OFR of 0.6328 which indicates that the e-tailer is capable of fulfilling 63.28% of the total demand. This can be

regarded as an outstanding performance, since it also indicates that the e-tailer has more control over the fulfilment of the order. It is very interesting in contexts where the contractual relation between e-tailer and supplier is not clear, so there is a risk that the supplier may steal customers. Another advantage of having more control over the fulfilment process is that the risk of the supplier being delayed is diminished. It is considered that the performance of the IR-DNN in terms of margin and OFR is satisfactory considering that the model has access to the information about the order that is being placed at a specific moment in time only, and that the decision over who should fulfil the order is taken immediately.

3.7 SENSITIVITY ANALYSIS

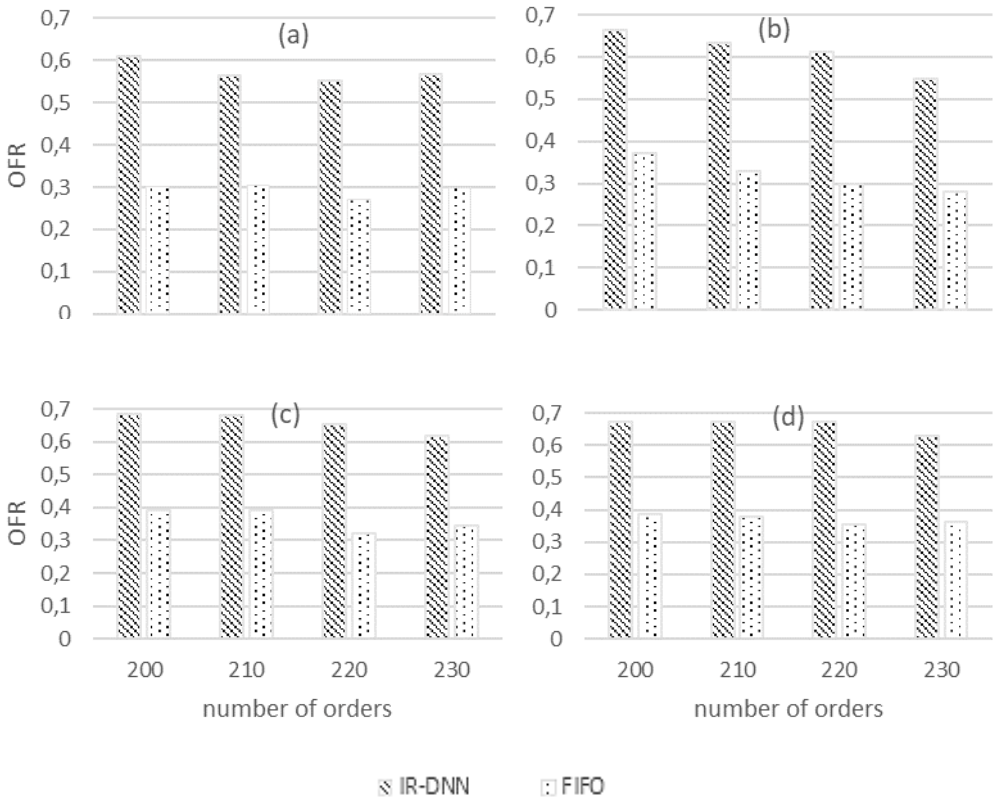
The efficiency of the IR-DNN model in comparison with the FIFO rule was tested in a sensitivity analysis varying the number of SKUs offered (e-tailer size) and the number of daily orders. For the first factor, 300, 350, 400, and 450 SKUs were tested, and for the second factor, 200, 210, 220, and 230 orders were tested. The IR-DNN model was trained for each combination of demand and e-tailer size using a 50-day period dataset, which constituted datasets of 10000, 10500, 11000, 11500 instances for cases with 200, 210, 220, and 230 orders, respectively. Once the model was trained, the performance in term of the OFR and Profit was investigated in new orders during a 30-day period and compared with the FIFO policy performance. The OFR and Margin performance of the models are presented in Figure 4 and 5, respectively.

From Figures 4 (a-d) and 5(a-d) it is possible to see that the IR-DNN model outperformed the FIFO policy in terms of the OFR and Margin regardless of the demand and the e-tailer size. The IR-DNN provides OFR gains from 79.9-102.7% and Margin gains from 5-13.1%. If one looks at each graph in Figure 6 it is possible to see also that the OFR decreases as the number of orders increases for each of the e-tailer sizes. This is an expected behaviour that occurs both for the IR-DNN and the FIFO policies because when the number of orders being fulfilled increases and the initial available inventory remain the same, then it becomes more and more difficult to manage the inventory for fulfilling orders. It can be proven by looking at Figure 5, in which even though the OFR decreases as the number of orders increases, the margin increases. It occurs because more

orders arrive to the warehouse and are fulfilled by the supplier, thus contributing for the total margin.

On the other hand, if we compare all the graphs of Figure 4 sequentially from 4(a) to 4(d), it is possible to see that the OFR increases as the e-tailer´s size increases. This is also an expected behavior that occurs in both policies because as the variety of SKUs offered increases, the variety of SKUs that compose orders increases as well and then the stock of each SKUs become less committed and it can be used more efficiently to attend more orders. A possible way to verify that is to look at Figure 4 and verify how the margin of both policies increases as the e-tailer size increases, indicating that more orders are being fulfilled with the same inventory.

Figure 4—OFR performance for FIFO and IR-DNN policies with variation in the number of SKUs and orders received. a)300 SKUs, b)350 SKUs, c)400 SKUs, d)450 SKUs

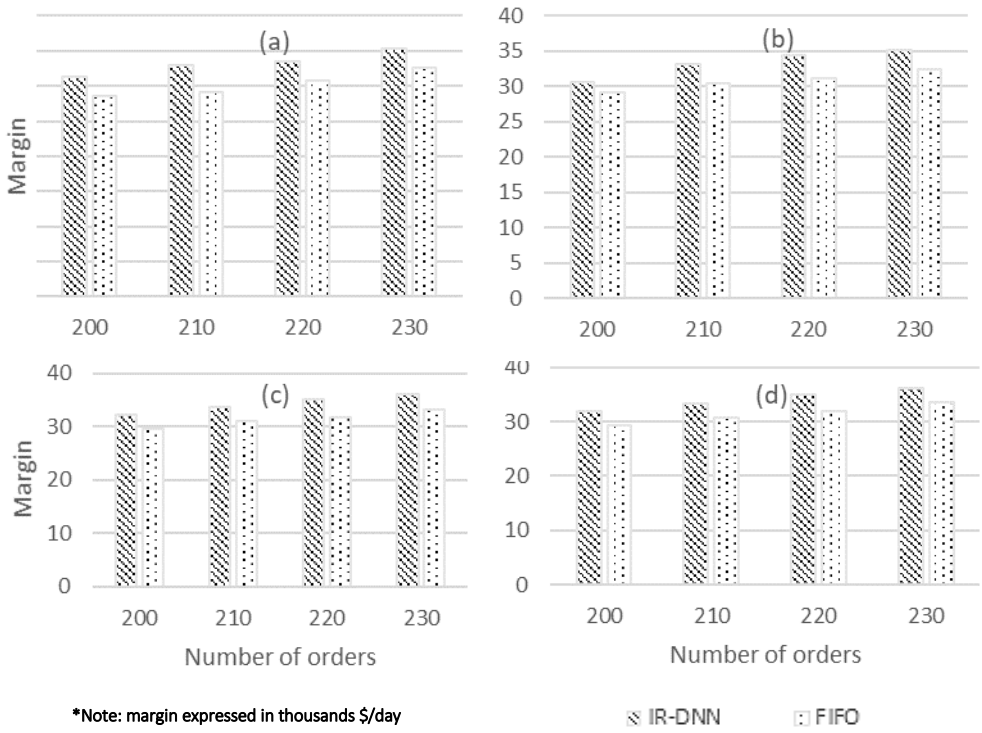


Source: The Author (2021).

It is important to consider that the result presented in relation to OFR and Margin is only the immediate result of allocating orders with the operating parameters (unit margin) with which the system was working at the time of the study. However, these results can be improved in the medium term after implementing the proposed model, which is a reflect of the improvement in the efficiency of operations along the supply chain such as the picking, packing, delivery of materials, and inventory replenishment planning. Therefore, they are not intended to fully measure the benefits of using the IR-DNN model, which are extended to several dimensions presented in the next section.

The effect of having information in advance about the order fulfillment actor is not fully measured in this study, because, besides this being more visible in the medium term, it involves more than one echelon in the supply chain and the operations linked to each of them, so it deserves a more in-depth study. For this reason, researchers in the field of contractual models in supply chains are encouraged to estimate what the strategic and financial gain would be in addition to the immediate gain already demonstrated in the numerical example of this paper. Studies on how to share these gains along the supply chain actors are also encouraged.

Figure 5— Margin performance of FIFO and IR-DNN policies with variation in the number of SKUs and orders received. a)300 SKUs, b)350 SKUs, c)400 SKUs, d)450 SKUs



Source: The Author (2021).

3.7.1 Managerial implications

By implementing the proposed model, the managers should be able to advance the decision making over which actor should attend each order. Advancing this decision making into one complete shift can be translated into greater efficiency not only in the e-tailer's operations as shown in the results, but also along the supply chain activities such as picking, packaging and order delivering in the medium term. This can lead to reductions in the cost of operations, in the final price of the products, or alternatively to an increase in the margin of each product/order. Having information in advance about the fulfillment of orders, also allows a better coordination of inventory replenishment systems, which at the same time enables stock-outs to be diminished and consequently more orders to be fulfilled. This last implication is very important because it allows the problem of high demand variability in the e-commerce environment to be tackled directly.

After implementing the model, it is vital to keep training it on new data in order to maintain the model's efficiency. Thus, it is necessary to carry on a deployment study in order to determine the best training interval that directly depends on the demand variability and the amount of SKUs offered by the e-tailer, which impact the model complexity and thus the amount of data that is needed to re-train the model. As a consequence, human and technological resources should be destined to this end.

3.8 FINAL REMARKS OF THE CHAPTER

This chapter proposed a DNN-based approach that allows e-tailers to optimize the decision about how to split the demand between players in a dropshipping system in order to better manage stock-outs. The model consists of a DNN that learns how to classify orders between players from a dataset of historical orders labeled by the inventory rationing model of Jimenez *et al*. (2019). The integration of the DNN and the aforementioned model allows to allocate the orders with the best balance between order margin and the inventory consumption level of orders for the e-tailer in order to provide more profit and level-service opportunities for it. This integration also enables the advancing of the demand splitting decision in near real time, since the classification task is done as soon as orders enter on the internet.

From this chapter it is possible to conclude that it is possible to offer more profit and service-oriented opportunities to e-tailers by smartly rationing the inventory and collaborating with other players in dropshipping systems. The model offers gains of up to 13.1% and 106% regarding the Margin and OFR, respectively, for different demands and types of SKUs offered when compared to traditional rules such as FIFO. Moreover, by allowing the demand splitting decision to be made earlier, the model also offers opportunities for improving the efficiency of operations in the supply chain links involved, such as the coordination of inventory replenishment systems, and the picking, packaging and delivery of products/orders.

Despite the outstanding gains in the OFR and Margin provided by the proposed model, the consequences of making the demand splitting decision in advance must be measured in economic terms more carefully. This lets managers

establish the advantages of implementing the proposed model in the medium term in a more accurate way. This is important to establish contractual models of revenue sharing, since the benefits are extended to all the actors of the supply chain. They are at the same time responsible for taking advantage of the in-advance information and making use of it in order to contribute to the overall efficiency of the activities of the supply chain.

4 DEEP REINFORCEMENT LEARNING MODEL FOR MULTI-CRITERIA OPTIMIZATION OF PREVENTIVE MAINTENANCE IN MULTI-STATE MULTI-COMPONENT SYSTEMS

Despite the great amount of works for maintenance optimization of multicomponent systems, there are few models that approach multi-criteria maintenance optimization. In this chapter a Reinforcement Learning-based model for multi-objective-maintenance decision-making optimization is proposed. The proposed model is able to identify optimal maintenance actions by maximizing the expected long-term system performance, which is related to both the expected reliability as a measure of worker safety, and the maintenance cost. The proposed model is developed by using a jointly Double Deep Q-Learning algorithm (DDQN) and Goal Programing optimization (GP). The first one allows considering the uncertainties of the system degradation process in multi-component systems and a large space in maintenance decision-making, especially under imperfect maintenance actions. The second one helps to model the reward function of the DDQN agent with multi-criteria optimization. The behavior of the optimal RL agent is analyzed in order to understand its rationality regarding prioritized component and actions. Finally, the performance of the RL agent is compared with four control limited based maintenance policies.

4.1 SYSTEM DESCRIPTION AND PROBLEM STATEMENT

4.1.1 System description

Consider a system consisting of N non-identical components. The components are connected in a redundant structure to increase the system's availability.

As the components degrade, each component i visits $m_i + 1$ discrete states including as good as new state, failure state, and m_i-1 degraded states. In that way, let $s_i(t)$ be the state of component i at time t , we get:

$$s_i(t) = \begin{cases} 0 & \text{if component } i \text{ is as goog as new} \\ 1 \leq j \leq m_i - 1 & \text{if component } i \text{ is in a degraded state} \\ m_i & \text{if component } i \text{ is failed} \end{cases}$$

The degradation of components follows a Markov process with the transition matrix presented below.

$$P_i = \begin{pmatrix} p_{00}^i & p_{01}^i & \cdots & p_{0m_i}^i \\ 0 & p_{11}^i & \cdots & p_{1m_i}^i \\ 0 & 0 & p_{22}^i & \cdots & p_{2m_i}^i \\ & & \cdots & \cdots & \\ 0 & & 0 & \cdots & 0 & 1 \end{pmatrix},$$

with $0 \leq p_{uv}^i \leq 1$ and $\sum_{v=0}^{m_i} p_{uv}^i = 1$ for $\forall\, u \leq v\ (u, v = 1 \ldots, m_i)$.

It is assumed that the transition matrix is unknown and the states of components are monitored at regular discrete times T_k, also any maintenance is performed between two successive inspections. Components can be maintained only at discrete times T_k, receiving whether a corrective action or a preventive action. The maintenance actions are assumed to be deterministic, which means that the maintenance operator can perform an action to recover a component of the system to a desired state. If a component fails between two consecutive inspection times ($T_k\ T_{k+1}$], it could be correctively replaced at time T_k, bringing the failed component to be as good as new, or to leave it broken. If the failure of component leads to a shutdown of the system, a downtime cost C_{down} is incurred. If a component is still functioning at discrete time T_k, an inspection operation is then carried out to reveal the current state of the component. Based on the inspection result, preventive maintenance can be performed. It could be a perfect or imperfect preventive action. A perfect preventive action brings the maintained component to be as good as new while an imperfect preventive one can only restore the maintained component to a state between its current state and the new one. In the last two cases, the system incurs in some maintenance costs such as the cost of taking one component from one state to another C_{ijk} and the setup cost C_s. Finally, it is also considered that when maintenance actions are carried out the system is stopped so the components stop degrading in this time.

We considered that the systems being studied are in a remote area or under stressful working conditions in which the failure of the system can bring adverse effects on the safety of workers. Thus, the maintenance manager is not only concerned about the cost of maintenance actions, but about the system reliability performance which impact the system availability and also the worker's safety.

4.1.2 Problem statement

As observed in the literature review, when it comes to optimize maintenance policies, generally, the system is assessed by criteria such as maintenance cost, inventory holding cost, defective product cost, downtime cost, among others, which can be converted into one single criteria, *i.e*., cost. This single measure enables the optimization of one or more of these criteria at the same time. However, it is shown in the literature that in systems such as nuclear power plants, offshore facilities, aerospace components, and Gas & Oil industry, in which there are stressful or remote working conditions, the maintenance policies can be assessed not only by the economic performance of the system but by other factors that are difficult to be expressed in monetary terms and therefore cannot be grouped into one single economic measure (Bend-daya, 2009; Okoh, 2015; Liu et al., 2020). The system reliability performance for example, can be optimized for guaranteeing a high system availability and not just a minimum cost (Liu et al., 2020), and it can also be used to assess other factors such as worker/plant safety in order to measure social and environmental impact of policies. Considering the system reliability in its regular dimension rather than expressing it in monetary terms might demand a multi-objective maintenance optimization when optimized jointly with other criteria. In some cases, the objectives to be optimized are also conflictive, which means that the maximization/minimization of one criterion tends to worse the performance of the other, which might be the case of the system reliability or any other criteria in relation to some of the economic criteria mentioned so far. In this case the multi-criteria optimization model should consider the elements of a decision-making that guide the search for a solution that enable achieving a trade-off between the objectives (Romero, 2014), *i.e.*, the best compromise between them. Such elements include the decision-maker preferences over each objective and a function that enable the joint assessment of the system regarding each criterion.

The multi-criteria optimization model should also consider the stochastic character of the system failure mode in multi-state multi-component systems, which is a consequence of the combined influence of components in different degradation states, the uncertainties in the degradation of components, and in some cases the variability in operational conditions. Therefore, the multi-criteria model should be capable of recommending maintenance actions dynamically at each scheduled

inspection based on the system state in order to contribute to an effective long-term system performance.

Finally, the model should also be capable of being extended to systems with different number of components, multi-state component, and imperfect maintenance actions which can increase the dimensionality of the problem representation exponentially and therefore the computing time.

According to Liu et al. (2020) as in the real-life maintenance applications the economic value may not be the sole performance criterion, a multi-objective CBM optimization based on deep reinforcement learning is an interesting bet. Therefore, to face the aforementioned problems, this work proposed a Reinforcement Learning-based CBM model for the maintenance decision-making and the optimization of multi-state multi-component system with imperfect maintenance actions. The detailed description of the proposed model is presented in Sections 4 and 5.

4.2 METHODOLOGY

As Reinforcement Learning (RL) and Goal Programing (GP) are two main items of the proposed model, in this section, as some principles of RL and GP are discussed.

RL algorithms are efficient tools in the field of maintenance decisions due to their immanent flexibility and performance in learning strategies for real or simulated systems (Kuhnle *et al.,* 2019). In multi-component systems they have the potential of enabling the understanding of the complex relation between components, the effect of combined maintenance actions, and the uncertainties in the operating conditions. Thus, they are able to capture the stochastic behaviour of the system degradation and find optimal maintenance actions that optimize the long-term system performance.

Specifically, DDQN allows the scalability of solutions to high dimensional problems, *i.e.*, problems with large state and action spaces, and enable a considerable decrease in the computational time in relation to other RL algorithms such as Sarsa (y), Q-learning and DQN. It also includes classical optimization approaches such as gradient methods, dynamic programming, integer programming, mixed integer and non-linear programming. Although we do not demonstrate the computational efficiency of our model for different system sizes, the capacity of

DDQN of extending the solution for high dimensional problems is very useful in the multi-component system maintenance context in which the problem representation complexity can be increased as the number of component, number of actions (perfect and imperfect) and degradation states increase. Specifically, Sarsa (y) and Q-learning algorithms can be inefficient for multicomponent systems composed of many components, since they use a tabular representation of the action-state function which is computationally inefficient for large problems, while the DQN algorithms suffers from an overestimation problem for the action values in noisy environments, which can be lessened by the DDQN algorithm by the inclusion of a second network that bring stability in the training process (Cheng at al., 2012).

Since we are considering a problem with a multi-criteria character, *i.e.*, total maintenance cost and system reliability optimization, it is necessary to frame the problem in order to take into consideration the preference of the DM over the objectives. In order to do so, the reward function, which is the ultimate goal that will guide the RL agent is defined using the notion of goal deviations of the GP optimization method. GP is very popular in multi-objective optimization since it enables including many elements of the decision process such as weights, aspirational levels, deviation of variables, rigid and flexible constraints, which enable to find a trade-off between the objectives and represent the preference rationality of the decision maker (Romero, 2014). Coupled to the GP approach, we will use a Deep Neural Networks (DNN) to represent the cost functions. More details on that are given in the numerical section.

4.2.1 RL background

In the RL paradigm an agent solves an MDP problem interacting with a simulated environment and obtaining experience progressively in order to understand the long-term impact of the actions and the benefit of visiting specific states. The agent observes the state S_{tk} at time Tk, selects an action A_{Tk} and performs the selected action to obtain a reward, which is cumulated to the reward of other decision points and analyzed by the agent in order to improve its policy π of acting. In order to select an action, the agent assesses the effect of each action A_{Tk} in the long-term, given a specific state S_{Tk}. Thus, actions are assessed based on the expected return G_{Tk}, which includes the immediate reward R_{Tt+1} obtained for taken an action A_{Tk}, and

a fraction (γ) of the rewards that could be obtained in future states. G_{Tk} also called *q-value* can be expressed as follows:

$$G_{Tk} = R_{Tk+1} + \gamma(R_{Tk+2} + \gamma R_{Tk+3} + \gamma^2 R_{Tk+4} + \cdots) \quad (15)$$

$$G_{Tk} = R_{Tk+1} + \gamma G_{Tk+1} \quad (16)$$

As the agent does not know the returns for any state-action pair when it is put in the environment, the training process is based on learning the optimal returns for each state-action pair, to efficiently evaluate the best course of actions trough a decision window. Then, actions are selected over time, according to the expected performance, based on the current knowledge of the agent, that is improved progressively trough iterations. This training process uses an epsilon-greedy strategy that controls the exploration/exploitation tradeoff, in which a rate of exploration decays at each decision windows in order to gradually increase the exploitation of the environment, until the agent knows the true returns of each state-action pair. At this point, it is possible to find the best policy π, i.e., the best course of actions for a specific system state when implemented. Specifically, when the agent explores the environment, it takes the risk of selecting an action that lead to a worse reward in order to find a solution that overcomes the ones already known. On the other hand, when the agent exploits the environment, it selects the best action that he/her knows so far, *i.e.*, the one that offers the maximum G_{Tk}.

If a DQN algorithm is developed, then an artificial neural network (ANN) is used to represent the state-action value function and perform a function approximation. It means that the *q-values* are predicted at each *Tk* by an ANN based on the current knowledge of the agent, i.e., initial state, actions, final state, and immediate reward experienced at each *Tk*, which serves as a pool for the ANN be re-trained progressively and to offer to the agent q-value predictions in order to take a decision at each *Tk*. The search for the optimal policy is guided by a reference optimality condition called the bellman optimum. It enables the agent to obtain the maximum expected return achievable by any policy for each possible state-action pair. Thus, the prediction power of the ANN is improved progressively by minimizing the loss between the q-values predicted by the ANN and the bellman reference

equation, by backpropagation. Equations (17) and (18) present the bellman optimum and the loss estimation respectively.

$$q^*(S_{Tk}, A_{tk}) = E[R_{t+1} + \gamma * \max_{A_{Tk}'} q(S_{Tk}', A_{tk}')] \quad (17)$$

$$Min\ loss = (q^*(S_{Tk}, A_{tk}) - q(S_{Tk}, A_{tk}))^2 \quad (18)$$

In DQN, the more recent state-action pairs and the respected bellman optimums are calculated by the same ANN between two successive runs, which turns the bellman optimum into a non-fixed target, bringing instability to the learning process. In order to overcome this issue, we use a DDQN algorithm, which includes a second ANN called “the target network”, which froze the weight’s ANN temporarily. For more details, please refer to Hasselt *et al.* (2015)

4.2.2 GP background

Goal Programming (GP) is a method used to solve multi-objective problems that deal with conflicting objectives. It minimizes deviations with respect to those objectives to find an adequate trade-off between them (Romero, 2014). A typical problem modelled by GP is presented in equation (19):

$$Min\ Z = \sum_{k=1}^{r} W_k\,(d_k^- + d_k^+)/P_k$$

Subject to:

$$C * X + d^- - d^+ = F \quad (19)$$
$$A * X + d^- - d^+ = B$$
$$X, d^-, d^+ \geq 0$$

In equation (19), *F* and *B* are soft objectives that the decision maker would like to achieve, but that can be violated, e.g., maintenance budget. Therefore, *d*+ and *d*- are degrees in which objectives are either violated or well performed. On the other hand, *X* are hard constraints that cannot be violated, e.g., maximum number of hours per shift. The soft constraints are the core in this approach since they allow the designer dealing with the trade-off between objectives.

GP is an interesting approach, since it contains many elements that allow the designer to model the decision maker preferences and perceptions about the relation between the criteria (Romero, 1991). Elements such as the aspiration levels (*F*, *B*), which are desired performances that the decision maker wants to achieve, but given to the conflicting nature of goals, they can be violated. *C* and *A* are functions representing the real objectives of the problem. They compose the soft constraints that try to achieve the aspiration levels as much as possible. It gives the DM the possibility of expressing soft goals as unilateral superior goals when constraints can be violated getting a value below the aspiration level (e.g., flexible revenue goal), unilateral inferior goals when the goal can be violated by getting a value above the aspiration level (e.g., flexible cost goal), and bilateral, when deviations are minimized around the aspiration level without forcing the value to be exactly equal to the aspiration level (e.g., flexible budget goal).

Since *d*+ and *d*- are deviations from the aspiration levels, and generally the soft constraints are expressed in different dimensions, then it is necessary to scale those deviations. It is made by means of P_k which corrects the scale of the deviations, preventing the model of being biased. P_k values are calculated based on aspiration levels and coefficients of variables in the soft constraints, so, these parameters are fixed and they do not change overtime, *i.e.*, no matter the vector state of the system, parameters of the models are fixed. This characteristic is important in the context of multi-state multicomponent-systems in which the feasible set of actions at a specific time T_k is variable and dependent on the current state. Finally, W_k, are the weights that represent the importance of the criteria for the DM (the preference structure). They can have a real meaning, such as the cost impact of deviations, margin impact, illness, impact on the organization's image, or any consequence that the DM judge as being useful for determining the preference for each criterion. These values drive the optimization process, in the sense that they tell how beneficial for the system performance is to improve one objective in a value *x* in comparison to improve other objective in the same amount *x*.

4.3 RL-BASED CBM MODEL FOR MAINTENANCE DECISION-MAKING AND OPTIMIZATION

In order to develop a RL-based CBM model for the maintenance optimization of multi-state multi-component systems it is first necessary to model the elements of the decision problem. They include the state space, the action space, and the effect of actions over the system performance, which can be specified through the definition of a reward function based on GP modelling. These elements are formalized as follows.

4.3.1 State space

Considering the problem defined in section 2, the state of the system observed at every inspection *Tk* can be defined as follows:

$S_{Tk} = [s_i(T_k), s_{i+1}(T_k), s_{i+2}(T_k) \ldots\ldots, s_n(T_k)]$

where each element of S_{Tk} represents the degradation state of each component of the system. Considering that, the state space of the problem is all the possible combinations of *n* components in all possible degradation states *m* defined by m^n possible state sets.

4.3.2 Actions space

Since we deal with a multi-component system, the action taken at each inspection T_k is a set of actions A_{tk} composed of the actions taken for each component as follows:

$A_{Tk} = [a_i(T_k), a_{i+1}(T_k), a_{i+2}(T_k) \ldots\ldots, a_n(T_k)]$

Considering the last, the action space is all the possible set of actions.

Since we consider that the maintenance actions are deterministic, *i.e.*, they can take the system to a desired state, from now on, when we recall actions we will referring to the state of components after performing maintenance, *e.g.*, if one wants to take a n-component system from $S_{T_k} = [020..n]$ to $S_{T_k}(post) = [010..n]$, then the initial state of the component is 2, and the action taken over it is 1. Formally, we have $A_{T_k=}S_{T_k}(post) = [010..n]$,. Besides, we consider that induction of defects or heterogeneous population of spare parts are not considered. Therefore, actions cannot worse the state of the components. For example, if a n-component system is in $S_{T_k} = [003..n]$ at T_k , $A_{T_k} = [002..n]$ would be a feasible action, but if S_{Tk} would

be $[001..n]$, then the same action would be unfeasible. This characteristic makes the action space variable and dependent on the current state S_{Tk} of the system.

Despite the action space being dependent on S_{Tk}, it is not possible to represent it like that. It happens because the output nodes of the neural network that represents the state-action value function in DDQN should remain constant which correspond to the number of actions. Thus, in order to fix the action space and consider that there are unfeasible actions inside the action space that depend on the current state S_{Tk}, we should first define the action space *AE* for a system with *m* possible states and *n* components, as being m^n possible action sets, and then, this action space should be constraint during the training process, allocating a high penalty to the reward when an unfeasible set of actions is selected. At the end of the training process, the RL agent should be capable of avoiding the unfeasible actions-those that worse the component state-and select the best set of feasible actions that lead to the maximum long-term reward.

It is important to highlight that as we are dealing with an inspection policy, the system passes through two state transitions. The first one is a transition from the current state S_{Tk} of the components to the state after maintenance $S_{Tk(post)}$ which is the same action A_{tk}, while the second one is a transition from $S_{Tk(post)}$ to the state after the natural degradation S_{Tk+1}, which occurs between two consecutive inspection times ($T_k\ T_{k+1}$] when the system is not being inspected. Therefore, S_{Tk+1} is the post-degradation state and the final state observed by the RL agent after performing an action.

4.3.3 GP-based Reward

Reward functions are cost-benefit models, which evaluates how suitable the transition from one state to another is, and what is the long-term benefit of performing a policy π of acting.

Since we are dealing with a multi-objective problem, where a trade-off between objectives must be achieved, the reward function should take into account the preference structure of the decision-maker (DM). The last guides the trade-off between the objectives and at the same time is aligned with the organizational goals. In our problem, we are dealing with reliability as a measure of worker's safety, and maintenance cost that measure the cost of taking each component from S_{Tk} to S_{Tk+1} and the downtime cost. The preferences over these two objectives strongly

determine the optimal policy of the agent since it will determine which actions the agent should take in order to favour one objective over the other. In order to consider the preference structure of the DM we have used a Goal programming approach to model the reward function.

Next, we present how the multi-criteria maintenance optimization of multi-state multi-component systems can be modelled in terms of GP (see Equation (20)). After that we will show how the reward function can be structured in order for it to be compatible with the RL paradigm.

$$Min\ Z = W_1 * \frac{(d_1^{-1})}{P_1} + W_2 * \frac{(d_2^{+})}{P_2}$$

$$subject\ to:$$

$$r_b + d_1^{-1} - d_1^{+1} = R_r$$

$$c_m + d_2^{-1} - d_2^{+1} = C_r \qquad (20)$$

In equation (6), r_b is the reliability at inspection time T_k after performing a maintenance action A_{tk} which represents the probability of the system surviving until the inspection T_{k+1}. c_m is the maintenance cost of A_{tk} at a specific inspection time T_k which includes the cost of taking each component from S_{tk} to S_{tk+1}, the downtime cost C_{down}, and the setup cost c_s. *rb* and c_m are functions that depend on A_{tk}. *Rr* and *Cr* are the aspiration levels of reliability and cost that the manager wants to achieve as reference thresholds. In the objective function, d_1^{-1} is the deviation of *Rr* when r_b underperforms the desired threshold, which is penalized in the objective function. The same logic is used for d_2^{+}, which is penalized when c_m underperforms the desired threshold.

As the maintenance decision in multi-state-multi-component system is solved in this paper by means of RL so to consider its stochastic character, it is necessary to adapt the aforementioned model to the RL model in order to include the preference structure of the DM in the reward function and then guide the agent during the learning process for balancing the objectives. Equation (20) is transformed in the reward function described by Equations (21), (22) and (23):

$$Reward_{Tk} = Rb + Cm$$

21)

$$Rb = \{W_1 * (r_b - R_r)/P_1$$

22)

$$Cm = \{-W_2 * (c_m - C_r)/P_2$$

23)

At each inspection *Tk*, the aforementioned reward function is used to calculate the immediate reward of A_{tk}, and the same is used to calculate the quality of the possible actions, *i.e.*, q-values, given a specific S_{tk}, which in turn drives the decision of which action to select at each T_{tk} when training the RL algorithm. *Rb* and *Cm* are scaled and weighted deviations of the aspiration levels *Rr* and *Cr*. Then, when the reliability level r_b is below *Rr*, it is penalized in the reward function, and when r_b is above *Rr*, it is awarded. The same thing happens with the cost deviation. When c_m is above Cr, it is penalized, and when c_m is below Cr, it is awarded.

Finally, deviations *d*+ and *d*- were scaled using proportional normalization in which deviations are proportional to the aspiration levels. Then we set $R_r = P_1$ and C_r=P_2.

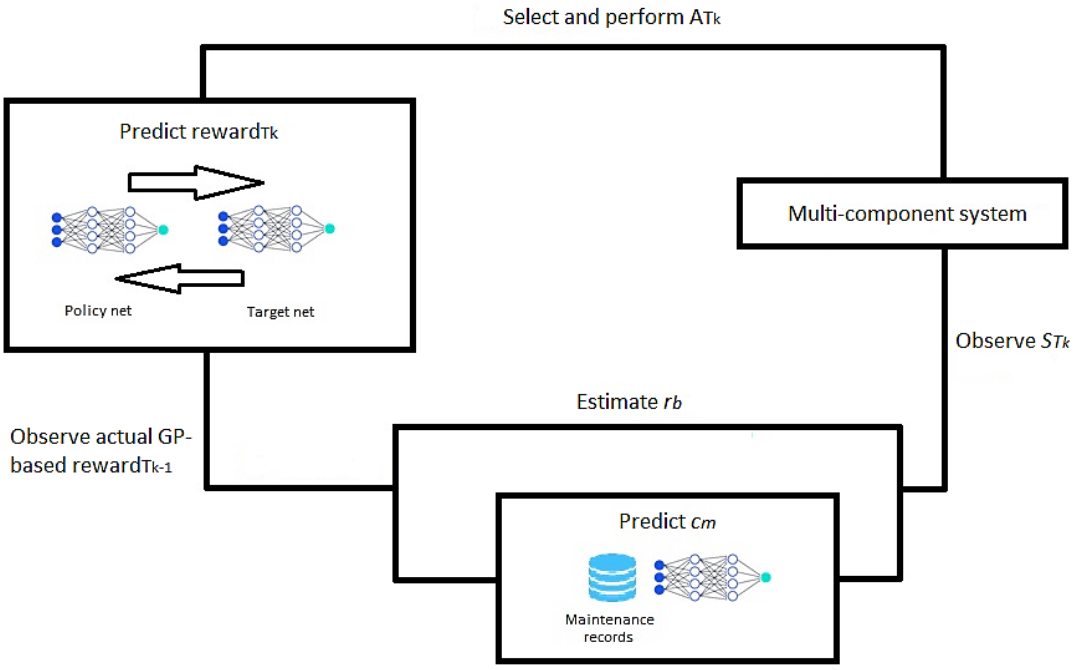
In this paper r_b and c_m are represented by DNNs and predicted by them at each *Tk* when an *Atk* is taken. DNNs map the relationship between *Stk*, *Atk*, and the system performance, and enable predicting the performance of actions between two consecutive inspection times $(T_k, T_{k+1}]$. More details on that are given in the numerical section.

4.3.4 RL training process

In the general training process, the agent follows the steps shown in Algorithm 1 presented in Appendix A. In Algorithm 1, all the components are set to AGAN (state $STk = [0,0,0 \dots,0]$) when each iteration begins (reset MaintenanceEnv). Besides, during the training process, unfeasible actions (those that worse the state of the components) are highly penalized in terms of cost as a way of constraining the action space. Penalties are proportional to the difference between the initial state of a given component and its final state, so the more, the action worse the component the higher it is penalized. Therefore, during the training process, all the q-values are optimized, even the unfeasible actions, but the agent will avoid selecting the unfeasible action as it learns what actions offer a better long-term reward.

Figure 6 summarizes the integration of all the element of the model described in sections 4.2 and 4.3 providing a bird's eye view of the training process.

Figure 6—RL-based CBM model

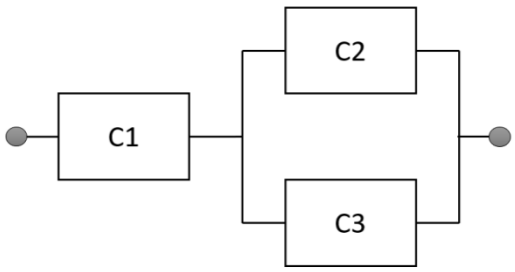


Source: The Author (2021).

4.4 NUMERICAL EXAMPLE

The proposed RL-based CBM model is applied to a simulated system composed of *i=3* components and *m=4* states, defined by $s_i(T_k)$. The reliability block diagram is shown in Figure 7.

Figure 7—Reliability diagram block of the system



Source: The Author (2021).

$$s_i(T_k) = \begin{cases} 0 & \text{if component } i \text{ is as goog as new} \\ 1 \leq j \leq 2 & \text{if component } i \text{ is in a degraded state} \\ 3 & \text{if component } i \text{ is failed} \end{cases}$$

We have considered that the cost relation of components is not known, but its relation to maintenance states and maintenance actions at each S_{tk} can be estimated from records by using Deep Neural Networks (DNN). It is especially useful in multi-component systems in which it is difficult to estimate the performance of the system based on combined degradation states of components and combined actions. Therefore, the DNNs can map these relations in a model-free way.

We have used a feed-forward DNN for predicting the expected system cost performance every time the agent performs A_{tk}. These DNNs were optimized using Adam optimizer, optimizing aspects such as the number of layers and the number of neurons per layers. They were fed with the dataset explained in the next section, receiving S_{Tk} and A_{Tk} at every inspection as inputs, and the system cost as output. Finally, the training set is set at 90% of the dataset instances, while the test set is the remaining 10%.

4.4.1 Data generation

Although the proposed model can benefit from real data to understand the one-step relation between degradation states, actions and the system performance, we have generated a simulated dataset in order to overcome the lack of real maintenance records and show the value of the model. The dataset is generated with the following conditions:

- maintenance inspections were generated for 5000 sequential discrete time points T_k (with k = 1,2, … ,5000), so at each inspection the degraded system is recovered by getting their components maintained;
- all components are new at the beginning $S_{T_k} = [0,0,0]$;
- preventive and corrective maintenance actions were generated for each component with the possibility of performing or not performing any maintenance action on broken or survival components;
- the maintenance cost includes the *Cd*, which is incurred when the whole system fails, the cost of taking each component from state S_{T_k} to state $S_{T_{k+1}}$ and the setup cost *Cs*.

In order to define whether the system incurs in downtime or not, it is necessary to estimate if the system fails based on the reliability diagram shown in figure 2. It is defined by equation (24).

$$F_{sys} = F(1) * [1 - \big(1 - F(2)\big) * (1 - F(3))], \quad 24)$$

Where,

- F_{sys} represents the system state. If the system fails, F_{Sys} = 0 and then the system incurs in *Cd=$500*, otherwise if the system is operational, F_{Sys} = 1 and then *Cd= $0*.
- *F(i)* represents the state of the component *i* at time T_k before performing any maintenance action. It is defined as follows:

$$F(i) = \begin{cases} 0 & if\ i\ is\ in\ state\ (m+1), i.e., failed \\ 1 & if\ i\ is\ in\ state\ 0 \le j \le (m_i - 1)\ i.e., operational \end{cases} \quad 25)$$

The maintenance costs of taking each component from state $s_i(T_k)$ to state $s_i(T_{k+1})$ are defined by the following matrix Mi (for i =1 to 3). Additionally, the setup cost is set as *Cs=20*.

$$M_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 100 & 0 & 0 & 0 \\ 100 & 40 & 0 & 0 \\ 100 & 0 & 0 & 0 \end{pmatrix} M_2 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 80 & 0 & 0 & 0 \\ 80 & 30 & 0 & 0 \\ 90 & 0 & 0 & 0 \end{pmatrix} M_3 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 70 & 0 & 0 & 0 \\ 70 & 30 & 0 & 0 \\ 80 & 0 & 0 & 0 \end{pmatrix}$$

The degradation of each component between two successive inspections (T_k, T_{k+1}] is generated according to the following transition matrixes Pi (for i =1 to 3). In the case of having a real maintenance record, the matrixes can be estimated from the dataset:

$$p_1 = \begin{pmatrix} 0.35 & 0.35 & 0.23 & 0.07 \\ 0 & 0.29 & 0.42 & 0.29 \\ 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad p_2 = \begin{pmatrix} 0.12 & 0.38 & 0.38 & 0.12 \\ 0 & 0.14 & 0.43 & 0.43 \\ 0 & 0 & 0.43 & 0.57 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$P_3 = \begin{pmatrix} 0.28 & 0.28 & 0.44 & 0 \\ 0 & 0.14 & 0.43 & 0.43 \\ 0 & 0 & 0.29 & 0.71 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The resulting reliability at each discrete *Tk* is also generated. As it means the probability of the system surviving until the next scheduled inspection T_{k+1}, it is necessary to take into consideration the probability of each individual component surviving until T_{k+1} after performing the maintenance action, as shown in equation (26).

$$R_{sys} = R_1 * [1 - (1 - R_2) * (1 - R_3)] \quad (26)$$

In equation (12), R_i is the probability that the component *i* survives until the next scheduled inspection T_{k+1}. It can be calculated from the matrix *Pi* for each component as shown in equation (27).

$$R_i = 1 - P_i(s_i(T_k), 3) \quad (27)$$

Where $P_i(s_i(T_k), 3)$ is the probability of the component *i* going from $S_i(T_k)$ to $S_i(T_k) = 3$, before the next scheduled inspection T_{k+1}. A sample of the dataset generated from the 3-component system is presented in Table 5.

Regarding the parameters of the reward function, it is considered that the DM preferences are *W1=1* and *W2=1*. The aspirational goals at each T_k are $R_r = 0.5$ and C_r=$500. Since we consider that the goals are scaled using a proportional normalization in which deviation are proportional in relation with the aspiration levels, we have R_r=P_1 =0.5 and C_r=P_2=500. The hyperparameters of the RL agent such as learning rate, batch size, decay rate, decay function, number of steps to update the target network weights were tuned.

Table 5—Dataset sample

Tk	$s_1(T_k)$	$s_2(T_k)$	$s_3(T_k)$	$a_1(T_k)$	$a_1(T_k)$	$a_1(T_k)$	***Cost***	***Fsys***
1	0	0	0	0	0	0	0	0
2	2	1	2	1	0	2	140	0
3	2	1	3	1	1	0	140	0
4	3	3	1	0	0	1	730	1
5	3	1	3	0	0	0	800	1

6	1	0	1	1	0	1	0	0
7	1	2	1	1	0	0	170	0
8	1	3	2	0	0	0	280	0
9	2	2	2	0	1	0	220	0
.	.	.	.	.	.	.	.	.
10000	2	3	2	1	0	2	150	0

Source: The Author (2021).

The training process is performed iterating over 10000 episodes each one composed of a decision window of 2000 inspections. The proposed RL-based CBM model and the decision environment were modelled in Python 7.0 using Keras API. Experiments were run in an Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz 2.40 GHz with 5,86 GB of usable RAM.

Besides the training process, a simulation of the deployment of the optimal policy is carried out. This simulation is performed in order to identify the possible scenarios on the decision making *i.e.*, which components are prioritized in each system state and which actions are performed on them. This analysis could enable understanding which component is more critical for the system reliability and cost which can help maintenance managers to take better investment decisions over the components and equipment of the system. In order to do that, a numerical example is developed in order to simulate the behavior of the agent when it is already trained. The example is composed of 3000 inspections that work as a real inspection set in which the maintenance operator observes the system's state S_{tk} and uses the optimal RL policy in order to get a maintenance action suggestion A_{tk}. The maintenance operator performs A_{tk}, leaves the system, and returns in the next scheduled inspection T_{k+1} to continue maintaining the system. The propose of this example is to obtain knowledge from the system and understand the relation between components to gain some strategic insights if possible. The performance of the agent is reported in terms of Total cost and Average Reliability as in the training process. It is important to notice the complexity of this analysis can increase with the complexity of the system, but it is worth it to obtain knowledge from it in order to obtain any strategic and managerial insight as explained before.

4.5 RESULTS AND DISCUSSION

4.5.1 Results of the training process

As follows, we present the “optimal” hyperparameters” of the DNN that predicted the maintenance cost at each inspection point, as well as the loss achieved. The main hyperparameters for DNN predictor are reported in Table 6. The system reliability can be calculated from equations (26) and (27). As it is mentioned in the numerical example, the reliability and the predicted maintenance cost at each inspection point are used to calculate the reward of A_{T_k}at each S_{T_k} by means of the based GP reward.

Table 6— Hyperparameters and performance of the DNN

Number of layers	2
Neurons of layer 1	737
Neurons of layer 2	1456
Optimizer	Adam
Dropout of layer 1	0.020
Dropout of layer 2	0.025
Learning rate	0.1231
Batch size	444
RMSE	3.27

Source: The Author (2021).

Table 7—Hyperparameters of the optimal RL agent

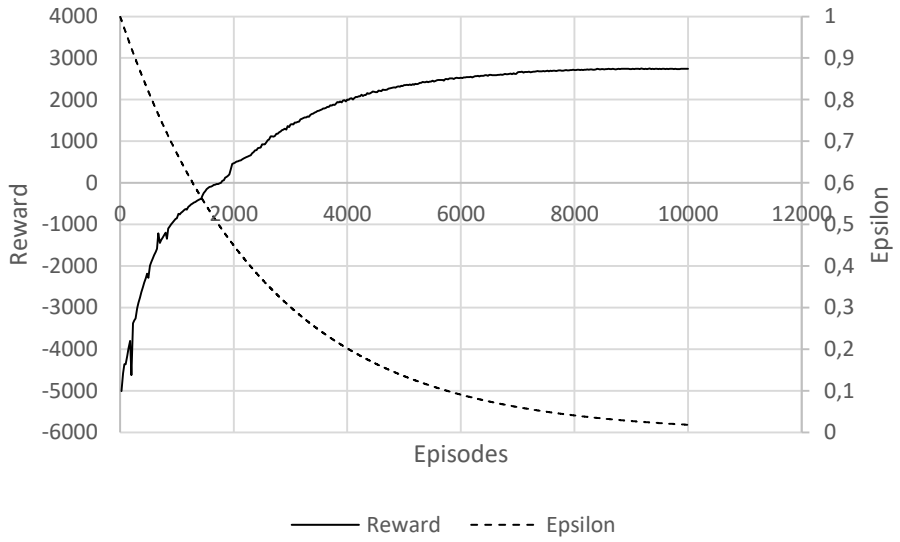
Hyperparameter	Value
Gamma	0.95
Decay rate	0.9994*epsilon
Epsilon minimum	0.005
Batch size	2000
Learning rate	0.0001
Target steps	100
Hidden layers of Target and Policy networks	3
Neurons of layer 1	1200
Neurons of layer 2	300
Neurons of layer 3	100
Optimizer	Adam
Dropout of layer 1	0.1
Dropout of layer 2	0.0
Dropout of layer 3	0.0

Source: The Author (2021).

The main hyperparameters of the optimal RL agent are presented in Table 7. After tuning the hyperparameters of the RL based CBM model the following results

were obtained. Figure 8 presents the long-term reward performance during the training process, *i.e.*, the cumulated reward of 2000 inspections. The results were averaged at each 25 episodes/iterations.

Figure 8—Average long-term reward



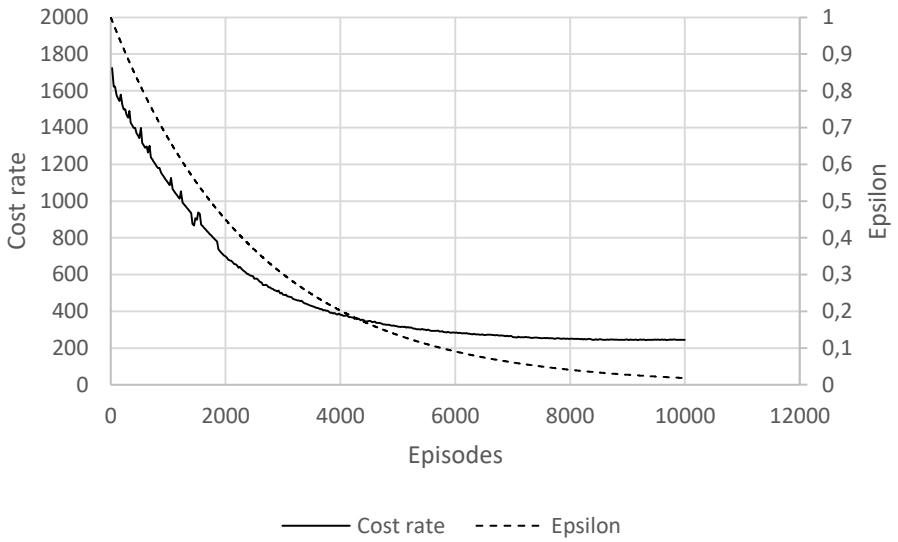
Source: The Author (2021).

From Figure 8 it is possible to see that the average reward has been improved overtime as the agent gains more experience and take it as a base for selecting actions (diminishing of epsilon), *i.e.*, decrease the exploration process and increase the exploitation process. After 10000 episodes of training it reaches the convergence, taking positive values of about 2743.2. From this Figure it is shown that the RL-based CBM model is capable of finding the best trade-off between the maintenance cost and the reliability performance, balancing these two conflicting objectives. It can be verified in Figures 9 and 10 which show how the system performs in terms of maintenance cost and reliability criteria during the training process. These figures offer another view of how the agent performs during the training process.

From Figure 9, it is possible to see how the system cost composed of maintenance, setup and downtime cost is improved over time as the agent gains more experience. The maintenance cost is expressed in terms of cost rate, *i.e.*, the average cost per inspection. The cost rate took a long-term optimum value of 244.05 monetary units at the end of 10000 episodes.

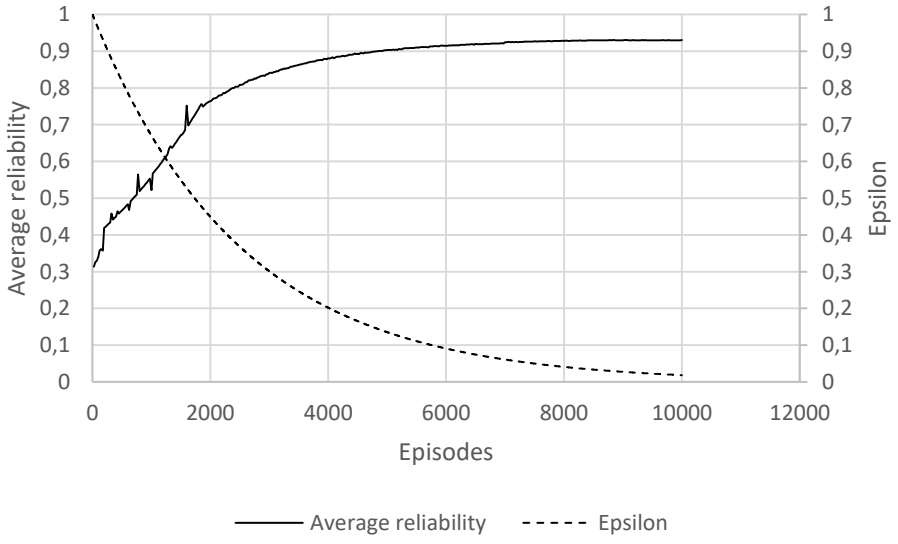
Figure 10 shows the improvement of the average reliability overtime, reaching an average value of 0.93 at the end of the training process.

Figure 9—Average long-term cost



Source: The Author (2021).

Figure 10—Average reliability



Source: The Author (2021).

4.5.2 Deployment simulation results

After simulating the optimal policy of the RL-based CBM model for 3000 inspections, it was possible to identify eight (8) actions that drive the policy of the agent. They are presented in Table 8. In each line, it is possible to see all the possible states $s_i(T_k)$ for each component *i*, and the action $A_{T_k} = [a_1(T_k), a_2(T_k), a_3(T_k)]$ taken for maintaining each component.

Table 8—Illustration of the policy in specific cases

Case	$\boldsymbol{s_1(T_k)}$	$\boldsymbol{s_2(T_k)}$	$\boldsymbol{s_3(T_k)}$	$\boldsymbol{a_1(T_k)}$	$\boldsymbol{a_2(T_k)}$	$\boldsymbol{a_3(T_k)}$
1	0,1,2,3	0,1,2	0,1,2,3	0	0	0
2	0,1,2,3	3	0,1,2,3	0	3	0
3	2	3	2	0	0	2
3bis	3	2	2	0	0	0
4	2	3	0	1	0	0
5	2	2	2	1	0	2
6	1,2	1,2	3	0	1	0
7	0,1	1,2	3	0	0	3
8	3	2	3	0	2	0

Source: The Author (2021).

Case 1 occurs when S_{T_k} is any combination of $s_i(T_k)$ in which *C2* is not broken. In this case the agent takes the whole system to the state AGAN. This is what the agent does most part the time, particularly 41.2 % of the time. This behavior suggests that replacing *C2* is expensive when it is broken, but it is not to recover it to AGAN state through a perfect preventive action. Case 2 occurs when *C2* is broken and the agent keeps it broken. It can indicate that this component is not critical for determining the failure of the whole system or that replacing it is expensive. Once *C2* is degraded, his preventive maintenance is prioritized as seen in case 1, but when it is broken his substitution is neglected. This action is performed 40.6 % of the time.

The behavior of case 1 and 2 might indicate that taking the whole system to the state AGAN is not expensive and that it has a good return in terms of average reliability. When it is not possible because *C2* is broken, a reasonable decision could be keeping *C2* broken most part of the time, since depending on the case it could not

be afforded to get it replaced. Case 1 and case 2 accounts for near 82 % of the action of the agent. In our problem, recovering *C1* and *C2* seems to be enough, since the system is a series-parallel one indicating that having *C3* or *C2* in an operational state jointly with *C1* is enough to keep the system operational.

The remaining six actions are very scarce and account for just 18% of the cases. Case 3 is an exception of case 2 in which the system seems to be near the failure. In this case, the substitution of *C2* that early on is neglected now is prioritized jointly with *C1*, which suggest that the degradation of these 2 components at the same time is critical for the failure of the system and its performance in terms of cost. *C3* does not seem to be that critical since it is kept in *s(tk)=2* even though it is near the failure. The agent behaves like this 4.73 % of the time. It is important to highlight that that combination of $s_i(T_k)$ is particularly critical, since in the case of other similar states such as case 3bis when the system seems to be near the failure, the action chosen by the agent is to replace all the components. It is possible to confirm that recovering *C2* to the AGAN state through a perfect preventive action is cheaper than replacing it when it is broken. Case 3 is very scarce and just happens when despite of the effort of the agent for maintaining all the components in AGAN state, the natural degradation of components between two successive inspections (Tk, T_{k+1}] eventually breaks *C2* and degrades *C1* to *s(tk)= 2*.

In case 4 it is possible to confirm that simultaneous degradation of *C1* and *C2* is critical. It is possible to claim that because when *C2* is broken and *C1* is near the failure the agent is obliged to substituting *C2* again. In this case, *C1* is just taken to $s_i(T_k) = 1$ instead of being replaced. A possible explanation for that is that *C3* is in AGAN state indicating that recovering *C2* is enough to recover the system health and that it can also lower costs. This action is taken 3.2% of the time and is very similar to case 3.

Case 5 occurs just 2.16% of the time, but the scenario is very useful to see how the agent prioritizes components when all of them are near the failure. In this case the action could be justified by a mix among the cost of recovering each component, the importance of each component in structural terms, and the failure rate of each component that could determine how urgent is to recover a specific component. The only thing that can be claimed for sure is that *C2* has the highest priority as it is suggested in other scenarios, while *C3* has the lowest priority. It is true

when *C2* is not broken since it seems like in that case the replacing cost has an important role in the decision.

Case 6 occurs 3.03% of the time. There is not a specific trend in this case, but it seems like the agent is just balancing costs. Case 7 occurs 4.6% of the time. In this case *C1* and *C2* are operational, and they are taken to AGAN state no matter its degradation state which seems to be not expensive. In the case of *C3*, it seems to be not critical, so it is kept broken. A possible explanation of not doing any maintenance in *C3* is that it allows the agent to make some savings and use the budget for maintaining *C1* and *C2* that are apparently more critical in structural terms and cost performance. Finally, case 8 is evidenced 0.4% of the time. In this case *C1* and *C3* are broken and so does the system. Then the agent needs to recover the health of the system, giving priority to *C1* and *C3*.

4.6 COMPARISON WITH CONTROL LIMITED BASED MAINTENANCE POLICIES

A comparison with three control limited based maintenance policies under two different scenarios was performed in order to benchmark the RL-based CBM model. The first scenario represents the case in which broken components ($s_i(T_k) = 3$) should be replaced immediately ($a_i(T_k) = 0$) when the system is found operational at T_k. The second scenario represents the case in which broken components can either be replaced immediately $a_i(T_k) = 0$ or not $a_i(T_k) = 3$ when the system is found operational at T_k. The comparison of the two aforementioned scenarios has been necessary since having one component broken into a system with redundancies does not necessarily mean that the system is down and that the broken component should be replaced immediately. Thus, we can investigate whether providing the policies with more flexibility favor one policy in comparison with the others. For both scenarios, we have three control limited based maintenance policies presented as follows.

1) Corrective based line policy: preventive maintenance actions are not considered and components are just replaced when they are broken ($s_i(T_k) = 3$) depending on the scenario (*i.e.*, scenario 1 or 2). Specifically, $a_i(T_k)$ is chosen randomly for scenario 2.

2) Random based line policy: components are maintained preventively if $0 < s_i(T_k) \leq 2$ and $a_i(T_k)$ is chosen randomly between $[0, s_i(T_k))$

The performance of the random based and the corrective based line policies for scenarios 1 and 2 was calculated using the multi-objective GP-adapted reward, in which it is possible to compute the reward, based on the maintenance cost performance predicted by the DNN and the reliability performance. The decision parameters of GP as well as the cost and reliability structure of the problem solved by means of the RL-based CBM model were kept the same for the aforementioned policies. Considering the last, the RL-based CBM model can be benchmarked with the aforementioned policies in terms of reward, and its compositional elements; cost and reliability.

3) Classical policy: in order to select a component to be preventively maintained, a preventive threshold m_i is introduced. It is necessary since a preventive maintenance action can bring a maintained component to be somewhere between the state just before maintenance and AGAN, and then different types of preventive actions (a_i with i=1, 2, ..., n) would be necessary to be selected. This may lead to many preventive thresholds. Therefore, to reduce the number of preventive thresholds for each component, it is assumed that the state after a preventive maintenance of a maintained component can be uniformly distributed between the state before maintenance and AGAN. That means that only one kind of preventive maintenance action is possible for each component. In that way, we need only one preventive threshold m_i for each component. More precisely, for a surviving component i (with i=1,2,3) at time Tk:

   - A preventive maintenance is needed if its state $s_i(T_k) \geq m_i$, mi being a decision variable to be optimized;
   - No preventive maintenance is required if $s_i(T_k) < m_i$.

To find the optimal preventive thresholds, a cost model is used. Monte-Carlo simulation is used to evaluate the average long-term maintenance cost rate for each value of preventive threshold, in which the reliability performance is a consequence of the cost performance optimization.

In order to make a comparison between the proposed model and an inspection CBM classical policy feasible, it was necessary to adapt the

proposed model as follows: a) the action space is constrained and the agent can only choose whether to perform a preventive maintenance action over a component ($a_i(T_k) = 1$) or not ($a_i(T_k) = 0$), *i.e.*, a binary action set per component; b) as in the classical based line policy the state after a preventive maintenance of a maintained component is uniformly distributed between the state before maintenance and AGAN $[0, s_i(T_k))$; c) the parameters of the GP-adapted reward were set as $W_1=0$,$W_2=1$,$P_2=1$,$C_r=0$ in order to emulate the cost function used to assess the classical based line policy. Given the aforementioned adaptations, the RL-based CBM model can be directly benchmarked with the classical policy in terms of cost, which turns out to be the negative value of the reward, and with the reliability performance which is a consequence of the cost minimization.

The control limited based line policies and the RL-based CBM model were performed over 100 episodes, each one representing the policy performance over 2000 inspections. The results were averaged and presented in Table 9, 10, and 11. Table 9 presents a comparison with the random based line and the corrective based line policy in scenario 1, while Table 10 presents a comparison with the same policies in scenario 2. On the other hand, the comparison with the classical based line policy for scenarios 1 and 2 were condensed in Table 11.

Table 9— Comparison with random based and corrective based line policies for scenario 1

Policy	Reward rate	Cost rate	Average reliability	Computing time* (s)
RL-based CBM	1.3579	248.2646	0.9273	9.8134
Random based line	1.0553	258.8799	0.7866	6.2526
Corrective based line	0.7100	284.8488	0.6376	6.1109

* The reported computing time corresponds to the average time for 2000 inspections

Source: The Author (2021).

Table 10— Comparison with random based and corrective based line policies for scenario 2

Policy	Reward rate	Cost rate	Average reliability	Computing time* (s)
RL-based CBM	1.3716	244.0583	0.9300	9.9608
Random based line	0.9711	269.8184	0.7555	6.4035
Corrective based line	0.5921	301.8333	0.5980	6.1325

* The reported computing time corresponds to the average time for 2000 inspections

Source: The Author (2021).

As can be seen in Table 9 and 10, the three policies were compared in terms of the reward rate, which represents the average reward at each inspection, the cost rate, which represents the average cost incurred at each inspection, the average system reliability at each inspection, and the computing time. It can be seen that the performance of the RL-based CBM model outperforms the performance of the two baseline models in terms of reward rate. Specifically, the random based line model and the corrective based line model were outperformed 28.67% and 91.25%, respectively, for scenario 1, and 41.23% and 64.01%, respectively for scenario 2. The superiority of the RL-based CBM model in terms of reward is achieved with a trade-off between cost rate and average reliability for which the RL-based CBM model also outperformed the two control baseline policies both in terms of average cost and average reliability. The random based line and the corrective based line policies were outperformed in terms of cost in 4.09% and 12.84%, respectively, for scenario 1, and in 9.54% and 19.16%, respectively for scenario 2. On the other hand, the random baseline and the corrective baseline policies were outperformed in terms of reliability in 17.88% and 45.43%, respectively, for scenario 1, and at 23.17% and 55.51%, respectively, for scenario 2. From the aforementioned results it is also possible to claim that the RL-based CBM model benefits from having more freedom for choosing actions. It is possible to say that because the policy outperforms the random and the corrective based line policies to a greater extend in scenario 2, in which the RL agent take advantage of the system redundancy for replacing broken components dynamically whenever it is most convenient. Specifically, the RL-based CBM model promoted a cost improvement rate from scenario 1 to scenario 2 of 133.25% and 49.22% over the random and the corrective based line policies, respectively. In the case of the reliability's performance, improvement rates of 29.58% and 22.18% of the random and the corrective based line policies, respectively were achieved.

From Table 9 and 10, it is also possible to claim that the gain in reward, cost, and reliability performance comes at the expenses of the computing time, which is higher than the random based line and the corrective based line models. It was outperformed 56.94% and 60.58%, respectively, for scenario 1, and 55.552% and 62.426%, respectively for scenario 2. However, we should consider that we are proposing an inspection condition-based model, that will take decisions/advice at

each T_k based on the system state that is observed/informed by the maintenance operator. Then, it is necessary to take into account the resultant average computing time per inspection of the RL-based CBM model, which can be computed as just 0.004498 seconds for scenario 1 and 0.004906 seconds for scenario 2. Therefore, we can say that the use of the proposed policy is justified even with the computing time drawback.

Table 11—Comparison with the classical based line policy for scenarios 1 and 2

Policy	Cost rate Scenario 1	Average reliability Scenario 1	Cost Rate Scenario 1	Average reliability Scenario 2	m_1^*	m_2^*	m_3^*
RL-based CBM**	234.8613	0.8291	229.6096	0.8405	-	-	-
Classical based line*	246.1056	0.7682	256.2817	0.7335	2	3	2

* Thresholds m_1*, m_2*, and m_3* have the same values for both scenarios 1 and 2 **no specific thresholds were found
Source: The Author (2021).

From Table 11 it is possible to see that the RL-based CBM model outperformed the classical based line policy in terms of reliability and cost for both scenarios 1 and 2. Specifically, the reliability's performance was outperformed 7.92% and 14.58% for scenario 1 and 2, respectively. On the other hand, the cost performance was outperformed 4.56% and 10.40% for scenario 1 and 2, respectively. These results confirm the even though inspection CBM classical policies have preventive thresholds that make the policies straightforward for decision making, they tend to become inefficient and suboptimal as the complexity of the problem increases. Another way of confirming that is noticing that when comparing the classical policy performance in scenario 1 and 2, it had a performance loss of 4.13% and 4.51% for cost and reliability, respectively. On the other hand, inspection RL-based CBM policies can map the degradation state of the system to the action space directly without the need of establishing preventive thresholds. As a consequence, the actions on each component can be selected dynamically according to the degradation level of the other components in a given moment. This flexibility characteristic allows better solutions to be achieved. This claim can be also confirmed when comparing the RL-based CBM performance for scenarios 1 and 2, in which it is possible to see that the cost and reliability performance were improved by 2.23% and 1.37%, respectively.

4.7 FINAL REMARKS OF THE CHAPTER

In this chapter a RL-based CBM maintenance model for multi-state-multi-component systems was purposed. The proposed model was developed using Double-Deep-Reinforcement Learning and Goal Programming modelling.

This chapter shows that the proposed model enables maintenance multi-objective optimization while consider the stochastic nature of the system degradation process in multi-state multi-component systems. It allows imperfect maintenance actions to be performed in order to recover the system to better degrades states and to balance the system availability and the maintenance cost. The RL-based CBM model outperformed four limited based line policies achieving cost and reliability gains of up to 19.16% and 55.51%, respectively. From the results it is also possible to say that the RL-based CBM model benefits from having a bigger action alternative set for achieving a better cost and reliability regarding to the benchmark policies. Specifically, the RL-based CBM model achieved an improvement rate over the benchmark policies of up to 133.25% and 29.58 % for cost and reliability, respectively when the two scenarios provided are compared. It was also possible to analyse the optimal policy given by the RL-based CBM model and discover the priority of components and actions based on the frequency of scenarios found in a deployment simulation, which allows one to claim that it is possible to extract knowledge from the agent behaviour (at least for the three-component system) and take advantage of this information to potentially improve maintenance managements in areas such as component life cycle, spare parts management and maintenance investments.

5 CONCLUSSION OF THE THESIS

In this thesis, two machine learning-based models for the optimization of support functions were proposed. The models enable the improvement of key indicator performances such as order fulfilment rate, total e-tailer's profit, maintenance cost rate and average system's reliability

The first model presented in this thesis proposed a DNN classifier that splits the demand between the e-tailer and other players in dropshipping systems in the B2C market. We expect that the model contributes to the improvement of the inventory rationing efficiency of e-tailers, bringing gains in terms of productivity and service-oriented performance. It is also expected to boost the improvement of subsequent activities such as picking, packing, delivering of items, and the inventory replenishment, that benefit from having the demand splitting information in advance. The model has the potential of benefiting e-tailers of critical niches such as the e-grocery market, in which the majority of products are perishable, and cumulating order can be highly inefficient, since the products degrade exponentially as they are kept in shelves. The model can also be adapted to other contexts such as Oil & Gas extractions and wind energy production, which take place in offshore facilities. In this contexts, the on-land inventory is very stressed due to geographical and weather conditions, and the size and cost of spare parts. Additionally, since maintenance plans should be in phase with the availability of components, the way in which the limited on-hand inventory is rationed can affect the availability of the system and consequently the production rate. In this sense, strategically outsourcing some maintenance activities can be an interesting bet for increasing the capacity of response of the in-house maintenance team and the access to spare parts.

The second model presented in this thesis proposed a RL-based CBM algorithm that allows the maintenance optimization of multi-component systems. It considers the necessity/possibility of performing imperfect maintenance and the necessity of assessing the efficiency of maintenance policies regarding conflicting criteria such as plant/worker safety, environmental, and social impact, and any other criteria that allow the integration of other decisions such as production, inventory, and quality control. The model is expected to be applied to a wide variety of contexts of complex systems with different number of components,

imperfect actions, and maintenance criteria. Particularly, the model can be useful in systems such as nuclear power plants, offshore facilities, aerospace components, Gas & Oil industry, and hospitals, in which the consequence of failures can have serious economic, social and environmental implications. The model can also be applied to offshore wind farms, in which the system degradation can be very stochastic due to components and subsystems dependencies, and to the influence of environmental conditions such as wind speed, and wave and storm formation. These environmental conditions also determine the access to the platforms for performing maintenance actions, reason why, imperfect actions can be at hand to take advantage of maintenance opportunities and then balancing the system availability provided by the actions and its costs. Considering the complex characteristics of off-shore wind farms, the design of dynamic maintenance actions based on the asset's conditions such as the provided by the proposed model seems to be adequate.

Since the models developed in this thesis were designed to take advantage of the potential increasing in data availability promoted by the adoption of industry 4.0, the applicability of the models in current real problems represents a challenge. In the case of the maintenance model, its applicability can be facilitated in two ways. The first one is to suppose that the cost relation of components between inspections is known. In this sense, it would not be necessary to use a DNN for mapping this relation. The problem associated with this assumption is that as the complexity of the system increases (number of components, states, actions), it turns less realistic and the cost estimation turns less accurate. A second way around for dealing with lack of maintenance records is to train the model in an on-line fashion. Using this approach, the actions recommended by the agent are performed in the real system, and the cost and reliability consequences would be real as well. In that way, the agent can learn by interacting in near real time with the system. The drawback of this approach is that RL algorithms take too much time and interactions with the system for learning efficient actions. Therefore, the consequence of bad actions during the training process can represent a high cost.

5.1 FUTURE LINES OF RESEARCH

The model presented in chapter 3 considers the case in which there are only two players; the e-tailer and one supplier. Therefore, we recommend to extend the model for more than one supplier/wholesaler in order to assessed its performance. Second, the model considers that the delivery window of the supplier is not an important factor for splitting the demand between players. Future works can include the delivery time or the risk of delivery delay for taking that decision, or even integrate inventory rationing models with last mile models. Additionally, the performance of the model was confirmed for a reduced range of demands and types of SKUs offered by the e-tailers. It is recommended to extend this analysis in order to determine if the proposed model can be generalized for other scenarios. Finally, researchers involved in the area of contractual models are encouraged to develop measures to assess the economic impact as well as revenue-sharing models for the strategic and economic gains derived from advancing the decision-making over the demand splitting, which provide efficiency gains to other actors beside the e-tailers both downstream and upstream the supply chain.

The model presented in chapter 4 has also positive implications for both academics and practitioners, however, it also has some limitations. First, we only confirmed the performance of the RL-based CBM model for a 3-component system so it is recommended to test the proposed model in systems compose of a higher number of components in order to verify if the results can be generalized, and to verify how difficult can be to understand the rationality of the optimal agent in order to extract maintenance management insights. Since the model consider multi-state systems tracking intermediate states of components, it can be also interesting to extend the aforementioned model to multi-yielding systems. In these systems, the intermediate degradation states on the components of the system can lead to a decreasing in production rates or quality of products. Therefore, the maintenance decision can be integrated with other functions such as production planning and quality control.

REFERENCES

ANDERSON, C. **The long tail**: why the future of business is selling less of more. Hachette Books, 2016.

ANDRIOTIS, C. P., & PAPAKONSTANTINOU, K. G. Managing engineering systems with large state and action spaces through deep reinforcement learning. **Reliability Engineering & System Safety**, 191, 106483, 2019.

AYANSO, A; DIABY, M; NAIR, S.K. Inventory rationing via dropshipping in internet retailing: a sensitivity analysis, **European Journal of Operational Research**, 171 (1), 135-152, 2006.

BAHRIA, N ; CHELBI, A ; DRIDI, I. H ; BOUCHRIHA, H. Maintenance and quality control integrated strategy for manufacturing systems. **European journal of industrial engineering**, 12(3), 307-331, 2018.

BARDE, S. R; YACOUT, S; SHIN, H. Optimal preventive maintenance policy based on reinforcement learning of a fleet of military trucks. **Journal of Intelligent Manufacturing**, 30(1), 147-161, 2019.

BARDE, S. R; YACOUT, S; SHIN, H. Optimal preventive maintenance policy based on reinforcement learning of a fleet of military trucks. **Journal of intelligent manufacturing**, 30(1), 147-161, 2019.

BASHEER, I.A; & HAJMEER, M. Artificial neural networks: Fundamentals, computing, design, and application. **Journal of Microbiological Methods**, 43(1), 3-31, 2000.

BEN-DAYA, M; DUFFUAA, S. O; RAOUF, A; KNEZEVIC, J; AIT-KADI, D. **Handbook of maintenance management and engineering** (Vol. 7). London: Springer London, 2009.

BEVILACQUA, M., & BRAGLIA, M. (2000). The analytic hierarchy process applied to maintenance strategy selection. **Reliability Engineering & System Safety**, 70(1), 71-83, 2000.

BRE, F; GIMENEZ, J.M; FACHINOTTI, V.D. Prediction of wind pressure coefficients on building surfaces using artificial neural networks. **Energy and Buildings**, 158, 1429-1441, 2018.

CAGLE, M. N., Yılmaz, K., & Doğru, H. Digitalization of business functions under Industry 4.0. In **Digital Business Strategies in Blockchain Ecosystems** (pp. 105-132). Springer, Cham, 2020.

CHEN, K; CHIU, F.R; LIN, W.H; HUANG, Y.C. An integrated model for online product placement and inventory control problem in a dropshipping optional environment. **Comput. Ind. Eng**., vol. 117, pp. 71–80, mar. 2018.

CHENG T; PANDEY MD; VAN DER WEIDE JA. The probability distribution of maintenance cost of a system affected by the gamma process of degradation: Finite time solution. **Reliability Engineering & System Safety**, 108:65-76, 2012.

CHENG, G. Q; ZHOU, B. H ; & LI, L. Integrated production, quality control and condition-based maintenance for imperfect production systems. **Reliability**

Engineering & System Safety, 175, 251-264, 2018.

CHENG, Y., LI, B., & JIANG, Y. Optimal choices for the e-tailer with inventory rationing, hybrid channel strategies, and service level constraint under multiperiod environments. **Mathematical Problems in Engineering**, 2016, 1–12, 2016.

DE GUSMÃO, A.P.H; SILVA, M.M; POLETO, T; SILVA, L.C & COSTA, A.P.C.S. Cybersecurity risk analysis model using fault tree analysis and fuzzy decision theory. **International Journal of Information Management**, 43, 248-260, 2018.

DEKKER, R. Applications of maintenance optimization models: a review and analysis. **Reliability engineering & system safety**, v. 51, n. 3, p. 229–240, 1 mar. 1996.

DIABAT, A., & DESKOORES, R. A hybrid genetic algorithm based heuristic for an integrated supply chain problem. **Journal of Manufacturing Systems**, 38, 172-180, 2016.

DING, Q; KOUVELIS P; & MILNER, J.M. Dynamic pricing through discounts for optimizing multiple-class demand fulfillment. **Operations Research** 54 (1), 169–183, 2006.

DING, S.-H; KAMARUDDIN, S. Maintenance policy optimization—literature review and directions. **The international journal of advanced manufacturing technology**, v. 76, n. 5–8, p. 1263–1283, 16 fev. 2015.

ENOS, L. **More e-holiday shoppers, but fewer smiles**, 2010. accessed august 26, 2021, retrieved from http://www.ecommercetimes.com/story/6914.html.

FATORACHIAN, H., & KAZEMI, H. Impact of Industry 4.0 on supply chain performance. **Production Planning & Control**, 32(1), 63-81, 2021.

FRANCIOSI, C., VOISIN, A., MIRANDA, S., & IUNG, B. Integration of I4. 0 technologies with maintenance processes: what are the effects on sustainable manufacturing?. **IFAC-PapersOnLine**, 53(3), 1-6, 2020.

GALLAGHER, T; MITCHKE, M. D; ROGERS, M. C. Profiting from spare parts. **The mckinsey quarterly**, v. 2, n. exhibit 2, p. 1–4, 2005.

GHARAEI, A; HOSEINI SHEKARABI, S.A; AND KARIMI, M. Modelling And optimal lot-sizing of the replenishments in constrained, multi-product and bi-objective EPQ models with defective products: Generalised Cross Decomposition. **Int. J. Syst. Sci. Oper. Logist.**, 2019a.

GHARAEI, A; HOSEINI SHEKARABI, S.A; KARIMI, M; POURJAVAD, E; AMJADIAN, A. An integrated stochastic EPQ model under quality and green policies: generalised cross decomposition under the separability approach. **Int. J. Syst. Sci. Oper. Logist**., 2019b.

GHARAEI, A; KARIMI, M; HOSEINI SHEKARABI, S.A. An integrated multi-product, multi-buyer supply chain under penalty, green, and quality control polices and a vendor managed inventory with consignment stock agreement: The outer approximation with equality relaxation and augmented penalty algorithm. **Appl. Math. Model**., vol. 69, pp. 223–254, May 2019c.

HAMDAN, B., & DIABAT, A. A two-stage multi-echelon stochastic blood supply chain

problem. **Computers & Operations Research**, 101, 130-143, 2019.

HAUGEN, S; BARROS, A; VAN GULIJK, C; KONGSVIK, T; & VINNEM, J. E. Safety and Reliability–Safe Societies in a Changing World: **Proceedings of ESREL 2018**, June 17-21, 2018, Trondheim, Norway. CRC Press.

HOMPEL, M. & Y SCHMIDT, T. **Warehouse Management: Automation and Organisation of Warehouse and Order Picking Systems**. Springer Science & Business Media, 2006.

HORENBEEK, A. VAN; BURÉ, J. Joint maintenance and inventory optimization systems: a review. International journal of production economics, v. 143, n. 2, p. 499–508, 1 jun. 2013.

HOSEINI SHEKARABI, S.A; GHARAEI, A; KARIMI, M. Modelling and optimal lot-sizing of integrated multi-level multi-wholesaler supply chains under the shortage and limited warehouse space: generalised outer approximation. **Int. J. Syst. Sci. Oper. Logist.**, vol. 2674, no. February, pp. 1–21, 2018.

HUANG, J; CHANG, Q; ARINEZ, J. Deep reinforcement learning based preventive maintenance policy for serial production lines. **Expert Systems with Applications**, 160, 113701, 2020.

HUANG, W; SONG, G; LI, M; HU, W; XIE, K. Adaptive Weight Optimization for Classification of Imbalanced Data. **International Conference on Intelligent Science and Big Data Engineering (pp. 546-553)**, Springer, Berlin, Heidelberg, 2013.

JARDINE, ANDREW KS, DAMING LIN, AND DRAGAN BANJEVIC. A review on machinery diagnostics and prognostics implementing condition-based maintenance. **Mechanical systems and signal processing** 20.7 (2006): 1483-1510.

JASIULEWICZ-KACZMAREK, M., ANTOSZ, K., WYCZÓŁKOWSKI, R., MAZURKIEWICZ, D., SUN, B., QIAN, C., & REN, Y. Application of MICMAC, Fuzzy AHP, and Fuzzy TOPSIS for Evaluation of the Maintenance Factors Affecting Sustainable Manufacturing. **Energies**, 14(5), 1436, 2021.

JIMENEZ G.H, ORREGO R.T; CAVALCANTE, C.A.V. Order planning policies for business to consumer e-tail stores. **Computers & Industrial Engineering**, 136c,106-116, 2019.

JIMENEZ, H.S; RODRIGUES T.F, DANTAS, M.M; CAVALCANTE, C.A. A dynamic inventory rationing policy for business-to-consumer e-tail stores in a supply disruption context. **Computers & Industrial Engineering**, 142, 106379, 2020.

JONES, B; JENKINSON, I; WANG, J. Methodology of using delay-time analysis for a manufacturing industry. **Reliability Engineering & System Safety**, 94(1), 111-124, 2009.

KAELBLING, L. P; LITTMAN, M. L; & MOORE, A. W. Reinforcement learning: A survey. **Journal of artificial intelligence research**, 4, 237-285. 1996.

KAZEMI, N; ABDUL-RASHID, S.H; GHAZILLA, R.A.R; SHEKARIAN, E; ZANONI, S. Economic order quantity models for items with imperfect quality and emission considerations. **Int. J. Syst. Sci. Oper. Logist**., vol. 5, no. 2, pp. 99–115, 2018.

KEIZER, M. C. O; FLAPPER, S. D. P; TEUNTER, R. H. Condition-based

maintenance policies for systems with multiple dependent components: A review. **European Journal of Operational Research**, 261(2), 405-420, 2017.

KHAN, S & YAIRI, T. A review on the application of deep learning in system health management. **Mechanical Systems and Signal Processing**, 107, 241-265. 2018

KHOUJA, M AND STYLIANOU, A.C. A (q, r) inventory model with a dropshipping option for e-business. **Omega**, vol. 37, no. 4, pp. 896–908, 2009.

Kim M and Lennon S. Consumer response to the online apparel stockouts. **psychology and marketing**, 28(2), pp. 115-144, 2011.

KINGMA, D.P & BA, J. **Adam: A method for stochastic optimization**, Arxiv Preprint Arxiv:1412.6980, 2014.

KUHNLE, A; JAKUBIK, J; LANZA, G. Reinforcement learning for opportunistic maintenance optimization. **Production Engineering**, 13(1), 33-41, 2019.

LECUN, Y; BENGIO, Y; HINTON, G. **Deep Learning. Nature**, 521(7553), p.436, 2015.

LINDSTRÖM, J., LARSSON, H., JONSSON, M., & LEJON, E. Towards intelligent and sustainable production: combining and integrating online predictive maintenance and continuous quality control. **Procedia cirp**, 63, 443-448, 2017.

LIU, Y ; CHEN, Y ; & JIANG, T. Dynamic selective maintenance optimization for multi-state systems over a finite horizon: A deep reinforcement learning approach. **European Journal of Operational Research**, 283(1), 166-181, 2020.

MAHAR, S & WRIGHT, P.D. The value of postponing online fulfillment decisions in multi-channel retail/e-tail organizations. **Computers & Operations research**, 36 (11), 3061-3072, 2009.

MARINI, F; BUCCI, R; MAGRÌ, A.L; MAGRÌ, A. Artificial neural networks in chemometrics: History, examples and perspectives. **Microchemical Journal**, 88(2), 178-185, 2008.

MARSEGUERRA, M; ZIO, E; PODOFILLINI, L. Condition-based maintenance optimization by means of genetic algorithms and Monte Carlo simulation. Reliability **Engineering & System Safety**, 77(2), 151-165, 2002.

MODAK, N. M.; MODAK, N., PANDA, S.; SANA, S. S. Analyzing structure of two-echelon closed-loop supply chain for pricing, quality and recycling management. **Journal of Cleaner Production**, 171, 512-528, 2018.

MODAK, N. M.; PANDA, S.; & SANA, S. S. Pricing policy and coordination for a two-layer supply chain of duopolistic retailers and socially responsible manufacturer. **International Journal of Logistics Research and Applications**, 19 (6), 487-508, 2016a.

MODAK, N. M.; PANDA, S.; & SANA, S. S. Pricing policy and coordination for a distribution channel with manufacturer suggested retail price. **International Journal of Systems Science: Operations & Logistics**, 3 (2), 92-101, 2016b.

MODAK, N. M.; PANDA, S.; SANA, S. S. Three-echelon supply chain coordination considering duopolistic retailers with perfect quality products. **International Journal**

of Production Economics, 182, 564-578, 2016c.

MODAK, N. M.; PANDA, S.; SANA, S. S. Two-echelon supply chain coordination among manufacturer and duopolies retailers with recycling facility. The **International Journal of Advanced Manufacturing Technology**, 87 (5-8), 1531-1546, 2016d.

MODAK, N. M.; PANDA, S.; SANA, S. S.; BASU, M. Corporate social responsibility, coordination and profit distribution in a dual-channel supply chain. **Pacific Science Review**, 16 (4), 235-249, 2014.

MORGAN, C. STRUCTURE. speed and salience: performance measurement in the supply chain. **Business Process Management Journal**, v. 10, n. 5, p. 522-536, 2004. http://dx.doi.org/10.1108/14637150410559207

NG, A. **Machine learning yearning**: technical strategy for ai engineers, in the era of deep learning, draft version 0.5, Harvard Business Publishing, 2016.

NGUYEN, K.T & MEDJAHER, K. A new dynamic predictive maintenance model using deep learning for failure prognostics. **Reliability Engineering & System Safety**, 188, 251-262, 2019.

NICOLAI, R. P.; DEKKER, R. Optimal maintenance of multi-component systems: a review. *in:* **complex system maintenance handbook**. London: Springer London, 2008. p. 263–286.

OKOH, P. Maintenance grouping optimization for the management of risk in offshore riser system. **Process Safety and Environmental Protection**, 98, 33-39, 2015.

OZER, O & WEI, W. Inventory control with limited capacity and advance demand information. **Operations Research**, 52(6), 988-1000, 2004.

PARASCHOS, P. D; KOULINAS, G. K; KOULOURIOTIS, D. E. Reinforcement learning for combined production-maintenance and quality control of a manufacturing system with deterioration failures. **Journal of Manufacturing Systems**, 56, 470-483, 2020.

PARK, C. A partial backordering inventory model with a dropshipping option under purchase dependence, **Korean Manag. Sci. Rev**., vol. 33, no. 1, pp. 1–16, Aug. 2016.

PARK. A Partial Backordering Inventory Model with a Dropshipping Option under Purchase Dependence. **Korean Manag. Sci. Rev**., vol. 33, no. 1, pp. 1–16, Aug. 2016.

PRECHELT, L. Automatic early stopping using cross validation: Quantifying the criteria. **Neural networks**, 11(4), 761-767, 1998.

QUESADA, G., R. RACHAMADUGU, M. GONZALEZ, AND J. L. MARTINEZ. “Linking Order Winning and External Supply Chain Integration Strategies.” **Supply Chain Management: An International Journal** 13 (4): 296–303. doi:10.1108/13598540810882189, 2008.

RABBANI, M; HOSSEINI-MOKHALLESUN, S.A; ORDIBAZAR, A.H; FARROKHI-ASL, H. A hybrid robust possibilistic approach for a sustainable supply chain location-allocation network design. **Int. J. Syst. Sci. Oper. Logist**., 2018.

ROCCHETTA, R ; BELLANI, L ; COMPARE, M ; ZIO, E; PATELLI, E. A reinforcement learning model for optimal operation and maintenance of power grids. **Applied energy**, 241, 291-301, 2019.

ROMERO C. **A Handbook of Critical Issues in Goal Programming**. Pergamon Press, Oxford, 1991.

ROMERO, C. **Handbook of critical issues in goal programming**. Elsevier, 2014.

S.C. RIM, S.C & PARK, I:S. Order picking plan to maximize the order fill rate. **Computers & Industrial Engineering**, 55 (3), 557-566, 2008.

SANTOS, A. C; CAVALCANTE, C. A; & RIBEIRO, L. F. The use of second-hand items based on delay time modelling. **Process Safety and Environmental Protection**, 146, 118-125, 2021.

SVOZIL, D; KVASNICKA, V; POSPICHAL, J. Introduction to multi-layer feed-forward neural networks. **Chemometrics and intelligent laboratory systems** 39.1: 43-62, 1997.

TAN, K. H., G. JI, C. P. LIM, AND M. L. TSENG. “Using Big Data to Make Better Decisions in the Digital Economy.” **International Journal of Production Research** 55 (17): 4998–5000. doi:10.1080/00207543.2017.1331051, 2017.

VAN HORENBEEK, A & PINTELON, L. A dynamic predictive maintenance policy for complex multi-component systems. **Reliability Engineering & System Safety**, 120, 39-50, 2013.

WANG, T & TOKTAY, B.L. Inventory management with advance demand information and flexible delivery. **Management Science**, 54(4), 716-732, 2008.

WANG, X; WANG, H; Qi, C. Multi-agent reinforcement learning based maintenance policy for a resource constrained flow line system. **Journal of Intelligent Manufacturing**, 27 (2), pp. 325-333, 10.1007/s10845-013-0864-5, 2016.

WANG, X; WANG, H; QI, C; SIVAKUMAR, A. I; WANG, X; WANG, H; ... & SIVAKUMAR, A. I. Reinforcement learning based predictive maintenance for a machine with multiple deteriorating yield lev-els. **J Comput Inf Syst** 10(1):9–198, 2014.

ZHANG, N., & SI, W. Deep reinforcement learning for condition-based maintenance planning of multi-component systems under dependent competing risks. **Reliability Engineering & System Safety**, 203, 107094, 2020.

APÊNDICE A – ALGORITHMS

Algorithm 1 DDQN-GP training

for *episode*=1 to *m* **do**
  initialize action dictionary *AE*
  initialize dictionary *Agent memory*
  for inspection $T_k = 1$ to *n* **do**
    read S_{T_k}
    generate *randomNumber*
    if *randomNumber* ≤ *epsilon* **then**
      select random A_{T_k} from *AE*
    Else
      predict $q(S_{T_k}, A_{T_k})$ for *AE* with *PolicyDNN*
      select A_{T_k} with $max_q(S_{T_k}, A_{T_k})$
    end if
    execute A_{T_k} in *MaintenanceEnv* and observe $R_{T_{k+1}}$
    receive $S_{T_{k+1}}$
    if $T_k = n$ **then**
      done ← True
    Else
      Continue
    end if
    store $E(S_{T_k}, A_{T_k}, R_{T_{k+1}}, S_{T_{k+1}}, done)$ in *AgentMemory*
  end for
  sample *minibatch* from *AgentMemory*
  step←*step*+1
  if $step = TargetSteps$ **then**
    $TargetDNNweights \leftarrow PolicyDNNweights$
    $step \leftarrow 0$
  Else
    Continue
  end if
  for $E(S_{T_k}, A_{T_k}, R_{T_{k+1}}, S_{T_{k+1}}, done)$ in *minibatch* **do**
    $target \leftarrow R_{T_{k+1}}$
    $S_{T_{k+1}} \leftarrow S_{T_{k+1}}$
    if not *done* **then**
      predict $q(S_{T_{k+1}}, A_{T_{k+1}})$ using $TargetDNN$
      select $max_q(S_{T_{k+1}}, A_{T_{k+1}})$
      compute $target = R_{T_{k+1}} + \gamma \times max_q(S_{T_{k+1}}, A_{T_{k+1}})$
    else
      compute $targetf =$ predict $q(S_{T_k}, A_{T_k})$ using $PolicyDNN$
      retrain $PolicyDNN$ with $(S_{T_k}, targetf)$
    end if
  end for
  if $epsilon > epsilonMin$ **then**
    Perform $epsilonDecay$
  Else
    $epsilon \leftarrow epsilonMin$
  end if
  reset *MaintenanceEnv*
end for

Source: The Author (2021).