INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Departamento de Engenharia de Electrónica e Telecomunicações e de Computadores

Machine Learning Applied to Asset Management

Pedro Costa Santos - 45366

Dissertação para obtenção do Grau de Mestre
em Engenharia Informática e de Computadores

Orientadores : Doutor Nuno Datia
Doutora Matilde Pós-de-Mina Pato

Júri:

Presidente: Doutor José Manuel de Campos Lages Garcia Simão

Vogais: Doutor Artur Jorge Ferreira
Doutor Nuno Miguel Soares Datia

October, 2022

INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Departamento de Engenharia de Electrónica e Telecomunicações e de Computadores

Machine Learning Applied to Asset Management

Pedro Costa Santos - 45366

Dissertação para obtenção do Grau de Mestre
em Engenharia Informática e de Computadores

Orientadores : Doutor Nuno Datia
Doutora Matilde Pós-de-Mina Pato

Júri:

Presidente: Doutor José Manuel de Campos Lages Garcia Simão

Vogais: Doutor Artur Jorge Ferreira
Doutor Nuno Miguel Soares Datia

October, 2022

Acknowledgments

Gostaria de expressar o meu agradecimento aos meus orientadores, Professor Nuno Datia e Professora Matilde Pato, por me guiarem ao longo desta tese, pela constante disponibilidade e por me ajudarem a tornar um engenheiro informático melhor.

Agradecer também ao Professor José Sobral, pelos contributos no domínio da mecânica.

Por fim, um agradecimento especial à família e amigos, de certa forma sempre fizeram parte deste percurso.

Abstract

Asset management is a process that aims to coordinate the life cycle of a company's assets. It involves keeping track of all of the various physical objects that are under the ownership of a given company, such as equipment, tools, and machines. A work order (WO) is a document (paper or digital) that describes all steps to perform the maintenance operation. When creating a work order using "free text", the sentences are left to the technician's discretion, which can lead to non-existent or grammatically incorrect words, or even make it difficult to identify tendencies and perform an appropriate analysis. The purpose is achieving a normalisation of the words that are used when the same asset maintenance situation is considered. The normalisation is reached by suggesting words while a technician is filling a work order. The word suggestion algorithm works based on the past of similar maintenance occurrences. The corpus comes from asset management records (work orders) collected in a health facility in Portugal, which is operated by a private company. Each maintenance record is used to characterise and keep history of a given asset management occurrence.

First, the application of Natural Language Processing (NLP) techniques to process the work order's description allow to understand how the words are used in this technical domain. Then, a recommender system based on a Bayesian Network is implemented. This recommender system has the capability of adapting itself to the present needs verified by the technicians through the inclusion of an implicit user feedback mechanism. The implementation of this recommender system aims at the objective of achieving the normalisation of the terms used by the technicians when filling a work order.

Keywords: Recommendation System, NLP, Word Embeddings, Asset Management, Bayesian Network

Resumo

A gestão de ativos consiste num processo que pretende coordenar o ciclo de vida dos ativos de uma dada empresa. Implica manter um registo dos vários objetos físicos que estão sob a propriedade da empresa, tais como equipamento, ferramentas e máquinas. Uma ordem de trabalho (OT) consiste num documento (em papel ou digital) que descreve todos os passos que permitem completar uma operação de manutenção num dado ativo. Ao serem criadas ordens de trabalho de um modo livre, isto é, sem qualquer restrição no que é escrito, o critério na sua elaboração é deixado ao cargo do técnico, o que pode levar a palavras não existentes ou gramaticalmente incorretas, ou até mesmo dificultar a identificação de tendências por parte de algoritmos de análise. O objetivo é atingir a normalização das palavras que são utilizadas quando se considera a mesma situação da gestão de ativos. A normalização é atingida através da sugestão de palavras enquanto um técnico está a preencher uma ordem de trabalho. O algoritmo que proporciona as sugestões funciona com base no passado, ao analisar situações passadas da gestão de ativos. O corpus provém de ordens de trabalhos recolhidas numa unidade de saúde em Portugal, que é operada por uma empresa privada. Em primeiro lugar, a aplicação de técnicas de Processamento Natural de Linguagem sobre as descrições das ordens de trabalho permite entender como é que as palavras são usadas neste domínio técnico. De seguida, é implementado um sistema de recomendação baseado numa Rede Bayesiana. Este sistema de recomendação tem a capacidade de se adaptar às necessidades verificadas no presente pelos técnicos, através da inclusão de um mecanismo implícito de *feedback*. A implementação deste sistema de recomendação pretende atingir o objetivo de alcançar a normalização dos termos que são utilizados por um técnico aquando a criação de uma ordem de trabalho.

Palavras-chave: Sistema de Recomendação, Processamento Natural de Linguagem, *Word Embeddings*, Gestão de Ativos, Rede Bayesiana

Contents

List of Figures

List of Tables

Acronyms

BERT Bidirectional Encoder Representations from Transformers. 33

BIM Building Information Modeling. 8, 9

BN Bayesian Network. xii, xiii, 11, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 49, 50, 51, 52, 53, 58, 60, 67, 68, 69

CB Content-based. 16, 17

CBoW Continuous Bag of Words. 24

CF Collaborative Filtering. 16

CMMS Computerised Maintenance Management System. 2

CV Cross-validation. xv, 55, 56

EHR Electronic Health Record. 15

FCI Facility Condition Index. 2, 3

FM Facility Management. 9

GD Gradient Descent. 17

GloVe Global Vectors for Word Representation. 14, 15

GTP-3 Generative Pre-Trained Transformer. 33

HMM Hidden Markov Models. 34

LDA Latent Dirichlet Allocation. 17, 22

LSA Latent Semantic Analysis. 17

LSTM Long Short-Term Memory. 34

MAP Mean Average Precision. 52, 53
ML Machine Learning. 13, 17, 32, 33, 34, 49, 52, 67, 69
MRR Mean Reciprocal Rank. 51, 52, 53, 54, 55, 56, 57, 58, 59

NB Naive-Bayes. 17
NLP Natural Language Processing. xi, xii, xiii, 7, 8, 13, 14, 15, 19, 20, 21, 23, 24, 34, 55
NN Neural Network. 24, 34

PCA Principal Component Analysis. 15, 25
POS Part-of-speech. 21

ROBERTA Robustly Optimized BERT Pretraining Approach. 33

TF-IDF Term Frequency-Inverse Document Frequency. xii, 22
TLP Technical Language Processing. 14
t-SNE t-distributed Stochastic Neighbour Embedding. 25, 26

WO Work Order. xiii, xv, 2, 5, 6, 7, 8, 9, 10, 14, 19, 20, 21, 22, 23, 25, 28, 30, 31, 32, 34, 35, 36, 38, 41, 43, 44, 45, 49, 50, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 67, 68, 69

1

Introduction

From a broad view, the area of civil construction encompasses the processes of building planning (design), construction, and facilities management [11]. Regarding the process of facilities management, it includes several professions to ensure functionality, comfort, safety and efficiency of the built environment, by integrating people, place, process, and technology [21]. One of the branches of facilities management is asset management, which deals with operation management and maintenance of assets, having in mind the maximisation of the working time of each asset, the minimisation of downtime costs, the prediction of when assets might malfunction, and other strategic optimisations and decisions that asset managers have to take into account, presented further in this document.

1.1 What is Asset Management?

The *Asset Management Primer* written by the *United States Department of Transportation* [2] offers the following working definition:

> ***Asset management*** *is a systematic process of maintaining, upgrading, and operating physical assets cost-effectively. It combines engineering principles with sound business practices and economic theory, and it provides tools to facilitate a more organised, logical approach to decision-making. Thus, asset management provides a framework for handling both short and long-range planning.*

As can be seen from this definition, asset management is a complex process involving coordination of several disciplines in order to ensure that physical assets are maintained, upgraded, and operated correctly. In an attempt to deal with such complexity, the article Asset management A to Z, written by D.J. Vanier [65], describes the asset management as the successful implementation of data collection related to six questions:

- What do you own?
- What is it worth?
- What is the deferred maintenance?
- What is its condition?
- What is the remaining service life?
- What do you fix first?

The idea is that through the sequential answering of these questions, an asset management plan can be implemented. For the sake of better understanding each question's intricacies, a brief overview, based in the article's explanations, is given.

What do you own? It is the first question and has the goal of getting to know the extent of the asset management portfolio. Typically, a tool named Computerised Maintenance Management System (CMMS) [8] is used in order to registry what assets are owned, among other functionalities such as resource and labour management and Work Order (WO) management.

What is it worth? Has the goal of knowing the asset's value. This value can have distinct forms, such as historical value, appreciated historical value, current replacement value, "performance in use" value, deprival cost, and market value.

What is the deferred maintenance? Refers to the maintenance tasks that were postponed. Postponing maintenance tasks has a compounding effect in which if the programmed maintenance for one year is not completed, then the costs of maintenance, repair or replacement are significantly higher in subsequent years. So that comparisons of the amount of deferred maintenance between different facilities or systems can be made, the term Facility Condition Index (FCI) is used. It provides a general impression of the overall state of the individual facilities. A higher FCI indicates a worsening relative state of the facility.

What is its condition? It is a question that can help to formalise an objective decision, due to the fact that the extent and value of an asset have been determined, and an appropriate FCI has been attributed.

What is the remaining service life? After determining the portfolio's extent and each asset's value and condition, it is important to consider one last factor, which is the remaining service life. By studying the asset's service life, it is possible to calculate the life-cycle costs and understand maintenance, repair and replacement strategies.

What do you fix first? The final question, not answered directly, but by combining the answers to the previous five questions and understanding that decisions have to be made *today* as to what to fix *tomorrow*. Within the boundaries of this final question/decision, there is the dichotomy of having both financial and technical challenges, that is, maintenance, repair and replacement of assets have a cost that will be converted into technical and functional benefits - these must be carefully weighted in this cost *versus* benefit trade-off. Adding to this final answer's complexity, is the challenge of planning the horizon of long-term decision-making regarding asset management, which can be broke down into three stages: the operational planning horizon is within the two-year frame, tactical planning horizon in the two to five years frame and strategic planning horizon in the beyond-five years frame. This challenge enlightens that decisions that might be good in a two-year frame, might not be that good in a strategic planning mindset (beyond five years).

This understanding of what asset management is, enables the possibility to dive into what influences this process.

1.2 What Influences Asset Management?

There are a number of factors that influence the way in which infrastructure assets are managed, so that they can continue to provide value to its owners and the community. According to the article "Challenges in infrastructure asset management" [6], the financial climate, the ageing of the infrastructure, the network effect and the "*silo mentality*" [54] are some of the factors, among others, that affect asset management. These factors are important to better understand the problems inherent to asset management, and are presented hereafter.

The financial climate factor states that in today's economic scenario, infrastructure owners and operators are continually under pressure to maintain an adequate level of service and performance, within a budget that tends to shrink. Success in these conditions is determined by the operator's ability to find the right balance of expenditure, without taking additional risks and negatively impacting the asset's performance over the life of the infrastructure.

The ageing of the infrastructure factor can be explained by the fact that assets are characterised by a long lifetime and present a complex deterioration, which implies that it is important to gather knowledge about the way these assets deteriorate over time and how this deterioration can affect costs, risks and performance. It is necessary to understand the condition and performance of the assets in an ageing infrastructure, in order to predict how these might evolve in the future.

The network effect states that an individual asset does not provide value on its own, and so it is from the combination of different types of assets that value is generated. On the other hand, a single asset can have the possibility of affecting the value generated by a network, if it is a critical one. With these considerations in mind, and knowing that the nature of assets can differ (e.g civil, electrical, mechanical), in order to achieve an effective management of an infrastructure network it is important to adopt multidisciplinary and systems-based approaches.

A silo mentality is "*the unwillingness to share information or knowledge between employees or across different departments within a company*" [54]. This adds complexity to the process of managing the assets due to the fact that the network is composed of assets that belong to different departments, and therefore it is very important that there is effective communication and sharing of information between them, in order to achieve effective asset management. These *silos* can be overcome by the introduction of artificial intelligence mechanisms that allow to share knowledge through automatic learning [55].

From these factors (and others not mentioned, equally important, present in the referenced article [6]) it is possible to conclude that asset management is a complex process, influenced by several factors. In order to deal with this magnitude of complexity, it is necessary to adopt appropriate solutions that promote the effective use of data, so that there can be a better understanding of risk and criticality regarding the asset's functioning. This knowledge enables informed decisions that improve the asset's lifecycle and its cost, and allows the application of predictive analytic techniques. These

cutting-edge techniques provide the ability to anticipate component failure through the analysis of the gathered data, with a given probability, so that action can be taken in advance, costs can be reduced, and performance can potentially be increased. All of these considerations are (or should be) taken into account in solutions known as WO management systems.

1.3 Into the World of Work Order Management Systems

In a WO management system, there is the necessity to combine information that is produced both by sensors and other intelligent devices, with information produced by technicians.

Data from non-human sources has errors associated with typical system functioning, which are quantifiable and can be reduced, while data that originates from humans usually has significant quality issues. They can be as diverse as categorisation errors, omission of information, non-registration of occurrences, wrongfully inputting date and time related values, or inputting values with the wrong order of magnitude. Quality issues such as the examples given, lead to the need of performing a careful and qualified human analysis on the provided data, in a way that enables to transform the provided information in reliable data, that can subsequently be studied and used to help making decisions.

Asset management is done through a methodology that has as a first step the creation of a request, which is used to report an occurrence that is detected either by a sensory device, a technician or non technical personnel; this occurrence can be something as simple as a loose or broken door knob, a regular worker noticing it is too hot in a room where the air conditioning is on, or something more complicated like a *chiller* [1] [7] that stopped working. These requests are used to point out that something is wrong in a given asset, and needs appropriate attention. Depending on the problem, a request can give rise to a WO, if its solution is not trivial, like the loose door knob example; on the other hand, the other example, which consisted in the malfunctioning of a chiller, will most likely lead to the production of a WO, so that a skilled technician can examine the asset and figure out what is wrong.

These WOs are technical reports that specify the asset that needs intervention and the details about the work to be done or, in the case that the work is unknown from the start, the characteristics of the malfunctioning. Every WO has a reserved field where

[1] A chiller is a refrigeration system used to lower the temperature of machinery or industrial spaces. It does so by removing heat from the system and transferring it elsewhere.

Table 1.1: Example of a Work Order with 11 considered fields out of 75.

Field	Value
Work Order ID	16715
Job number for the given WO - WOs can be completed through several jobs	1
Asset complete identifier	06001MCTEEE
Asset ID	6001
Work Order description	Periodic review of leaks
Date when the WO began	2014-06-04T09:00:00
Date when the WO ended	2014-06-04T11:00:00
Description of the work performed	Periodic review of leaks
Asset designation	Chiller 1
Asset Family	CT
Asset Subfamily	CH

the associated technician describes the job that was accomplished, along with other important details that characterise it.

Both requests and WOs are the heart and lungs of asset maintenance, since every occurrence, from the more urgent to the least, is reported through them. Essentially, the creation and management of requests and WOs allows to achieve a regulation of asset's life cycle. In the specific case of this research, the facility where the process of asset management occurs is a healthcare facility in Portugal. Each maintenance record in this healthcare facility is used as an input to support the developed research; they characterise and keep history of a given situation in the context of the healthcare facility (e. g. asset malfunction, routine inspection, component substitution, etc.) and can therefore be used to analyse tendencies and relationships between the words that are used to describe the situation/problem. These maintenance records consist of WOs which are composed of several fields, and in which the technician fills the information in order to best describe the given situation. The collection of data used consists in a total of $38,445$ WOs, all concerning the healthcare facility and its assets.

Of all the seventy five fields that compose a WO, the ones with the most significant information are presented hereafter, with an example for each field, taken from a real WO as shown in Table 1.1. It is important to stand out that all of these fields were used when performing an exploratory analysis of the data, however the application of the

Natural Language Processing techniques that are described in Chapter three only considered the fields "description of the WO" and "description of the work performed". In reality, only one of these fields was chosen, being the one with the most characters, which indicated more detailed information about the WO. Other important fields such as date when the WO began and ended and the subfamily to which it belongs to were also used, but in post-processing phases after the Natural Language Processing (NLP) techniques. A subfamily is a domain specific 2nd level classification, where assets with similar characteristics (e.g. similar components) are grouped together to form a subfamily. A first level classification consists in a family, through which a first segregation of the assets occurs.

1.3.1 Domain known problems that influence the asset management process

Data that sources from human minds can have problems such as quality issues or inconsistencies that can be troubling. When creating a WO, the sentences that are created are considered "free text", left to the criteria of the technician that writes it. This can lead to spelling errors, words that don't exist, or a lack of consistency in the words that are used, that is, the usage of different words to describe the same job in WOs that are alike, instead of always using the same words and sentence constructions. These are some of the problems normally encountered when dealing with data that regards natural language, so there must be strategies to overcome them, specially when the information is used to feed algorithms that run a decision-making process.

The first solution that might come to mind is the usage of a rigid taxonomy, so that the words inputted by the technicians are no longer unchecked. This solution fails due to the fact that technicians usually do not like having a computer tell them what they can or cannot write, and the creation of a WO would be more time-consuming. On the other hand, as stated before, having the sentences of each WO as "free text" makes it harder for intelligent algorithms to analyse it. That said, it is important that there is a normalisation of the terms and words used by technicians, so that subjective sentence constructions can be excluded and there may be a full coverage of the totality of the actions that are performed in the assets; this normalisation can be achieved through the suggestion of words while a technician is filling a WO's description.

In this thesis, the goal is to suggest words to a technician while he/she fills out a new WO, involving the asset management procedure carried out in a Portuguese hospital.

The research that supports the development of this suggestion algorithm uses the previously described WOs, that were created in the context of the hospital's facility management. The idea is similar to what occurs in modern search engines, that is, while the user (in this case, the technician) inputs words relevant to characterise a given WO, words that are identified as related to the previously inputted ones are suggested. This method of suggestion enables to optimise and regulate the process of writing a WO.

An algorithm that supports the prediction of the next word when creating a description does so by processing information from previously created WOs and by recurring to NLP [42]. This means that after its processing using NLP techniques, this information can enrich the asset management process, that is, further knowledge about the WOs and how they are created can be derived through the analysis of how the technicians use the terms to describe asset management situations. Through the normalisation of the terms that are used to describe a WO, it is intended to avoid ambiguities when describing a given asset management situation, avoiding also the use of several different words and expressions to describe the same malfunction/asset management situation.

1.3.2 How can the information present in the work orders be used?

Sensory information, and humanly produced information present in the WOs, can have several applications. This kind of information can be used to feed the Building Information Modeling (BIM), which can be defined as the use of a virtual building information model to develop building design solutions, to design documentation, and to analyse construction processes [5]. This definition can be extended to include the operational phases of built assets (such as maintenance and decommissioning). That said, the BIM consists on a virtual construction that presents several details about the building, and in which it is possible to observe interesting aspects such as the composition of the materials used in elements like doors, windows etc., the layout of each floor, the existing assets, among others. Models created in BIM software are detailed 3D representations of built assets, but the information present in those models is not merely visual. A model is generated from a relational database containing information regarding the elements of a structure and their relationships; built asset elements can contain data such as manufacturer's name, model number, composition, cost, manufacturer, relationship to other elements, and related properties such as dimensions, weight, fire resistance, or combustibility (among others). From these 3D simulations, it is possible to create 4D simulations, which add the element of time to a built asset model and automatically prepare construction schedules together, to show a 3D simulation of the construction progress over time. The visualisation of the construction

process enhances the understanding of the whole process, so that nontechnical people (such as clients) can identify any problems or inconsistencies. The visualisation may point out any constraints that had not been previously identified.

By adding a time element to a built asset model, it is possible to understand the asset's life cycle across time. This can be done by visualising all the WOs that are associated to a given asset, and perform metadata analysis in order to find patterns or recurring events (such as the failure of a given component every three months, for example) that might enable to predict asset failure.

Building Information Modeling can also enhance the maintenance phase of built assets [39]; all the specifications for a built asset, down to an individual component level, are recorded in a repository of detailed information, that can be used after the completion of the built asset for Facility Management (FM). This way, FM technicians can have easy and quick access to important information during the maintenance phase and can also update this information over time, which can improve the management of the asset. This functionality also provides the possibility that the owner of the built asset can easily change from one facility manager to another; there is only the need to exchange a single BIM file.

Sensory information and humanly produced information present in the WOs can also be used to aggregate and analyse data regarding the assets and its associated WOs, in order to predict component malfunctioning; this is achieved, for example, by analysing the WO's target component in a given asset and search for how many times this component has malfunctioned in a previous time window (e.g six months). Conclusions taken from the analysis of one asset can help predict the malfunctioning of another asset, that also has the same component.

Visualisation tools are of great use and can help discovering patterns and tendencies present in this sensory and human-made information, which can then help the decision-making process.

1.4 Asset Management and the Industry 4.0 paradigm

This thesis and the associated work presents itself as a good example of the challenges that Industry 4.0 brings and what this paradigm aims at achieving. Industry 4.0 [30] sits on the basis of an advanced digitalisation within factories, the combination of Internet technologies and future-oriented technologies in the field of "intelligent" objects; this technological revolution results in a new fundamental paradigm shift in industrial production, in which there is an increase in mechanisation and automation, that is,

more technical aids will be used to enhance the work process, and automatic solutions will adopt the execution of versatile operations. This paradigm is also characterised by a merging of physical and digital levels, where the digital level virtualizes all the characteristics inherent to the physical level, so there is no distinction between them; what changes in the physical world also changes in the digital world, and both live in symbiosis.

As a result of the analysis of the WOs, it is possible to understand how assets life-cycle might evolve in the future and predict asset component failure through predictive analytic techniques. The described work is the application of Industry 4.0 to the asset management challenge, with focus on next word prediction while a technician is creating a WO's description.

1.5 Project overview and main goals

Once understood that asset management is a complex process, all the developments that can help reduce that complexity are welcome. That said, the objective of this research is to normalise the words used by technicians when creating a WO, so that the same asset management situation is described using the same (or very similar) words. This is achieved through an algorithm that suggests words while a technician is writing a new description. The suggested words are dependent on already written words, and the algorithm adapts the suggestion every time a new word is selected. This objective is measurable through the creation of a prototype in which a technician can create a WO, and the field where the description is inputted is endowed with the capability of sentence auto-complete, similar to what happens in modern search engines. The research process that leads to the final objective is guided through the *CRISP-DM* model [13], which serves as base for achieving the desired data science process. It consists in six phases, namely, business understanding, data understanding, data preparation, modelling, evaluation, and deployment.

1.6 Contributions

As a result of the developed work, two articles were published and two articles were being written at the time of this thesis delivery. One article publication documents the exploratory phase conducted in this thesis which, using Natural Language Processing techniques, allowed to better understand the data present in the domain. It was presented in the 26th International Conference Information Visualisation, entitled

"Comparing Word Embeddings through Visualisation" [52]. The second published article documents the implementation of the BN with implicit user feedback and its details. It was presented in the 13th Symposium of Informatics — INForum 2022 —, entitled "Suggesting Words using a Bayesian Network" [53]. Regarding the articles in the process of writing, one complements the information present in the [52] article, to be a contribution for an edited book. The other consists in a commit to a Special Issue of a peer-reviewed journal of *MDPI*.

1.7 Document Structure

This document is divided into six chapters. The first chapter consists of an introduction to what is asset management, followed by an overview of the associated research and the goals to achieve. The second chapter consists in the state of the art and related work. On the third chapter, the work developed regarding an initial exploratory phase is documented, in order to understand the domain, how words were used in the corpus, and how to accomplish the goal of this work. In the fourth chapter, the developed solution is explained. The fifth chapter consists in the evaluation of the solution and, at last, the conclusion and future work in Chapter 6.

2

State of the Art

This chapter pretends to illustrate the state of the art regarding the asset management domain and the branch of Machine Learning (ML) that includes sentence completion and next word suggestion algorithms. It is divided into two sections. The first one regards the use of Natural Language Processing techniques to improve asset management, and the second one focuses in sentence completion and word suggestion algorithms.

2.1 Usage of Natural Language Processing to improve Asset Management

There are several studies that use Natural Language Processing (NLP) techniques to improve asset management.

The article by Yassine Bouabdallaoui et. al. [44] describes the development of a ML algorithm based on natural language processing for the classification of maintenance requests. The problem described in the referenced article has several similarities to the one portrayed in our research due to the usage of word embeddings. The authors used "*FastText*"'s [22] *pre-trained word embeddings* [47] which allowed to perform transfer learning [63]; transfer learning refines the word embedding model for a specific case, fine-tuning the words that exist in that model, in order to better adapt to the problem's domain. There were created four different types of ML models, one with convolutional

neural networks [12], another with convolutional neural networks with filters, another with long short-term memory neural networks [38] and another with convolutional and long short-term memory neural networks. The used metric was accuracy due to the fact that the dataset was not imbalanced and the model that obtained the best results was the convolutional neural network with filters.

The article written by Christer Stenström, Mustafa Aljumaili, and Aditya Parida [43] aims to demonstrate the benefits of applying NLP techniques to WO descriptions. The presented case study was the Swedish railway infrastructure and it addresses the problems related with the lack of quality of the data present in the WOs. There were taken into account around 11.000 WOs, with 69.382 words in total; after the application of a NLP pipeline, the number of distinct words was down to 8 442. From this subset of words it was possible to perform analysis that allowed to help the decision-making process. The article's conclusion reinforces that the usage of NLP techniques improves the identification of failures, as so the quality of data in general.

Michael P. Brundage et. al. [60] pretends to point out the inefficiencies of the current NLP landscape when it comes to dealing with industrial use cases, that is, when "out of the box" NLP tools are used in more technical domains. Current tools and implementations are not sufficient to deal with the complexity of more technical domains, so the authors propose the Technical Language Processing (TLP) paradigm, emphasising that the industrial asset management community should redouble the efforts when it comes to the production of domain-specific resources. TLP is presented as a methodology that pretends to refine the NLP paradigm through the obtaining of a consensus of the linguistic resources that represent the technical domains (e.g. dictionaries, datasets) across all industry.

One of the most commonly used concept when regarding projects that use NLP is *Word Embeddings*. Computers operate mostly using numeric representations, so they do not understand textual representations the way humans do. A way to overcome this problem is to convert each word into a numeric vector that tries to capture the meaning of the word through a combination of numbers, taking into account its context. Thus, for words with similar meaning we have a similar numerical representation. Furthermore, the concept of word embeddings is not new in the landscape of natural language processing. In recent years, several word embedding models have been developed, such as *Word2Vec* [67], *Fasttext* [22], *Global Vectors for Word Representation (GloVe)* [24], among others [36]. The difficulty lies in understanding which word embeddings model gives a better fit for the given problem at hand, since each model has its own intricacies and each problem has its own complexity. Plus, to better decide which word embedding model to use in each situation, it is important that the performance and results

of these models are compared in different scenarios. Schnabel et al. [20] presents a comprehensive study of evaluation methods for unsupervised embedding techniques, pointing out that there is not always a single, more efficient, go-to solution. As a result of different evaluations, the embedding methods are ordered in different ways. Wang et al. [10] empirically evaluates word embeddings trained from four different corpora, namely clinical notes, biomedical publications, *Wikipedia*, and news. For the case of clinical notes and biomedical publications, word embeddings were trained using unstructured Electronic Health Record (EHR) data available at *Mayo Clinic*, and articles (*MedLit*) from *PubMed Central* [49], respectively. For *Wikipedia* and the news, pre-trained word embeddings *GloVe* [24] and *Google News* were used. The conclusions drawn were that the word embeddings trained in EHR and *MedLit* have the ability to better capture the semantics of medical terms and to find semantically relevant medical terms closer to human judgements than those trained in *GloVe* and *Google News*. It was also concluded that there is no consistent global ranking of word embeddings for all downstream biomedical NLP applications. As a last conclusion, word embeddings trained in the biomedical domain corpora do not necessarily have better performance than those trained in the general domain corpora, for any given downstream biomedical NLP task. Bhoir et al. [9] presents a comparative analysis of different word embedding models: *Continuous bag of words* [67], *Skip gram* [67], *GloVe* [24] and *Hellinger — Principal Component Analysis (PCA)* [27]. The models are compared, having into account different parameters, which include performance with respect to the size of training data, basic overview, the relation of context and target words, memory consumption, the supported classifier used, and effect of changes in dimensions. After an extensive comparison of each of the models taking into account the mentioned metrics and after weighting all the pros and cons, authors conclude that *GloVe* [24] is the best model compared to other models. The decision is supported on how the model scales to large corpus, but it also works well with small corpus; it improves the quality of learned representations by normalising counts and log smoothing them.

Hartmann et al. [48] concerns to the evaluation of Portuguese word embeddings; the authors point out the considerable variety of word embeddings models that can be used nowadays, so it is important that methods to evaluate them becomes a topic of interest. In the context of the research documented in the referenced article, 31 models are created, publicly available, and each of them is evaluated intrinsically and extrinsically, indicating the lack of correlation between performance in syntactic and semantic analogies and syntactic and semantic NLP tasks. The research developed in the referenced article has some similarities with the work developed, specifically regarding the comparison between the *Word2Vec* [67] embeddings and the *Fasttext* [22] embeddings.

The knowledge of how well does a word embeddings model behave in a certain domain is important due to the fact that this technique can be used to explore the corpus and the relationships that exist between the words. One of the research paths taken in this thesis was to understand what word embeddings model to use, from all of the available ones, in order to better understand how words are used in this technical domain of asset management. Therefore, the present thesis conducts a word embeddings comparison in order to better understand what is the most adequate library to use in this asset management domain in specific.

2.2 Sentence Completion and Word Suggestion Algorithms

A tag is a way to associate a relevant keyword or phrase with an entity (such as an image, a document, or a video). Tags are simply one or more pieces of information that describe the content of a document, a web page, or another digital file. The tags provide details about an item and make it easy to locate related items. Tag recommendation refers to the automated process of suggesting useful and informative tags to an emerging object, since tag text manually is usually time-consuming and labour-intensive [56]. It focuses on describing, summarising, and organising the content of the objects while also identifying the user's interests. In other words, tag recommendation allows users to analyse and annotate the web pages, musics and articles with keywords, as well as it can also help users to find multimedia content. For instance, on social networking sites, Last.FM[1] and Flickr[2] use tags to classify music and images, respectively. CiteULike[3] tags articles and photos.

Tag recommendation methods can be divided into: (1) personalised collaborative filtering, and (2) object-centred content-based [4].

Collaborative Filtering (CF) uses historical user behaviour to recommend subjective and personalised tags, often ignoring the content; otherwise, the Content-based (CB) tag recommendation is done by keywords extraction, topic modelling, and text categorisation. On the one hand, in CF the sparsity of users' data and cold-start are the most common problem. On the other hand, overspecialisation, limited content analysis, lack of serendipity and diversity, and new users problems are some of the limitations of the CB. The present thesis research focuses on CB tag recommendation by text

[1] http://www.lastfm.com
[2] http://www.flickr.com
[3] http://www.citeulike.org

classification.

The use of tagging is widely found in Recommender Systems. There have been several methods of tagging proposed from different perspectives [4]. Regarding to CB tag recommendation, Krestel et. al [33] introduced an approach based on Latent Dirichlet Allocation (LDA) for recommending tags of resources in order to improve search. Annotated resources are used to uncover latent topics for which only sparsely tagged new resources are mapped.

To suggest hashtags for microblogs, Ding et al. [17] proposed a topical translation model, which combines the advantages of both topic model and translation model from content to tags. Godin et al. [25] developed an approach to recommending hashtags for tweets. The binary classifier is based on Naive-Bayes (NB) technique, which discriminated between English and non-English language.

Goulart et al. [26] applied NB and Latent Semantic Analysis (LSA) models to develop a hybrid model to complete sentences. Considering each sub-model separately, this hybrid model infers the next word based on a set of words. NB stored co-occurrences patterns of words, and LSA the language semantics. The optimisation is conducted with the Gradient Descent (GD) technique, where these values are adjusted based on an error function. Research documented in [26] presented a hybrid model for word prediction using NB and Latent Information. The model is developed using a methodology that has three stages: training, optimisation, and inference. The first stage consists of training sub-models created through the application of NB and LSA; the idea is to keep two sub-models, each generated by one of the algorithms, allowing to represent two types of information. The former, NB, is used to store co-occurrences patterns of words; the later, LSA, is used to model the language semantics. Then, model optimisation is conducted in order to find the best parameters that allow to combine both sub-models together, using the GD technique, where these values are adjusted based on an error function. These parameters affect the way in which both sub-models contribute to the final outcome. After that, inference is performed and a set of suggestions are given.

Recently, Lei et al. [35] proposed a text classification system that is capable of automatically tag a piece of text. It is called the capsule network, which is a dynamic routing system that can be used for the recommendation task. The system learns from the data collected by the capsule network and automatically updates its recommendations to novel perspectives.

Several studies have shown that ML can improve the asset management process, but they did not find a way to implement a specialised word suggestion algorithm based

on the data collected from these activities.

3

Exploratory Analysis of the Data

This chapter pretends to illustrate the exploratory analysis of the data present in the Work Order (WO), having in mind understand the relationships between words and how they are used. Following the CRISP-DM [13] methodology, this chapter pretends to document the exploratory analysis that was conducted regarding the phase of data understanding. Due to the nature of the assets and their maintenance, the asset management domain is a highly technical and specific one. It is possible to understand that the words that appear in this context and the way they are used are very different from day-to-day speech. In order to understand what is the most adequate solution to this problem of suggesting words to a technician, the first phase in this research consisted of studying Natural Language Processing (NLP) techniques. The usage of these techniques allow to obtain knowledge about the WO, enabling to further explore the possible ways to reach the objective.

3.1 *Stanza* Natural Language Processing Pipeline

The first step, that consists in analysing the words and sentence construction present in the WO descriptions, is done through NLP techniques [42]. The referenced article, written by Chowdhury defines NLP as "*an area of research and application that explores how computers can be used to understand and manipulate natural language text or speech to do useful things*". NLP consists is an area of investigation that aims at creating structured knowledge about natural language and speech, so that every aspect of a sentence, such

as syntax or the constituting words can be interpreted, and patterns and tendencies can be observed. That said, the goal of this NLP processing is to analyse the lexicon used and the sentence constructions, remove words that are too common in day-to-day speech and don't present any significant information, and output simplified descriptions that can easily be interpreted by subsequent algorithms. It is possible to observe the NLP pipeline that was created and the respective techniques in Figure 3.1. The *JSON* file that contains the WO is created through the querying of the health facility. That said, the goal of this NLP processing is to analyse the lexicon used and the sentence constructions, remove words that are too common in day-to-day speech and don't present any significant information, and output simplified descriptions that can easily be interpreted by subsequent algorithms. It is possible to observe the NLP pipeline that was created and the respective techniques in Figure 3.1. The *JSON* file that contains the WO is created through the querying of the database that exists in the health facility; the WO are gathered into this file, and appropriately inputted into the pipeline.

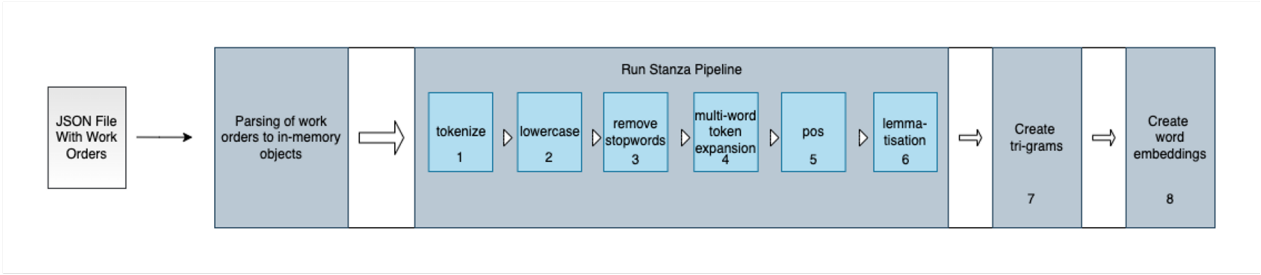


Figure 3.1: Natural Language Processing (NLP) Pipeline

Figure 3.1 presents the architecture of the pipeline. In a first phase the only components were the ones identified as part of the *Stanza* pipeline, until step six. These components allowed to process the WOs and gather important information. Steps seven and eight were added in a posterior phase, and will therefore be described in posterior sections. Stanza [58] consists in a *Python* framework that allows to process a collection of documents through the application of NLP techniques. In this particular case, a document consists in the WO's description. It has a predefined pipeline with implemented NLP operations, although it can also be customised, so there is the possibility to add new operations/modules (known as processors).

Tokenization The first applied technique is *tokenization* [61], which is a way to separate a piece of text into smaller units called tokens. In this particular case, *tokenization* is achieved by splitting words by spaces and keeping the resulting tokens. Multi word tokens such as life-cycle were dealt separately, in the *Multi-word token expansion* phase.

Lowercase Module designed to convert every description to lowercase letters.

Stopword Removal After applying lowercase, it is performed the removal of words that do not have relevance in the problem domain (commonly known as stopwords [59]). The main motive behind the elimination of stopwords is to increase the execution speed and the accuracy of the subsequent algorithms, since only relevant words are considered. Here, we have a domain expert indicating which words should be discarded.

Multi-word token expansion The multi-word token expansion module is used to treat compound tokens that are not processed during the tokenization module. These are tokens such as life-cycle; in this case, life-cycle is considered as a token composed by words life and cycle.

Part-of-speech (POS) Tagging Part-of-speech tagging [46] enables to produce important indicators that will enable to understand the relationships between words, what each word represents in each sentence, and how each word influences the overall meaning of a sentence. It works by categorising words in a text (corpus) in correspondence with a particular part of speech, depending on the definition of the word and its context.

Lemmatisation From the knowledge gathered by POS Tagging, it is then performed lemmatisation, a process of grouping together the inflected forms of a word, identified as the lemma. For example, the lemma of the word chillers is chiller, because the process of lemmatisation converts the words from plural to singular form. Another good example are verbs, in which conjugated verbs are converted to its infinitive form (e.g grouping is converted to group, fixed to fix). This process is commonly used to minimise the number of possible words that the subsequent algorithms deal with. This reduces significantly the complexity and variance present in the data.

From the output of this NLP pipeline (see Figure 3.1), in step six, a simplified version of the descriptions is obtained, that is, it is created a new file with all of the processed WOs descriptions. From these, it is possible to dive deeper into this research and use distinct algorithms to best understand how the words are used by the technicians. That said, several strategies were explored.

3.2 TF-IDF and Topic Modelling

The first strategy was the creation of a dataset using the Term Frequency-Inverse Document Frequency (TF-IDF) metric [51]. TF-IDF is a statistical measure that evaluates how relevant a word is to a document in a collection of documents. In this specific case, a document is a WO's description. This value is obtained by multiplying two metrics: how many times a word appears in a document, and the inverse document frequency of the word across a set of documents. The inverse document frequency represents how common is a word in the whole corpus of documents. On the one hand, if a word is common across several documents, TF-IDF does not provide specific information about one particular document, so its value is low. On the other hand, if a word appears frequently in one given document but infrequently on the remaining documents, this means that the word has a strong relationship with the document, can be used to characterise it and has a high TF-IDF value. Putting this concept in practise, for the case of the WOs, it is possible to say that if a word is used in a WO and it is rarely found in the remaining ones, that given word is very important to describe the given WO — high TF-IDF value. If a word is found in a multitude of WOs, it is not relevant to characterise any given WO and its TF-IDF value is low. Figure 3.2 presents an excerpt of the dataset generated through the TF-IDF technique. The rows correspond to the WO identifiers and the columns correspond to all of the words found in the corpus. Therefore, each cell represents the TF-IDF value of a given word (presented in the column) and a given WO (presented by its id, in the row). A TF-IDF value higher than zero means that that given word was found in the given WO. Values of zero mean that the word was not found in the given WO.

As most TF-IDF values are around 0.5, we can formulate the hypothesis that the lexicon used to describe the WOs is technical, and their usage is very dispersed. This means that there are several words that are used in the same way but in different contexts (e.g the word "substituição" applies to every situation where a given component needs to be replaced, independent to what type of asset it is). The simplicity of the TF-IDF metric was not enough to draw any other findings. From this knowledge, the next step was to try to find groups/topics in the corpus of the WOs descriptions.

Topic modelling [62] consists in an analytical tool for evaluating data, more specifically to identify groups/topics in the corpus. The algorithm that was used was Latent Dirichlet Allocation, which is described as "*a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar*" [34]. As a result, it was concluded that the words are used too dispersedly across different fields of expertise (e.g mechanics, electricity, etc.), and the same word

	1º	2016	2017	2018	2019	21	22	23	230v	27	aferir	agua	aleração	alheta	alteração	analógico
4174	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
13548	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
16715	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
17835	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
18266	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
38001	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
35127	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
35716	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
35855	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
36349	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
37962	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
37968	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
32610	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
32979	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
33708	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
34422	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
34426	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
34686	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
31002	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
31045	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
31464	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
31571	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
31932	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
32599	0.4479899456882494	0.0	0.0	0.0	0.5417605519192981	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
28348	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
28752	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
29160	0.4681606360698589	0.0	0.0	0.566153251931674	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
29417	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
30252	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.6666098582819034	0.0	0.0
30989	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
24329	0.0	0.6407033270007815	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figure 3.2: Excerpt of the created dataset. Rows correspond to WO ids and columns to words.

is used to describe different situations; no groups that made sense, in terms of defining a given concept, were found. We were unable to find any groups that defined a concept in a meaningful way. Using topic modelling, it was possible to confirm previous findings and maintain the hypothesis that the same words are used too widely across asset management scenarios.

3.3 Word Embeddings

From this time, it was understood that more complex NLP concepts were necessary in order to further explore the corpus, so the concept of *word embeddings* was studied and applied. Computers operate mostly using numeric representations, so they do not understand textual representations the way humans do. A way to overcome this problem is converting each word into a numeric vector that tries to capture the meaning of the word through a combination of numbers, having into account its context. The idea is that each vector represents a position in a highly dimensional space, so it is possible to observe how each word relates to each other in terms of meaning. This way, for words with an approximate meaning, the distance between the vectors will be small.

This is a possible definition of what a word embedding is, and there are several libraries that can be used to generate them from a corpus of text. The ones chosen were *Word2Vec* [67] and *Fasttext* [22], which were used to create word embeddings from

scratch [52]. There was no pretension of using pre-trained word embeddings, since the lexicon found in the domain problem is technical and too specific [23]. Preceding the creation of the word embeddings, and with the intent to reduce the amount of possible words, the technique of creating n-grams was applied to the corpus (step seven of the Figure 3.1).

An *n-gram* is a combination of n words that often occur together, so the algorithm specified to find these n-grams concatenates these words using an underscore (_). For example, after the *Stanza NLP* pipeline, the description "preventive maintenance report in annex" is rewritten as "preventive maintenance report annex" (insignificant word "in" removed) and, when the n-grams are created, all its words are concatenated using underscores, since they often occur together (preventive_maintenance_report_annex). The purpose of creating n-grams is that when words appear very often together, it is possible to consider them as just one word.

Word2Vec is a two-layer Neural Network (NN) that is trained to reconstruct linguistic contexts of words. It does so by taking a corpus of words and producing a vector space with a large number of dimensions (usually between 100 and 300 dimensions are recommended, depending on the complexity of the corpus). For each unique word in the corpus, a corresponding vector in the space is assigned. Word vectors are positioned in the vector space such that words sharing similar meaning in the corpus are closely located in the vector space. It is possible to generate the *Word2Vec* model from two different architectures [67]: Continuous Bag of Words (CBoW), and *Skipgram*. Continuous Bag of Words predicts target words from surrounding context words, while Skipgram predicts surrounding context words from target words (inverse of CBoW). There is a single hidden layer in the *Word2Vec* NN architecture, whose goal during training is to adjust its weights to reduce a loss function. However, the outputs of the NN are not considered, and, instead, the hidden weights are used as the word embeddings.

Fasttext uses the same principles of *Word2Vec* but with a slight difference: it operates at the character level, while *Word2Vec* operates at the word level. Therefore, the vector for a word is made up of the sum of the word's constituents. For example, the word vector matter is a sum of the vectors of the constituents <ma, mat, att, tte, ter, r>, where '<' means Start of Word and '>' End of Word.

Both libraries are designed as a NN and provide the CBoW and *Skipgram* architectures. However, *Fasttext* uses word embeddings enrichment using sub-word information [22]. Figure 3.3 represents the *Fasttext* model, where a zoomed portion of the image allows to see some of the word embeddings. The creation of word embeddings consists in the last step of the NLP pipeline, represented in Figure 3.1.

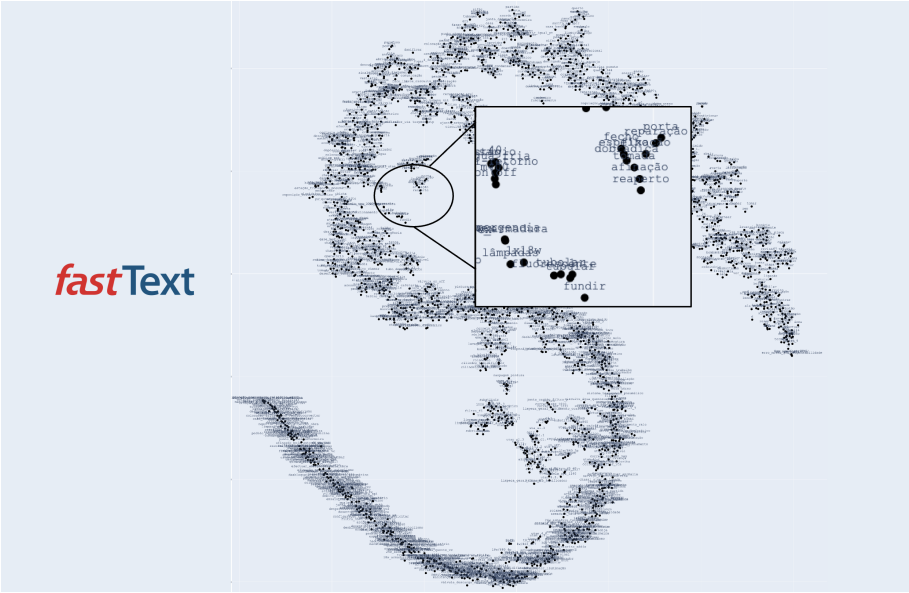


Figure 3.3: Plot of the Word embeddings learned by the *Fasttext* model, in 2D

3.3.1 Visualisation of the word embeddings

Both libraries are put to the test and word embeddings are generated using the WO's descriptions. Both the *Fasttext* and *Word2Vec* models are generated using standard values for the parameters, and from the total of 38.445 descriptions, 2435 words are learned in both models. Since each vector (word embedding) has 100 dimensions, to properly visualise the vector space, it is necessary to conduct a dimensionality reduction. A Principal Component Analysis (PCA) [45] first reduced the dimensions of the vector space in 3D and then t-distributed Stochastic Neighbour Embedding (t-SNE) [64] is used to visualise word embedding positions. The reason for using PCA before applying t-SNE is that it allows to suppress some noise and speed up the computation of pairwise distances between samples. It is important to notice that the t-SNE dimensionality reduction technique operates in a stochastic manner, so for each creation of a plot, there are slightly different representations. This happens due to the fact that one of the first steps of the t-SNE algorithm is to select a random point in space, so the algorithm reduces the dimensions having into account that first selected point. After computing several plots, it is possible to observe a common trend in all of them: words in the 3D space are displayed following a helical shape, both in the *Fasttext* model and the *Word2Vec* model. The 3D representations are dense, and difficult to analysis without an interactive interface. Thus, for displaying purposes, a dimensional reduction for two dimensions was generated, as depicted in Figures 3.3 and 3.4. Nevertheless, the same conclusions can be taken.

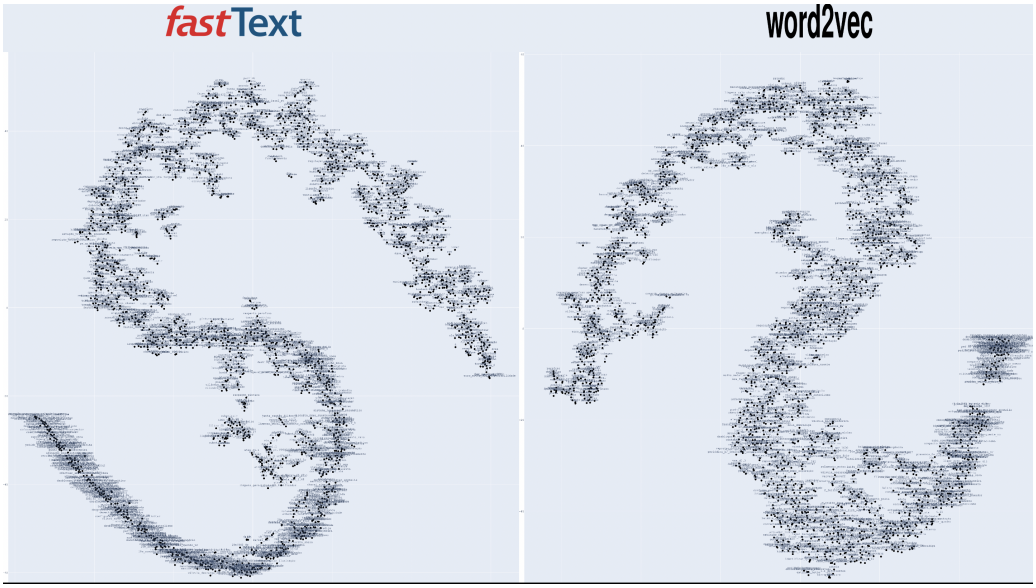


Figure 3.4: Plot of the Word embeddings learned by the both models, in 2D

The first impression that comes to mind when observing both images is that the words are displayed in a helical shape, but with slight differences between the two models. In the case of the *Fasttext* model the resulting spiral is precisely drawn, especially in the lower left corner, where the words are placed so that they almost form a straight line. In the plot related to the *Word2Vec* model, a similar spiral is observed. For the *Word2Vec* case, the spiral is drawn in a less precise way, that is, the words are not placed in a way that resembles a straight line. Accordingly, the representations appear to be mirror images of each other because the dimensionality reduction technique t-SNE [64], which was applied, is stochastic in nature. The reason for a less precise placement of the words in the case of the *Word2Vec* model might be explained by a lower accuracy of the *Word2Vec* algorithm in determining with precision the meaning of each word, that is, the computed word embeddings aren't as precise as the *Fasttext* ones. This probably occurs due to the fact that the latter uses sub-word information to generate the final word embeddings, while *Word2Vec* only uses the words. This hypothesis is analysed and discussed further in this document.

Moreover, it is important to analyse the meaning of a helical shape in the context of the visualisation of a word embeddings model. The fact that the observed shape is a spiral helps to understand that the corpus of words where the model was trained on do not present any relevant groups/topics. If that was the case, there would be dense groups of words (e.g. a circular shape made of several words), which meant that the words present in that group define an abstract concept in the context of the domain (e.g. if the domain was food then a topic that represented the concept of fruit would have the words apple, peach, pear placed together, in a dense way). Since the shape is a spiral,

defined by semi-straight lines, it is possible to conclude that there are no topics in the descriptions of the word orders, as affirmed previously with topic modelling. The words are dispersed across different areas of expertise (e.g. mechanics or electricity), and the same word is used to describe different situations, so no groups are identified.

3.3.2 Performance of the word embeddings models

It is important to understand how to compare the performance of the word embeddings generated by each library, to find the one that fits best with the domain of asset management and its intricacies.

The comparison of both word embeddings models is made through the querying of words that are semantically similar to a given word, assessing each model's response, and studying whether the returned words are coherent. This is done after the word embedding models are created, and using specific methods in each library that allow to find the closest words (in the vector space) to a given inputted word. While the semantics of the word and the domain problem cannot be objectively compared, they do need to be qualitatively compared.

The first example of the words searched is the word 2020. It was chosen in order to understand how well can the models recognise numeric concepts like an year. On the one hand, the *Fasttext* model returned the words 2019, 2017, 2018 and 2016 as the closest words. On the other hand, the *Word2Vec* model returned the words alarm, cable_passage, anomaly, and collage. The second example is break_room, and it is chosen to evaluate if the models can manage to formulate relations using n-grams. The words returned by *Fasttext* were delivery_room, work_room, waiting_room and room. The words returned by *Word2Vec* were floor_six, dressing_room, fall (verb) and fix. The third example is employee, an important word in the domain, and the words returned by *Fasttext* are schedule, malfunction, employee_entrance and necessary. The words returned by *Word2Vec* are sug (not a word, probably it is an acronym from the domain), badly, together and key. The last example is a more curious case to analyse due to the fact that the words returned by the *Fasttext* model were not as strongly semantically linked, but it is possible to understand that there is a relationship between an employee (queried word) and the returned words — schedule, malfunction, employee_entrance and necessary. For the cases of the words malfunction and employee_entrance it is possible to deliberate that the *Fasttext* algorithm captured the existing relationship between the queried word and these responses, since a malfunction is fixed by an employee and the entrance of an employee has the concept of employee in itself. For the other two cases, it is not possible to assume whether the *Fasttext* model captured the

relationship between an employee and a schedule (an employee has a schedule) and between an employee and the concept of necessity (an employee is needed to perform some operation), or if there was a matching of sub-word information, since in Portuguese the three words (employee, schedule and necessary) end with the same four letters, that is, "funcionário", "horário" and "necessário", respectively.

From the exploration of many examples, where these three are just a representative sample, it was possible to conclude the semantical accuracy of the word embeddings generated by the *Word2Vec* model is poor. This is visible because the returned words had little or nothing to do with the queried word in most of the examples. On the contrary, the *Fasttext* model responded to the queried examples with words that generally made sense — are semantically similar — or are related. Either way, the results provided by the *Fasttext*'s word embeddings made sense in most of the queried examples, although in more technical words from the domain the returned words did not make much sense, so there is room for improvement that can be further explored by tweaking the parameters of the *Fasttext*'s train algorithm. Since the objective of this thesis is not to progress towards finding an accurate word embeddings model, since the idea is to use them only as a way to analyse the corpus and find relationships between words and how they are used, this path of improving the *Fasttext* model is not taken. However, after this comparison and performance evaluation, it is possible to further explore the analytical capabilities of the *Fasttext* word embeddings model.

3.3.3 Exploration of the *Fasttext* model

The next step taken consisted in using the *Fasttext* model to better understand how the usage of the words might differ when comparing between subfamilies. In the context of the case study, there are assets that cover a total of one hundred and eight different subfamilies. A subfamily is a domain specific from second level classification, where assets with similar characteristics (e.g. similar components) are grouped together to form a subfamily. The idea is to understand if the same words are used across the subfamilies and, more importantly, how they are used. The first analysis that was conducted consisted in plotting a *Fasttext* model with all of the words from the entirety of the WO and identify the twenty most used words in a given subfamily. Figure 3.5 presents the *Fasttext* model, this time in three dimensions, and the twenty most used words for the subfamily "Rede Águas Residuais Doméstica" (Domestic Residual Water Network). It is possible to observe the dimensionality reduction of the model to 3D maintains the same spiral-like shape as in 2D.

What stands out immediately is how the words are scattered across the spiral-like

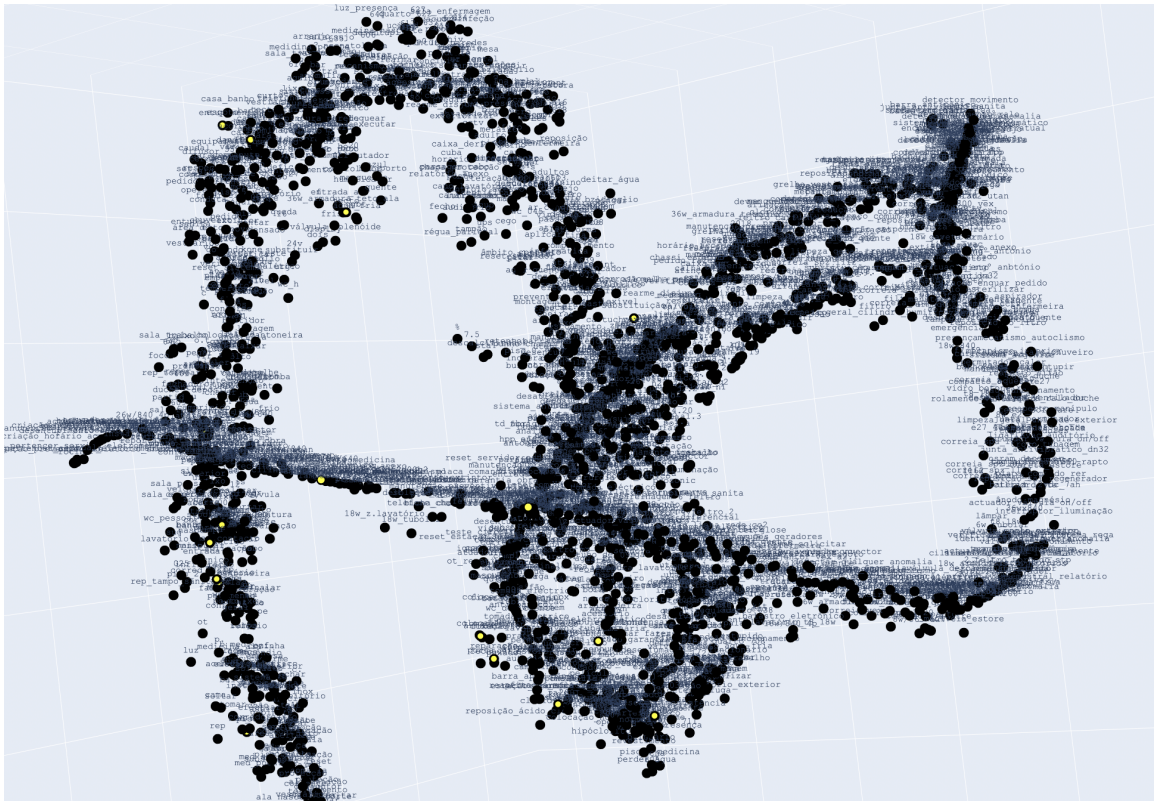


Figure 3.5: Plot of the *Fasttext* model with the 20 most used words for subfamily "Rede Águas Residuais Doméstica"

shape. Word embedding models are generated based on the meaning of the words and through the contexts where they are found. What would be interesting to see would be the identification of words that appear in a common context (the subfamilies), and so their position would be close in the model. This is not verified, so it is assumed that the *Fasttext* model could not capture the fact that some set of words are correlated with a given subfamily. Now, the question is whether that correlation actually exists, and it is possible to link a set of words with a specific subfamily, or the words are used in such a sparse way that all of them are found in most of the subfamilies. Additional explorations were made with different subfamilies, however the same was verified: the most important words in a given subfamily are found placed in a scattered way in the *Fasttext* model.

In order to further explore the *Fasttext* model regarding its capability to identify subfamily specific information, another analysis was made. The idea is to find the most important word in a given subfamily (the most frequently used) and observe what are the closest words to it. This analysis is made in the first place with the subfamily "Rede Águas Residuais Doméstica" (Domestic Residual Water Network), where the most frequent word is sewer. Then, it was possible to analyse what are the closest

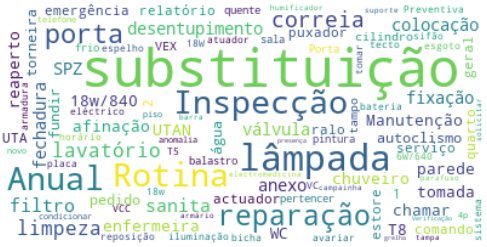
words to sewer through the verification of the position that each vector encodes and finding what are the closest positions to the word *sewer*. It is also important that the closest words are verified in the same subfamily, so if a word is closely related but not found in the subfamily, it is dismissed. The algorithm to find the closest neighbours to the given word consists in a k-dimensional tree [31], which is a space-partitioning data structure for organising points in a k-dimensional space (where k equals to the length of the vector's dimension, in this case); the cited article ([31]) describes the details of its implementation, which is available in the *scipy* Python module. The conclusions are that the words photo (photography), pantry, exterior and zone are the closest neighbours. Reminding of the hypothesis that words that appear in similar contexts should be placed closely in the word embeddings model, it would be expected that the words that are the closest to sewer would be used in the same contexts, or at least in the same phrases that appear in the WOs from the subfamily "Rede Águas Residuais Doméstica". However, they practically don't appear alongside with sewer, with only one evidence of the word pantry.

Other tests in other subfamilies were conducted, such as in the "Proteção Contra Intrusão" (Intrusion Protection) subfamily, that belongs to the *Information Systems* family. The most frequent word found in the WO s that belong to this family is tag. Applying the same algorithm as before, the closest words to "tag" are several, detector and according_to. Again, it was verified that these words do not appear alongside tag, neither they appear in similar contexts.

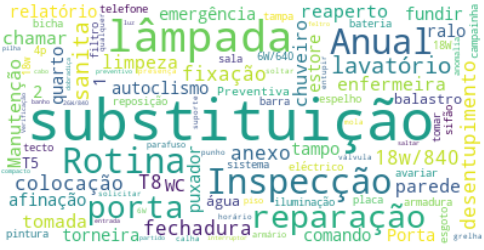
From these findings it is possible to reach a conclusion: Each word's position in the word embeddings model is not influenced by subfamily specific information, but rather by the information present in the entire corpus. In order to understand whether the word embeddings model was lacking accuracy and could not learn that specific subfamily patterns, or simply if there weren't actually subfamily specific patterns in the words and how they are used, tag clouds are created.

3.4 Visualisation of *Tag Clouds*

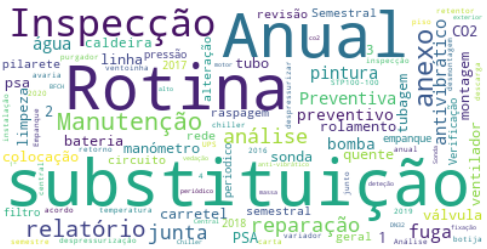
The next step taken is the creation of tag clouds. The 38.445 WOs are filtered and grouped by subfamily, and each description is used to generate a *tag cloud* per asset subfamily (from Figures .3.6a to 3.6c). *Tag clouds* consist in a visual representation of words, where each word appears represented according to its frequency in the corpus. The WOs descriptions are used to create the tag clouds, and it is possible to observe the most frequent words standing out due to the size of the font and/or the colour. A tag



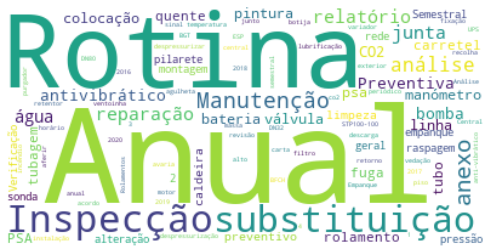
(a) *Tag Cloud* computed for all the Work Orders



(b) *Tag Cloud* for subfamily "Piping"



(c) *Tag Cloud* for subfamily "Chiller"



(d) *Tag Cloud* for subfamily "Gas Centrals"

Figure 3.6: Illustration of some of the generated *Tag Clouds*

cloud is also computed using all the WOs (Figure 3.6a) to have a general view of the lexicon used. The words are in Portuguese because the main objective of presenting these tag clouds is to analyse the frequency of each word in the subfamilies Piping (Figure 3.6b), Chiller (Figure 3.6c) and Gas Centrals (Figure 3.6d), along with the tag cloud that provides a general view of the lexicon (Figure 3.6a).

The words that stand out instantly in all of the tag clouds, thus for any arrangement of WOs, are the same, namely, substituição, which means substitution, inspeção which means inspection, reparação which means repair and others less noticeable like rotina, meaning routine, and anual, meaning annual. It is curious that in three very different asset subfamilies, the more frequent words are practically the same, just as in the global tag cloud (Figure 3.6a). A possible cause for this finding is that the listed words can be used generically in the domain. For example, the word substitution can be used in a multitude of situations where it is necessary to perform the replacement of some component. This leads to the dissemination of the word across all subfamilies. This consideration also allows us to corroborate the hypothesis discussed in the previous sections: the words present in the corpus are used dispersedly, and the same words are used in different contexts, meaning there seems to be no groups of words used in specific contexts. Therefore, it is possible to conclude that the word embeddings model did not find subfamily specific information because it practically does not exist. The same words are used to describe very different asset maintenance situations, making them less informative. So, the differences between the lexicon used in these three subfamilies can only be detected when observing the less frequent words.

3.5 Conclusions on the Exploration of the Data

From the observation of the tag clouds and the conclusions taken from previous exploration techniques, it is also possible to conclude that a general ML model trained on the entirety of the WOs descriptions will not likely yield good results, due to the fact that the global tag cloud presents roughly the same most important words as the three subfamilies, so it may have difficulty in identifying subfamily-specific data that is required for it to learn properly. Therefore, it is necessary to consider other solutions, such as a ML model per subfamily. The downside of this solution is that it requires managing a large number of models, but it has the advantage of retaining the specific details that are unique to each subfamily. The option of using a single ML model presents itself as a harder challenge due to the fact that the learning process lies on patterns and intricacies present in the data that can be detected; if the same words are used in the same way in very distinct asset management problems and corresponding subfamilies, it would be very difficult to detect those patterns.

Another important consideration is that the frequency with which words appear is not the only indicator of importance. Domain knowledge allows one to understand that words that rarely appear (e.g. one or two times) might be equally as important as words that appear very often. This happens due to the fact that critical failures are not common in asset management, so words used specifically in these situations won't appear very often. However, when these critical failures occur, they present very important information that needs to be included in asset management decision-making processes. Subfamily-wise ML models, since they are smaller and deal with specific patterns, will in theory have a better capacity to "keep an eye" in these rarer, important cases.

All of the research documented in this chapter was published in the International Conference Information Visualisation [52], where the main theme is the visualisation of the word embeddings and how it was possible to evaluate the most adequate one for the given domain.

4

Word Recommendation Algorithm

This chapter presents the development of an asset management word suggestion algorithm. It discusses possible solutions that can be implemented to solve this problem and describes the chosen one, with the respective justification. Following the CRISP-DM [13] methodology, the research progress documented in this chapter consists in the data preparation and the modelling phases.

4.1 Existing Word Recommendation Algorithms

There are several state of the art algorithms that can be used to complete word suggestion and sentence completion tasks. The most recent models with state of the art performance are based on transformer architectures [66], where algorithms like Generative Pre-Trained Transformer (GTP-3) [50], Bidirectional Encoder Representations from Transformers (BERT) [15] or Robustly Optimized BERT Pretraining Approach (ROBERTA) [37] can be found. These algorithms have been considered to solve the problem at hand in this research due to their outstanding performance, however when there is an understanding of how they work it is safe to say that the cost of using them is much higher than the benefits they bring. The first reason is because these algorithms need vast quantities of data, and in the given case the amount of data that exists is not sufficient. There would be the need to create a lot of synthetic data. The second reason is that they are complex models that take a considerable amount of time to train and usually require more space than traditional ML models [29]. As concluded in the

previous chapter, there is the need to create one hundred and eight different models, which will take considerable proportions if each of these models is as complex and "big" as the ones discussed.

Still in the area of deep learning, there was the consideration of using Long Short-Term Memory (LSTM) Neural Network [68] to solve the problem at hand. However, this kind of NN also requires a lot of data in order to learn properly, which leads to the need of having to generate synthetic data. The problem with generating data synthetically is due to the fact that in this specific domain of asset management there would be the need to verify if the generated data makes sense, usually by a technician of the domain. Therefore, it can be said that synthetic data generation is dependent on domain experts validation, which can be difficult to obtain in a timely manner. Furthermore, each WO occurs at a specific time and asset, which means synthetic data generation is time and location dependent. Consequently, it is more difficult to generate reliable synthetic data.

Another algorithm that was considered were the Hidden Markov Models (HMM) [28]. It consists in a statistical model that can be used to describe the evolution of observable events that depend on internal factors, which are not directly observable. In the specific case of NLP, Hidden Markov Models (HMM) are used to model Part of Speech Tagging [46]. This solution is not considered due to the fact that the part of speech types that are found in almost every Work Order (WO) are the same, therefore the added complexity of dealing with part of speech tags does not present a high benefit in the specific case of the sentence constructions found in this domain of asset management. However, the solution that was implemented is inspired by the Hidden Markov Models (HMM) [28].

Due to the need of creating and managing a considerable amount of different models, plus the fact that the technicians use tablets to create the Work Order (WO)s and these might not have internet connectivity in some of the places where asset management situations occur, there is the need to solve this problem with a simple but effective ML model that can be easily maintained and adapted (optimised) over time. The implemented solution consists in a Bayesian Network (BN) with implicit user feedback, which allows it to adapt to the present needs of the technicians.

4.2 Recommending Words using a Bayesian Network

After understanding how the words are used across all the WO and the relationships between them, the next step is to create a recommendation system that has the functionality of suggesting a word given a context; this context is composed of previously inputted words. The chosen algorithm is a Bayesian Network (BN), which consists in a type of probabilistic graphical model that uses Bayesian inference for probability computations. BNs aim to model conditional dependence, and therefore causation, by representing conditional dependence by edges in a directed graph. A BN is a directed acyclic graph, in which each edge corresponds to a conditional dependency, and each node corresponds to a unique random variable.

When applying these concepts to the problem at hand, the occurrence of a given event is understood as the occurrence of a given word in a sentence. Hence, the BN is modelled by considering the occurrence of a word as an event with a corresponding probability. For example, if the sentence "fan assembly" is considered, the probability of occurrence of the word "fan" is given by $P(fan)$. The probability of occurrence of the word "assembly" is given by $P(assembly)$. Therefore, the conditional probability of occurring "assembly", knowing that "fan" occurred, is given by $P(assembly|fan)$. A frequency analysis of words in the corpus allows us to calculate conditional probabilities by identifying words that occur together. The network makes inferences based on these probabilities. The BN is created with a specific structure, described in the next paragraph, using the *Pomegranate Python framework*[1]. The probabilities found in each state of the network are calculated by analysing a set of WO descriptions. These descriptions serve as a reference for the recommendation of words. In other words, the capacity for the network to suggest words will be tailored according to the sentence constructions verified in the historic file that contains the WO — these can be altered by providing a different file, with different descriptions, when constructing the network. Figure 4.1 presents an excerpt of a file that can be used in the network's creation. The frequencies found for each word and the other words that are its neighbours is what enables to model the probability distributions. The structure of the BN is composed by having each state representing the n-th word in the given sentence, i.e. the first word is associated to the first state (State 1), second word to the second (State 2), and so on as illustrated in Figure 4.2a. The idea is that the connections between states represent words that appear together in the WO. In State 1, the probability distribution, represented by P, is created by analysing the words that appear as first word in the sentence and calculating their respective frequency. For the next states, the procedure is slightly

[1] https://pomegranate.readthedocs.io/en/latest/BayesianNetwork.html

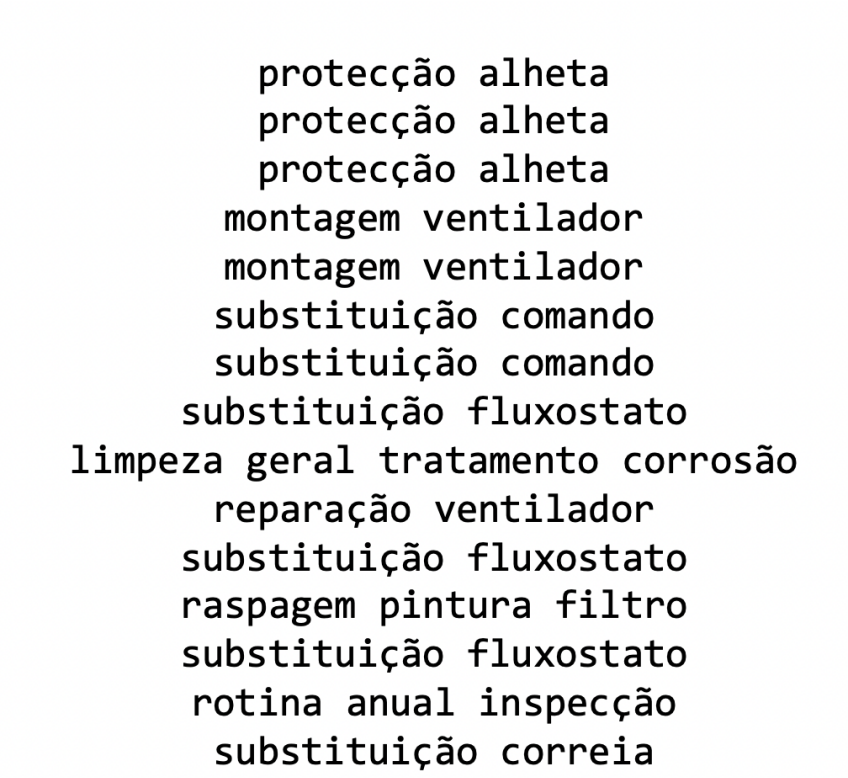


Figure 4.1: Excerpt of one example file used to create the BN and its probability distributions

different. The *P*(State 2) has into account the previous words and the current words. Therefore, *P*(State 2) is modelled by the frequency in which bi-grams occur, i.e. a bi-gram is a combination of a word from the State 1 of the sentence with another from the State 2 (e.g. "válvula" has position 2 in the sentence "substituição válvula" (valve substitution)). Hence, the probability table that allows to model the *P*(State 2) has all the possible combinations between words verified in the State 1 and words verified in the State 2. Bi-grams that are found in the inputted WO will have associated their respective probabilities of occurrence.

For the remaining states in the network, there are two possible implementations, depicted in Figure 4.2. The only difference between them is that the first one does not respect the Markov property (Figure 4.2a), while the second one does (Figure 4.2b). The first possible implementation regards each state as being dependent on all of the previous ones. The second possible implementation regards each state as being dependent on only the previous one — this is the definition of the Markov property, which states that a given state has all the information that is needed to proceed to the next state in itself and in its previous neighbour. In order to better understand which of the

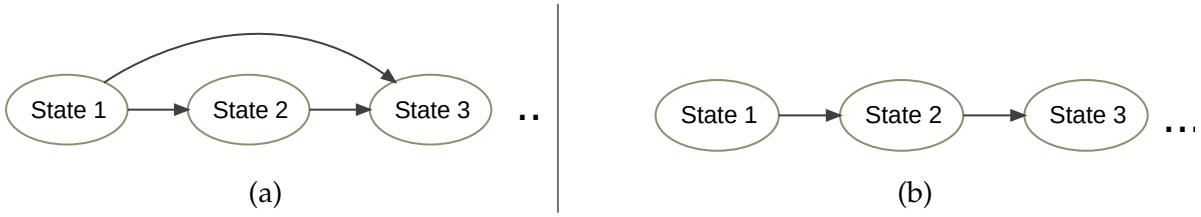


Figure 4.2: A fully dependent BN (a) and a BN that respects the Markov property (b).

two options is more adequate, the first implementation, which considers that a given state is dependent on all previous states, is described hereafter.

After modelling States 1 and 2, in State 3 there is the need to take into consideration the words chosen in previous states. An argument that the first implementation has in its favour is that since any given state is dependent on all the previous states before it, the recommendation that is given has information about all the previously inputted words by the user. The problem with this assumption is that when considering all the previous states, all the combinations of words found in the previous states have to be considered. This means that the probability table of any given state would have to express all the combinations between the words found in previous states. This can be achieved by gathering a list of words that appear in position one, and another list of words in position two. Finally, combine all the words to form all possible sentences. The need to perform this combination of words leads to exponential growth as the number of states increases. This is due to the need to consider all the prior words in all of the previous states, which will require high memory usage. Depending on the number of states, this approach may be unfeasible. If a network with five states is considered, for example, and there is an average of thirty words per state, there would be the need to perform 30^5 combinations in order to model all possible sentence constructions in the probability table of state five. This entails high computational costs, that do not present a significant gain.

Due to the exponential growth problem, it is defined that the BN is created having into account the Markov property, the second implementation, as shown in Figure 4.2b, which means that a state is conditionally independent of the previous states, except for the previous one. This way, the joint probability tables associated to each state are significantly reduced, since they only contain bi-grams — formed by the words of the current state and words from the previous state. The structure of the network is then simplified, and each state only depends on the previous one. This simplification allows to drastically reduce memory usage, and it does not prejudice the ability of the network to conduct inference and answer queries adequately [3] — chapter two of

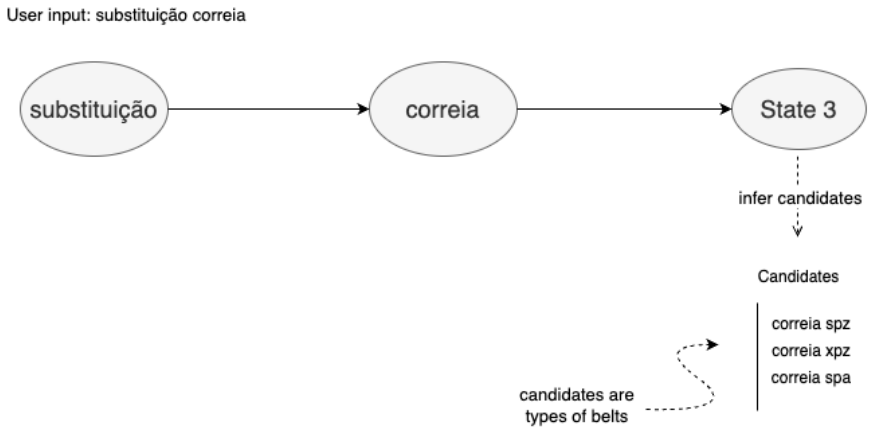


Figure 4.3: Illustration of the algorithm for predicting the next word

the referenced book introduces BNs and presents an overview on the Markov property. The fact that the sentence constructions in the WOs are composed of up to eleven words maximum (and sentences that big are really rare), where the average for the 38445 WOs is $3,32$ (about 3) words per sentence also allows to understand that choosing the second implementation is more adequate.

4.2.1 Bayesian Inference

Inference in the network can be conducted in two different ways: (1) infer the next word, or (2) infer the remaining words that allow to complete a sentence. Both alternatives are based on previously inputted words and, in both cases, inference is driven by the use of Bayesian inference.

For the first case of inference (1), the procedure is to look at how many words are already inputted and consult the state in the same position as the word to be suggested (e.g. in the network depicted in Figure 4.2b, if a sentence with 2 words is inputted, State 3 will be consulted to infer the next word). Since the BN respects the Markov property, the only factors that influence the network's response to an inference of the next word are found on the current state and its previous neighbour. The candidates for next word are found through the application of the Bayes theorem.

For example, if the user inputs the words "substituição correia", the state that is going to be consulted for finding the candidates for next word is State 3 due to the next word being in position 3, as depicted in Figure 4.3. In this state, the algorithm looks for all the bi-grams that begin with the word "correia" and sorts them according to their probabilities — highest first. The probability of each bi-gram is calculated through a frequency

analysis on the inputted corpus of text (input file) during network creation. This operation respects word position, that is, for example, State 3 only contains bi-grams that are composed of words found in position 2 and 3 of a sentence. The candidate bigrams that are returned are types of belts, and the words that are suggested are the last word in each bigram (spz, xpz and spa).

In the second case of inference (2), it is important to describe that the sentence is considered to reach its end when the network does not have more information. This can happen whether because there are no more states in the network, or because there aren't found bi-grams, in a given state, that allowed to recommend any word that made sense together with the previous word. Moreover, the inference of a whole sentence requires the search of the whole network for possible word combinations that make sense. The implemented algorithm of inference makes use of recursive programming [16] to travel through the network; the idea begins with the initialisation of a matrix M, depicted in Figure 4.5, where possible sentence constructions are saved as rows. When the algorithm is done, this matrix will have all the sentence constructions ordered by highest probability. A cell of the matrix yields a tuple formed by the word and corresponding probability of occurrence given by the Bayes theorem. The algorithm is illustrated in Figure 4.4, representing a BN with four states. The algorithm for searching the network begins by using the first kind of inference — inference of the next word —, having as input the last word entered by the user (Step ① in Figure 4.4, word a is representative of any given word that the user could input); the Top@5 words that are better suited to succeed as the next word are returned. Since the algorithm works by searching in depth, the next step is to look at the first word returned from the group of 5, and infer the next possible candidates to this word in the next state using inference of the next word again (Step ②). In the next state, only 2 suggestions are given, and the depth of recursion is now at level one. From this new group of 2 possible candidates, the first one is chosen again to infer the next word in the next state (Step ③). The depth of recursion is now at level 2, and no candidates are verified in State 4. When there are no more candidates, the words present on the path traversed during the search are appended to the matrix to form a sentence; information about previous chosen words are kept in order to when there are no more candidates the matrix line can be filled with the resulting sentence. In the next step, (Step ④), the algorithm goes back to the previous state (State 3) and infers the next word of the second candidate. The procedure continues, always searching in depth through recursion until there are no more words and, every time that is verified, a new sentence construction is appended to the matrix. The algorithm chooses a word in a given state and searches for the next group of candidates to succeed this word, in the next state. If the next state has candidates,

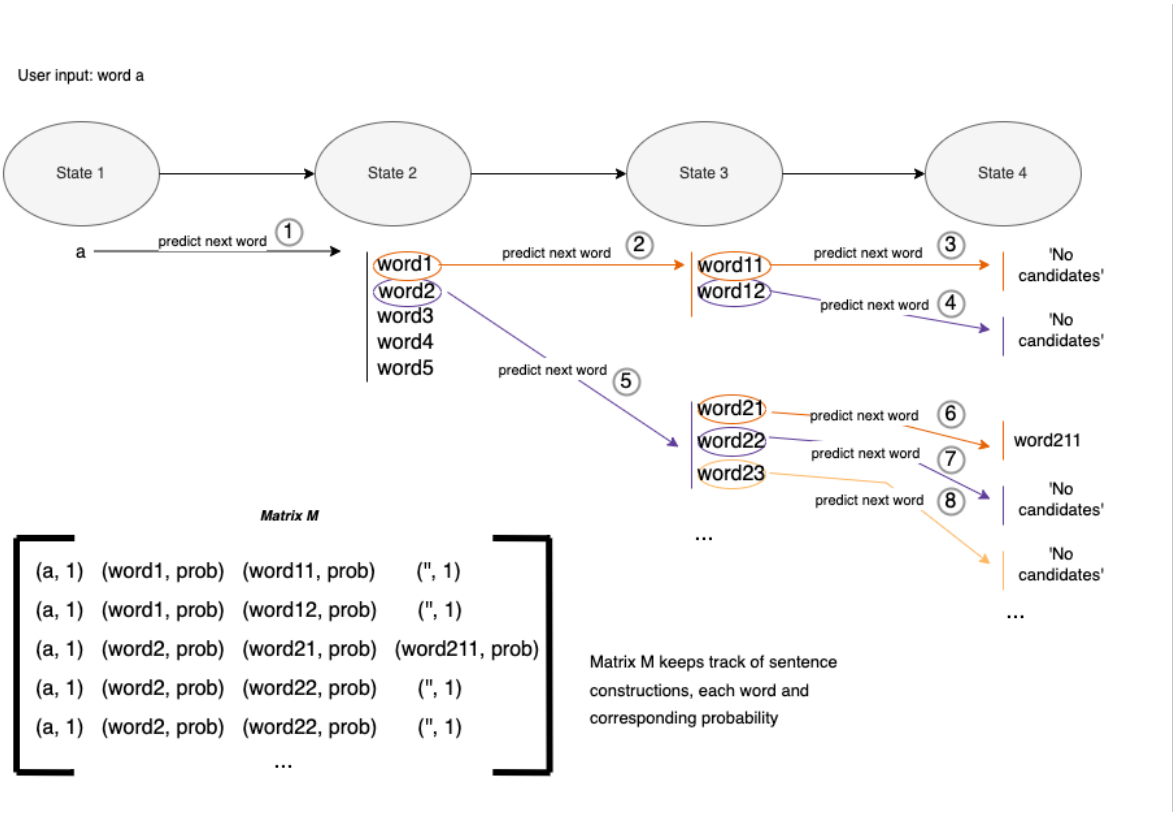
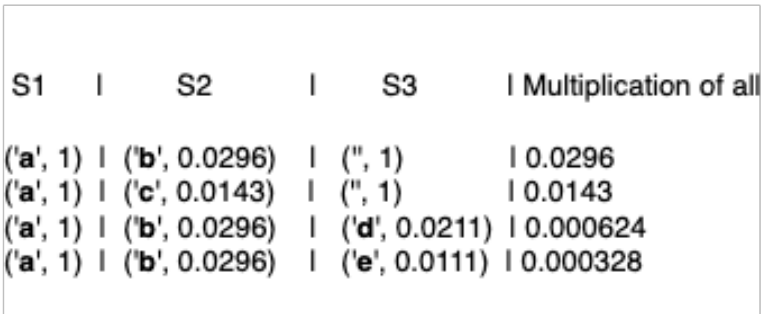


Figure 4.4: Illustration of the algorithm for inference of a whole sentence

then the same procedure is applied for each candidate at a time. If the next state does not have information about possible candidates to the given word, the sentence is considered to be constructed, it is appended to the matrix, and the focus shifts again to the previous state. Then, the next word from the list of possible suggestions is considered and the process repeats itself until all the network is searched.

Therefore, the *modus operandi* of the algorithm is to look to the suggestions given by the inference of the next word, and recursively search for the next five suggestions for each of the words in the next state, until there are no more possible candidates. In the end of this inference of the whole sentence, the matrix created in the first step has the sentences constructions, that are then accordingly sorted, after the traversing of the network is completed. The sorting procedure is done by multiplying the probability values associated to each word, with each other, in each sentence (line). An example of the returned matrix can be observed in Figure 4.5. The matrix holds possible sentence constructions, ordered by highest probability of occurrence. It has the States 1, 2, and 3 depicted in columns (exemplifying a network with only three states), followed by the resulting probability given by multiplying all of the probabilities associated with each word in each sentence. **a**, **b**, **c**, **d** and **e** are example words.

The fact that the sorting is done through multiplication makes the final probabilities



S1	S2	S3	Multiplication of all
('a', 1)	('b', 0.0296)	('', 1)	0.0296
('a', 1)	('c', 0.0143)	('', 1)	0.0143
('a', 1)	('b', 0.0296)	('d', 0.0211)	0.000624
('a', 1)	('b', 0.0296)	('e', 0.0111)	0.000328

Figure 4.5: Sketch of the matrix returned from the inference of a whole sentence

in each suggestion small. The lines with the highest probability are considered first, and sentences that are shorter are given priority, considering the tuple ("", 1) - empty string with probability one - for cases where no candidates were found. In the given modelling in Figure 4.4, the network structure presents four states. Knowledge from the domain specifies that WO descriptions should be small and straight to the point, so the cases with big sentences that would require support for bigger networks are rare in most subfamilies, which allows to avoid the problem of the network occupying a lot of space. The pseudo-code for the algorithm of sentence completion can be found in algorithm 1 and algorithm 2. It is divided into two algorithms for the sake of simplicity and readability, where algorithm 1 has as input parameter the words inputted by the technician ("known_words"), and algorithm 2 is the recursive algorithm in which the traversing of the network occurs.

Algorithm 1 Predict Sentence

Require: $self \neq NULL$; $known\_words \neq NULL$;
  if network does not know any of the words in *known_words* **then**
    return []
  end if
  $known\_words\_tupled \leftarrow list(map(lambda\ x{:}\ (x, 1), known\_words))$
  $suggestions\_matrix \leftarrow$
    $self.predict\_sentence\_possibilities(known\_words,$
      $known\_words\_tupled, [])$ ▷ call to algorithm 2
  $self.calculate\_final\_probabilities(suggestions\_matrix)$ ▷ multiply all the probabilities in each word for each found sentence
  $suggestions\_matrix.sort()$ ▷ sort by highest probabilities first
  return $suggestions\_matrix$

Algorithm 2 Predict Sentence Possibilities

Require: $self \neq NULL$; $known\_words \neq NULL$; $known\_words\_tupled \neq NULL$;
$matrix \neq NULL$;
$known\_words\_level \leftarrow len(known\_words)$ ▷ last state with known word
if $known\_words\_level >= len(self.number\_of\_states)$ **then**
return $matrix$
end if
$level\_bigrams \leftarrow$
$self.states\_distributions[known\_words\_level]$ ▷ states_distributions has the probability distributions of each state
$candidates \leftarrow$
$self.predict\_next\_word(known\_words\_tupled, level\_bigrams)$ ▷ predict_next_word returns a list of tuples, where each tuple is a combination of the candidate to be the next word and the associated probability
if $len(candidates) == 0$ **then** ▷ if there were no candidates
if has not reached last state **then**
fill remaining states with $('', 1)$ in $known\_words\_tupled$
end if
$matrix.append(known\_words\_tupled)$
return $matrix$
end if
for $candidate$ in $candidates$ **do** ▷ when there are candidates
if $candidate[0]$ not in $known\_words$ **then** ▷ to prevent suggesting words already in the sentence
$known\_words\_tupled\_plus\_candidate \leftarrow known\_words\_tupled$ +$candidate$
if has not reached last state **then**
fill remaining states with $('', 1)$ in $known\_words\_tupled\_plus\_candidate$
end if
$matrix.append(known\_words\_tupled\_plus\_candidate)$
$self.predict\_sentence\_possibilities(known\_words,$
$known\_words\_tupled\_plus\_candidate, matrix)$ ▷ recursive call to find candidates from the new known word (current candidate)
end if
end for
return $matrix$

4.2.2 Bayesian Network Domain Adaptation

The asset management domain is volatile in terms of the situations that may arise, that is, domain knowledge allows to understand that different assets malfunction at different times and that there are different preventive operations at different times, so distinct WOs are created. This means that the network's suggestions need to be adapted according to present situations. Given the fact that the probability distributions of the network are created using a specific set of historic WOs, the validity of those probabilities, in terms of how close they are to present reality, expire over time. In order for the network to adapt to the needs verified at the present, the feedback of the user (technician) is included through the analysis of the chosen suggestions. The idea is to reward asset management situations that have happened recently for the last time, for example, if the operation "fan disassembly" was verified two times in the same day, then it makes sense to assume that the technicians might be doing maintenance operations that include this situation, so it is given a higher probability of being suggested by the network.

4.2.2.1 Including user feedback in the network

So, how can the inclusion of the user's feedback be achieved? As explained earlier in this chapter, the BN works by modelling in each state a given probability distribution, based on the frequencies of each bi-gram present in the corpus. The main idea behind the network's adaptation is the update of the probability distributions in each state by re-calculating the associated frequencies after a given suggestion is chosen. This adaptation is guided by a reward function, which works by giving priority to situations that happened more recently and, consequently, depends on two variables: (i) time and, (ii) maximum frequency found in a given state. Figure 4.6 presents a sketch of this reward function. As can be seen, it is a decay function that decreases as time passes, assuming that the highest rewards are given for situations that occur often. The values given for the function were initial values that appear merely out of domain assumptions, however, since this is not absolutely correct, an empirical study on the most adequate values for the function are conducted in the next chapter. It was not found any documentation that allowed to give the function different values, so this empirical study revels itself as important.

Moreover, it is observable that the values in the X-axis are sequentially larger — day, week, month, trimester. This idealisation of the values for the X-axis in this reward function is justified with the fact that in the domain of asset management, part of the

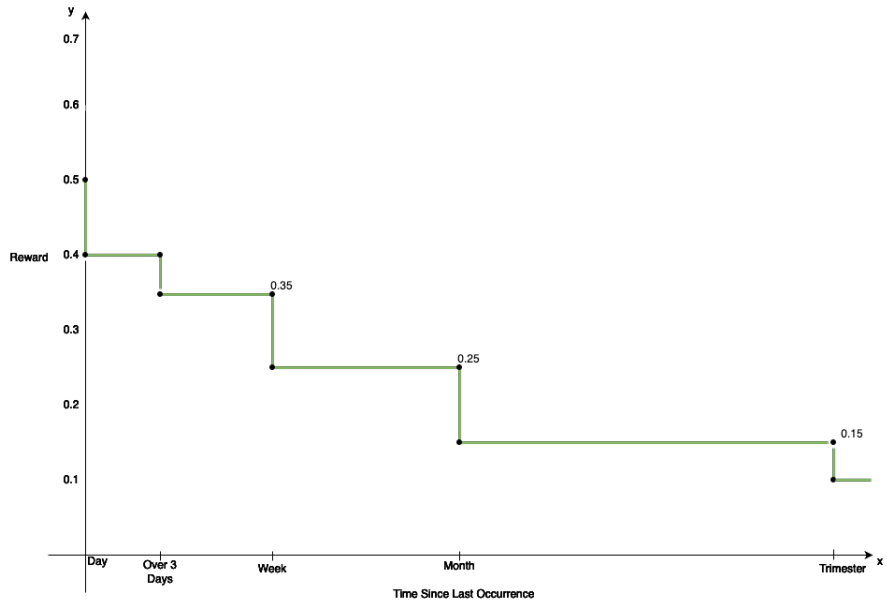


Figure 4.6: Sketch of the reward function that guides the network's domain adaptation

preventive measures consist in performing WOs in specific time spans, that is, WOs are scheduled so that technicians analyse the assets in order to prevent that the components degrade until asset failure. Since these components do not degrade by day (daily granularity), it is important to consider that for most of the WOs there will be weeks, months or even years before they happen again in the same asset. This is the main reason why the time specificity of the reward function grows exponentially, and the reward values also become smaller over time. Likewise, the lower values of the X axis, such as day and week, present higher values of reward due to the fact that if a given asset management operation occurs several times in a short time period, than it means that the technicians can be doing preventive maintenance operations in several assets of the same kind, creating several WOs that are similar. Consequentially, the reward function gives high priority to situations that happen more often, in order for them to appear as suggestions in higher rankings. The reward function is designed in a way that the recommendation system has the capacity for adaptation to the needs that the technicians are presenting. These needs are identified by the operations that are chosen in the suggestions, and if no suggestion is chosen, when the technician finishes writing the WO, the network is properly updated with the corresponding description.

Additionally, the idea is that through the reward obtained using the decay function it is possible to calculate a new value for the frequency of the words that compose the chosen suggestion. The whole procedure consists in observing the suggestion that was chosen by the technician, e. g. "substituição válvula" (valve substitution), and calculate the reward value by looking up when was the last time this particular situation

happened. All of the dates and times of when a situation last occurred are kept in the BN. Let's assume there was another valve substitution in the present day, so the reward is 0.5. After calculating the reward value, it is possible to calculate the new frequencies for the word "substituição" in State 1, and for the bi-gram "substituição válvula", in State 2, by applying the Equation 4.1.

$$NewFrequency = OldFrequency + (Reward * MaxStateFrequency) \tag{4.1}$$

By looking at the Equation 4.1, it can be understood that the new frequency for the word "substituição" is obtained by doing the sum of its old frequency with the maximum frequency found on this first state, times the reward that was given, 0.5 in this case. The weighting by the maximum frequency found in the given state is done so that the resulting value for the new frequency keeps respecting the probability distribution, so the changes that occur are not disproportionate (too big) when compared to the previous configuration of the probability distribution. To end this example, let's assume the word "substituição" had a previous frequency value of forty, and the maximum found is sixty five. The new value for the frequency of this word is given by $40 + (0.5 * 65)$ which is equal to 73. Frequencies are maintained as integers. The procedure repeats for the bi-gram "substituição válvula" in State 2, calculating the new frequency for this bi-gram. The maximum value that the new frequency can take is not defined, due to the fact that the maximum state frequency has no established maximum. The minimum value that it can take is one, because zero would mean that the given bigram never appeared in the corpus, hence it would not make sense that it was included in the network.

In the case of the technician finishes creating the WO without choosing any suggestion, the network is updated with the description and, in the case of being a description with unknown words, these are added in the recommendation system, in the corresponding states, with a frequency equal to the mean of the frequencies found in each state. It was chosen to add new words with the mean of the frequencies so that new words that are inserted in the recommendation system do not start with frequency equal to one. If it was the case, these new words would have to be chosen in the suggestions a few times, so that their frequency was raised, and they could appear in the suggestions.

The pseudo-code for the algorithm for including user feedback in the network can be found in Algorithms 3 and 4. It is divided into two algorithms for the sake of simplicity and readability. Algorithm 3 is run every time a technician chooses a suggestion from the list presented by the recommendation system. The chosen suggestion consists in the "chosen_sentence" parameter. Algorithm 4 is responsible for changing the

probability distributions of each state.

Algorithm 3 Inclusion of User Feedback

Require: $self \neq NULL$; $chosen\_sentence \neq NULL$;
$chosen\_words \leftarrow chosen\_sentence.rstrip().split()$
$reward \leftarrow self.calculate\_time\_based\_reward(chosen\_sentence)$ ▷ calculate time-base reward using reward function
$self.reconfigure\_state\_distribution(chosen\_words[0], 1, reward)$ ▷ reconfigure first state distribution
for n in $remainingstates$ **do**
    $bigram \leftarrow$ (current state word and previous state word)
    $self.reconfigure\_state\_distribution(bigram, n, reward)$
end for

Algorithm 4 Reconfigure State Distribution

Require: $self \neq NULL$; $bigram \neq NULL$; $level \neq NULL$; $reward \neq NULL$;

$prob\_table \leftarrow self.states\_distributions[level-1]$ ▷ states_distributions has the prob. distributions of each state

if *level* != 1 **then**

$distribution \leftarrow$ conversion of $prob\_table$ to dictionary

$distribution \leftarrow$ denormalisation of distribution ▷ get frequencies instead of probabilities

if *bigram* not in current state distribution **then**

$bigram\_value \leftarrow$ mean value of all the frequencies

$distribution[bigram] = bigram\_value$

$self.added\_frequencies[level] += int(bigram\_value)$ ▷ to keep record of the added frequencies, allowing to maintain the properties of the distribution

add each word in $bigram$ to $self.all\_words\_distribution$

else

$max\_frequency \leftarrow$ maximum frequency value in the distribution

$frequency\_to\_sum \leftarrow int(max\_frequency * reward)$

$self.added\_frequencies[level] += int(frequency\_to\_sum)$ ▷ to keep record of the added frequencies, allowing to maintain the properties of the distribution

$distribution[bigram] = normalized\_distribution[bigram] + frequency\_to\_sum$

end if

$distribution \leftarrow$ normalisation of distribution ▷ get probabilities instead of frequencies

$prob\_table \leftarrow$ conversion of $distribution$ to list

$self.states\_distributions[level-1] = prob\_table$ ▷ update state distribution

else

$self.alter\_first\_distribution(bigram, reward, level, prob\_table)$ ▷ first state has independent distribution, dealt in the same manner, treating single words instead of bigrams

end if

5

Recommendation System Evaluation

A ML model is created with the intent of providing a solution to a complex problem, by understanding the problem domain, the associated data and the most adequate algorithm(s) to use in the given situation. Once the model is created, there is the need to evaluate it and understand how well could it solve the problem. Hence, in the specific case of this research, there is the need to obtain metrics to understand how the recommendation system performs in real scenarios and if its suggestions make sense or not. This chapter documents the research progress regarding the evaluation phase of the CRISP-DM [13] methodology. Hereafter, the reference to the BN is equivalent to referring to the developed recommendation system.

In order to determine whether or not this word recommendation algorithm can be used in technicians' daily operations, it is necessary to evaluate its response. Ideally, the proper way to evaluate the network would be with external independent data, alongside with the judgement of several domain experts. The procedure would be to include the word recommendation algorithm in the tablets that the technicians use to create the WO. As soon as the technicians had used it for a while, they would fill out a report where they would specify how appropriate the suggestions were and summarise an accuracy for the network. This approach is problematic because there is not enough time to evaluate a large number of technicians and summarise their experiences. Due to the impossibility of performing the ideal procedure, an alternative to evaluate the solution needs to be considered. Moreover, as the state of the art regarding the evaluation of ML models in asset management is limited, no methodology is found

for evaluating solutions without the participation of technicians. As a result, the next section presents the proposed methodology for the evaluation of the recommendation system.

5.1 Methodology for Recommendation System Evaluation

The idea behind this methodology is that it pretends to simulate the process of using the BN; this process consists in what the technicians would perform in their day-to-day operations, in terms of creating WOs and observing the recommendations that the network would give. Therefore, the idea is to test the network against an unseen sequence of WO, representative of what the technicians would input. This sequence of WOs is created in a way that it represents the asset management situations that the technicians would encounter, simulating the usage of the BN as if it was included in the tablet.

In order for the influence of the sequence of WOs to be as close to reality as possible, it is taken into consideration the date of each WO. This consideration allows for the reward function described in the previous chapter to be properly evaluated. Furthermore, the order in which these WOs appear in the sequence and the time span between them are carefully taken into account so that specific asset management situations, that appear one after the other, in real life scenarios, are simulated. For this to be possible, a real set of WOs is used in the testing of the recommendation system. This sequencing of WOs is important, alongside with the consideration of how much time spans between them, in order to evaluate the model as close to reality as possible. The designed methodology is illustrated on Figure 5.1.

The observation of Figure 5.1 allows to understand that the methodology begins with a sequence of carefully chosen WOs; it is also important to state that the network hasn't "seen" this sequence before, that is, from the entirety of the WOs there is the need to take a subset for training and a different subset for testing. For the sake of simplicity in the methodology's overview, let's consider a sequence that starts with the situations — "limpeza cilindro humidificador" (humidifying cylinder cleaning) and "substituição filtro" (filter replacement). In step ① the procedure starts by taking the first sentence, looking at the first word, and querying the network for suggestions, as if the technician was inputting this WO. After querying the network for "limpeza", the suggestions are observed, in step ②. The performance indicators of this given case are calculated in step ③ by observing in what rank did the network put the word "cilindro", due to the fact that this word is what corresponds to the reality, that is, what the technician

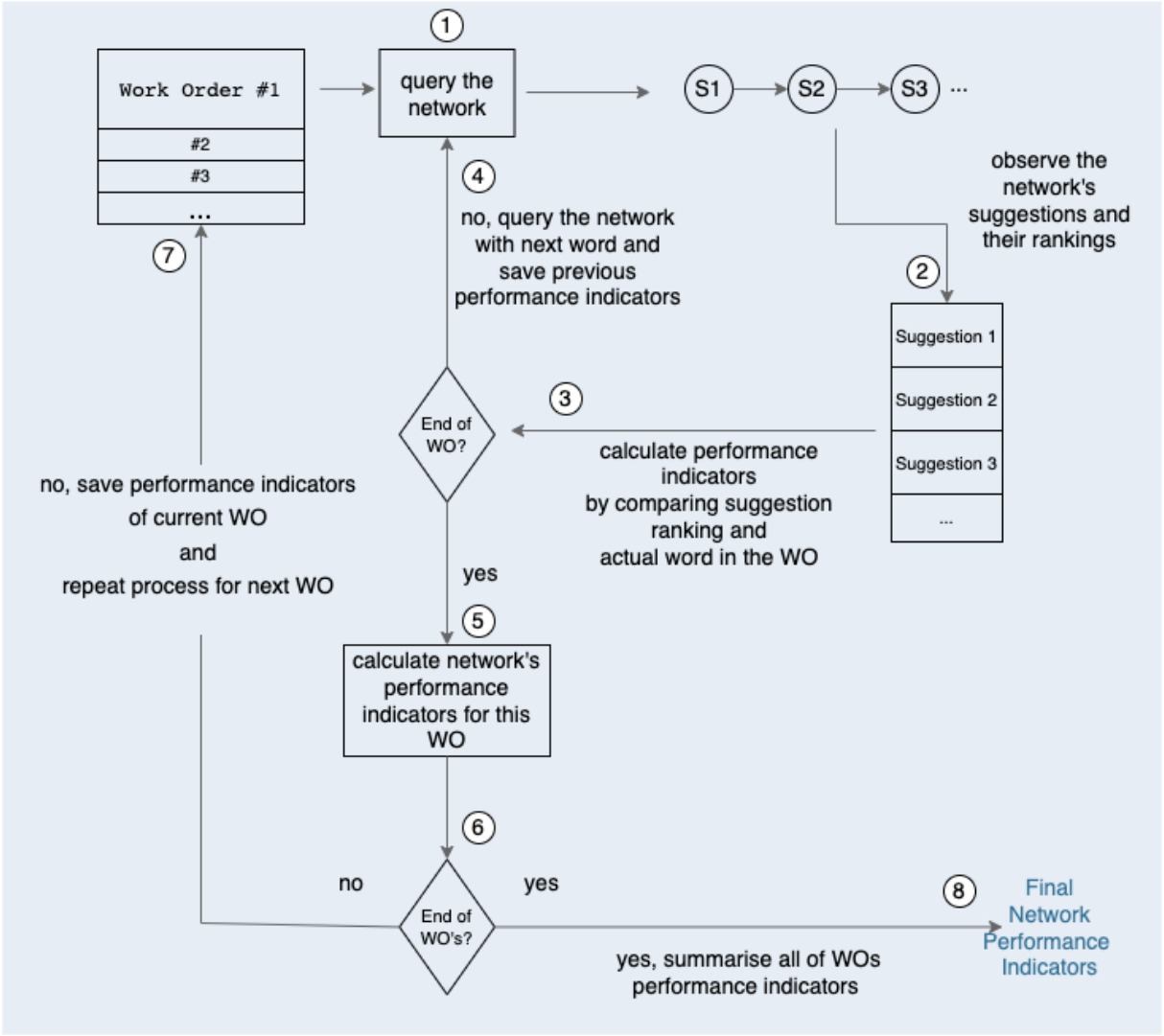


Figure 5.1: Methodology for Evaluating the BN

intended to write. The metrics that are chosen to evaluate the network are discussed in the next subsection. Let us assume that there is one metric in the performance indicators which ranges from zero to one hundred percent, named Mean Reciprocal Rank (MRR) [40] and discussed in the next subsection. Let us also assume that the network ranked "cilindro" in the first place, so that means 100% of MRR for this case. After calculating the MRR for this word, the procedure is repeated, because the first sentence still has another word; step ④ takes the procedure back to the first sentence, and now the network is queried for suggestions for the case "limpeza cilindro". The network's suggestions are observed in step ②, and in step ③ there are calculated the performance indicators for this new case. This time, the network ranked "humidificador" in place five out of ten, for example, so let's assume the MRR is 20% (the calculation of the MRR and other performance indicators is left for the next subsection, for simplicity of

explanation of the methodology). Once the performance indicators are calculated for this case and the sentence reaches its end, in step ⑤ it is possible to summarise the performance indicators for this first WO: considering only the MRR for simplicity of explanation, the average between 100% from the first suggestion and 20% from the second gives a total of 60% MRR for this first WO. In step ⑥, it is evaluated whether the sequence of WO has reached the end. Since there are more WOs in the sequence that haven't been processed, the procedure shifts focus to the next one in step ⑦, which is "substituição filtro", and saves the performance indicators of the previously tested WO. In this second WO, the procedure is repeated. Take the first word, "substituição", and query the network for it. Observe the rankings and compute the performance indicators based on what position is the intended word "filtro". Since this WO only has two words, only one suggestion is made and the final performance indicators of the sentence is the same as the suggestion.

This procedure is repeated for all of the sequences, saving each WO 's performance indicators, and summarising all of the indicator's values into one to obtain the final network's performance, in step ⑧. After a given sentence is evaluated, the algorithm for including the user's feedback is run with the WO that was just analysed, allowing for the adaptation of the network for future WOs.

The experiment of testing the network with and without the inclusion of user feedback is made in the next section. This kind of study, where a component of the solution is removed and its performance is tested, is called *ablation study* [1]. It enables to understand the contribution of a component to the overall system.

5.1.1 Metrics for evaluating the network

Several metrics can be used to evaluate recommendation systems [19]. The characteristics of these specific ML models require that different metrics are used in order to understand how well did the model satisfy the user's query intent [18]. In this case, the model is evaluated for how relevant are the suggestions that are given to the technician while he
she is filling a WO. The model is evaluated based on the rankings attributed to each of the suggestions and whether the word(s) presented match the ones present in the sequence of WOs used in the testing methodology.

A first analysis was conducted to determine which metric would be most appropriate to evaluate the BN's ability to suggest words. The Mean Average Precision (MAP) metric was chosen because it takes into account the rank of suggestions in its calculation,

which is exactly what is intended here. The Mean Average Precision rewards lists of suggestions with a lot of "correct" (relevant) recommendations, and rewards those at the top with the most likely correct recommendations. The problem with this metric is that there can be several right answers (suggestions), so the final value for this metric is given by the combination of how many correct suggestions there were, having into account in what rankings as well. For the problem at hand, only one suggestion is considered to be right, due to the fact that there can only be one suggestion chosen to form a WO 's description. This leads to labelling the MAP metric as inadequate, since if it was considered with only one possible correct answer, its value would be very low.

Once it is understood that the chosen metric can only consider one right answer from the set of suggestions, the chosen one was the Mean Reciprocal Rank (MRR) [40]. It consists in a "*statistic measure for evaluating any process that produces a list of possible responses to a sample of queries, ordered by probability of correctness. The reciprocal rank of a query response is the multiplicative inverse of the rank of the first correct answer: 1 for first place,* $1/2$ *for second place,* $1/3$ *for third place and so on*"[40]. So, if a technician is presented with a list of suggestions and chooses the third one, for example, the MRR would be $1/3$. If none of the suggestions is what the technician intended and no suggestion is chosen, then the MRR is considered to be zero. In this case, the metric is appropriate since it is calculated based on ranking the suggestions, and it only considers one correct answer. This metric alone, however, can be insufficient to understand the network's performance. The MRR can be high due to the fact that there are suggestions where the technician picked the first rank suggestions, but it is also important to analyse the number of times that none of the suggestions were chosen, that is, the number of times the network produced a list of inadequate results and none of them was considered. Therefore, a second metric is considered alongside the MRR, which is the *False Negative Rate*, also known as *Miss Rate* [41]. This metric is calculated by how many times the network presented a list of suggestions where no suggestion is the right answer, divided by the total number of suggestions that were made.

5.2 Experimental Results

The experimental results consist in the application of the presented test methodology and the chosen metrics in order to evaluate the BN in carefully chosen scenarios. There were considered three subfamilies with distinct characteristics — *Chillers*, *Air Treatment Units* and *General Equipments*. The *Chillers* subfamily has the characteristic of dealing only with assets of the Chiller type, however it registers a low quantity of data, only

Chillers				
train / test	*MRR*	*Total Suggestions*	*Missed Suggestions*	*Miss Rate*
70 / 30	0.5153	167	73	0.4300
80 / 20	**0.5666**	94	35	**0.3700**
50 / 50	0.4636	278	123	0.4400

Table 5.1: Results obtained for the Chillers subfamily

one hundred and sixty eight (168) WOs. The *Air Treatment Units* represents the assets that are responsible for air treatment and holds a total of two thousand eight hundred and seventy one (2871) WOs. The *General Equipments* subfamily is slightly different from the previous two, since it represents a broad spectrum of assets known as general equipments; the WOs present in this subfamily are very distinct from each other and the malfunctions/situations that are found are very specific. It holds a total of eight hundred and eighty six (886) WOs.

The procedure consisted in, for each subfamily, hold a subset for testing, and train with the rest of the WOs. There were chosen three possible partitions for training/test percentages - 70/30%, 80/20% and 50/50%. The considered metrics were the ones approached in the previous subsection, MRR and Miss Rate, with information about the total number of suggestions that were made in total and how many of them did not have any relevant suggestion (Miss Rate). Tables 5.1, 5.4 and 5.5 present the obtained results.

From the observation of these Tables, it is possible to discuss the obtained results. Starting with the Chillers subfamily in Table 5.1, it is noticeable that the *Miss Rate* is quite high for the three combinations of training and testing. This means that the network had difficulty in understanding what were the most adequate suggestions to the user, failing to include relevant suggestions about 41% of the time (Miss Rate mean). Looking at the MRR, it is about 50% for the three cases. Remembering that the MRR is calculated by doing the multiplicative inverse of the rank of the first correct answer, that is, 1 for first place, $1/2$ for second place, $1/3$ for third place and so on, a MRR of $1/2$ means that when there is a correct suggestion from the list presented, it usually sits on the top ranks. Therefore, the conclusion is that the network missed a big part of the suggestions that it should make, but when it presented a list with a correct option, this correct option was one of the top suggestions, which is the intended — correct answers ranked first indicates a good functioning of the network. It is also noticeable that the best combination of train and test percentages was 80/20, which allows to understand that the more data the network has to train, the better its suggestions will be.

Chiller with Cross-validation				
train / test	*MRR*	*Total Suggestions*	*Missed Suggestions*	*Miss Rate*
8-fold CV	0.5470	518	176	0.3400
10-fold CV	0.5540	518	182	0.3400
12-fold CV	0.5570	518	173	**0.3300**
80 / 20	**0.5666**	94	35	0.3700

Table 5.2: Results obtained for the Chillers subfamily with Cross-validation

Since this Chillers subfamily has the characteristic of presenting low amounts of data, the technique of ten-fold Cross-validation (CV) [14] was conducted for better network evaluation. Table 5.2 presents the obtained results, where it is possible to conclude that with a bigger k in the k-fold CV process it is possible to obtain better results, but by a very little margin. As a result of applying the data from the subfamily chillers in the network, the conclusions that are drawn are that there are low quantities of data to generate an adequate model, so other techniques might be necessary such as synthetic generation of data, with the help of domain experts. This subfamily is representative of all of the other subfamilies that present low quantities of data, so they are good candidates to follow the same process of synthetic data generation in order to create adequate models.

Another consideration that is important to have in mind is the inclusion of domain knowledge into the network, in order to obtain more accurate results. Since the network is trained with historic WOs that pass through the created NLP pipeline, there can be the consideration of stopwords that are specific to a given subfamily. The words that are considered not to be included in the final sentences are given by a stopwords file that can be altered, as described in chapter three. The specification of a subfamily-specific stopwords file, with information about the words that appear in the subfamily that do not present significant information, will allow the network to achieve better results. The idea is that a domain expert can point out the words that do not present significant information, therefore domain knowledge that allows to improve the network's adequacy to a given subfamily is included. From a series of conversations with domain experts it was possible to generate a stopwords file, specific to the Chillers subfamily. The results are presented in Table 5.3, where it was compared the best example without subfamily specific stopwords (80/20) with the same example, using subfamily specific stopwords. The conclusions are that the inclusion of domain knowledge allows for more adequate results in terms of the MRR metric, however the Miss Rate remains the same due to the amount of data present in this subfamily is low. For

Chiller with Specific Stopwords				
with / without CH stopwords	*MRR*	*Total Suggestions*	*Missed Suggestions*	*Miss Rate*
80/20 with	**0.7083**	27	10	0.3700
80/20 without	0.5666	94	35	0.3700

Table 5.3: Results obtained for the Chillers subfamily with the application of a specific stopwords file

the same reason, that the amount of data is low, the conclusions cannot be taken as absolute. Further investigations would be needed, with bigger quantities of data in different subfamilies, however it is difficult to ask a domain expert to catalogue all of the unimportant words to a given subfamily, since it is a systematic task of considerable proportions. The main idea here is that the inclusion of domain knowledge in the model surely makes it more adequate.

The second subfamily in which the network is tested is "Unidades de Tratamento de Ar" (Air Treatment Units). It was tested in the same conditions as Table 5.1, without any CV or inclusion of specific stopwords. The results can be observed in Table 5.4. The first thing that is noticeable is the amount of suggestions that were evaluated, a much higher number than in the Chillers subfamily. This subfamily had a considerable amount of data, 2871 WOs. The MRR is about the same as the Chillers subfamily, 50%, which allows to conclude that when there are correct answers in the list of suggestions they sit on the top ranks. However, there is a noticeable difference, which is the Miss Rate. It is a considerably lower value when compared to the chiller's 37%, with about 27% miss rate. This allows to conclude that with a considerable amount of data the network can be properly trained, therefore the results are more adequate. This UTA's subfamily has a larger amount of data than the Chillers subfamily, which allowed to conduct training and testing in better conditions. The best proportions of training and testing are of 70/30. The subfamilies with the same characteristics as the UTA's are considered the most appropriate for training the network, since they present patterns that can be discovered and a considerable amount of data. The same does not occur with the subfamilies similar to "Equipamentos Gerais" (General Equipments).

The third subfamily in which the network was evaluated was the "Equipamentos Gerais". It was tested in the same conditions as Table 5.1 and 5.4, without any CV

UTA's				
train / test	*MRR*	*Total Suggestions*	*Missed Suggestions*	*Miss Rate*
70 / 30	**0.5700**	3548	928	**0.2600**
80 / 20	0.5070	2600	743	0.2800
50 / 50	0.5550	4762	1267	0.2600

Table 5.4: Results obtained for the UTA's subfamily

Equipamentos Gerais				
train / test	*MRR*	*Total Suggestions*	*Missed Suggestions*	*Miss Rate*
70 / 30	**0.2787**	750	481	**0.6400**
80 / 20	0.2751	491	317	0.6400
50 / 50	0.2610	1253	830	0.6600

Table 5.5: Results obtained for the "Equipamentos Gerais" subfamily

or inclusion of specific stopwords. The results can be observed in Table 5.5. While observing it, the first thing that is noticeable is the low score on the MRR metric, about 27%. If the focus is shifted to the Miss Rate, the score is not adequate either, about 65% of missed suggestions. These results are poor, however, the explanation for them is the fact that there is a high variance in the WOs that this subfamily encompasses. The name "Equipamentos Gerais" (General Equipment's) suggests just that, there are a lot of possible WOs in a considerable variety of different assets. The words that are used are not restricted to describe a given kind of assets such as in the Chillers or the UTA's subfamily; there is a high diversity in the words found in the WOs and the assets that are considered, so the network does not have the capacity to identify patterns that will lead to adequate suggestions. Having said that, this finding results in the need of having a different approach with this kind of subfamilies that encompass a broad spectrum of asset management situations and assets. As future work, it would be interesting to study how to discover the patterns present in these broad-spectrum subfamilies, probably by drilling-down the hierarchy and study the WOs by asset instead of by subfamily.

5.2.1 Results Regarding Implicit User Feedback

As described in Chapter 4, the developed BN has the capability of adapting to the present needs of the technicians by including user feedback when a given suggestion is selected from the list. In this model evaluation chapter, it is also important to evaluate how well does the mechanism of implicit user feedback work. Therefore, an ablation study [1] is conducted, where the most adequate models in each subfamily were taken and the experiment of testing the models with and without implicit user feedback is made. The results can be observed in Table 5.6.

The conclusions drawn are unanimous, and the observation of the table allows to understand that the network performs more adequately when the mechanism of implicit user feedback is on. The MRR for all of the subfamilies was higher when the mechanism was on, which indicates that the correct suggestions are found in higher rankings of the list. The Miss Rate is lower for cases where the mechanism is on, and there is a noticeable loss of ability to suggest lists with a correct answer when the mechanism is off — in the Chiller subfamily it went from 36% to 56%, for example. The switching of the mechanism of implicit user feedback to off implies that the only knowledge the network has is the one obtained when the historical WOs file was presented. The probability distributions in each state are modelled according to the frequencies found in this historic WOs file. These probability distributions remain static across the lifetime of the network if the mechanism is off, proving unable to adapt to new sequences of WOs. If the mechanism is on, the probability distributions are altered at each selection of a suggestion, the technicians are presented with more relevant suggestions and the network keeps on learning.

One important consideration to have in mind is the evaluation of the reward function presented in the previous chapter, which provides to the implicit user feedback mechanism the notion of how much time has passed between the same asset management situation, and how should the suggestion that was chosen be rewarded. The values given for this reward function are initial values that appear merely out of domain assumptions, however, it is important that an empirical study on the most adequate values for the function is conducted. Figure 5.2 presents three different reward functions, coloured in blue, green and red. The motivation for this study is to understand what is the impact of the values given to the reward function by shifting it in the reward axis. Hence, by shifting the values of reward associated to each time period present in the X axis, it is possible to produce three functions and study the effect of the reward values in the overall solution. The models with most adequate performance indicators are chosen for each subfamily, that is, 80/20 (training and testing percentage) for

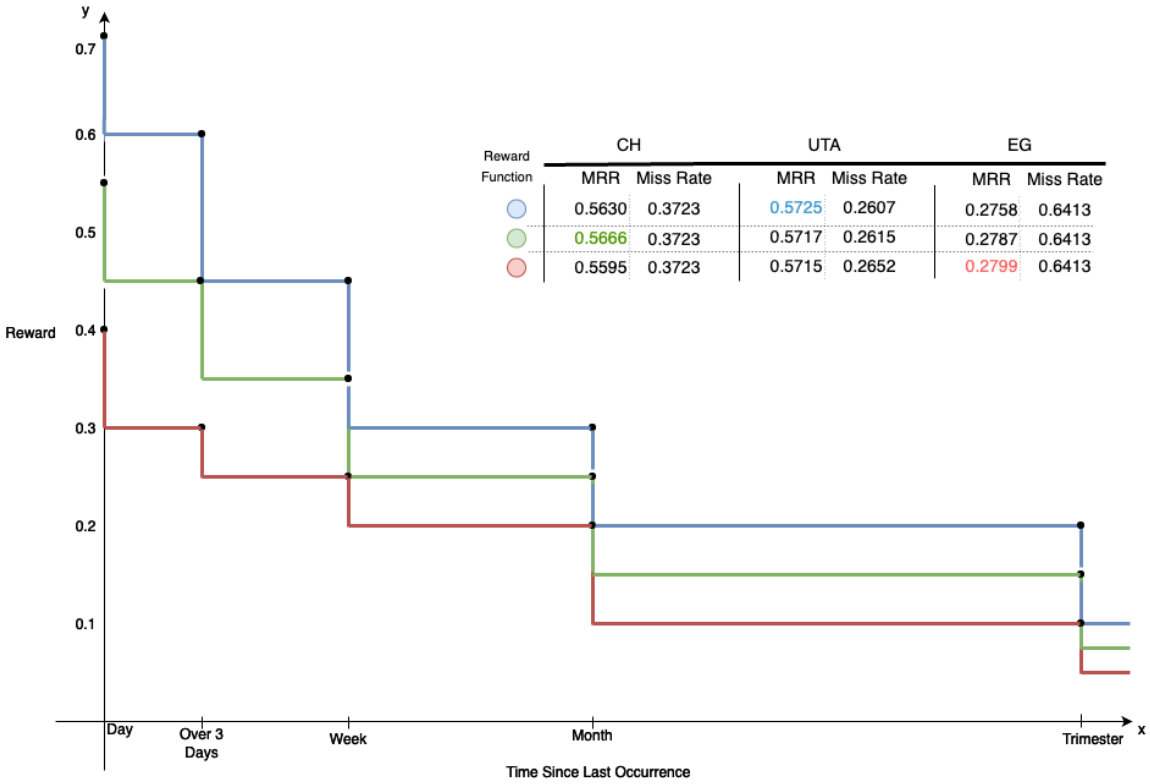
Implicit User Feedback					
Subfamily - Best Eval	*User Feedback*	*MRR*	*Total Suggestions*	*Missed Suggestions*	*Miss Rate*
CH 80/20	**Yes**	**0.5666**	94	35	**0.3700**
CH 80/20	No	0.3480	94	53	0.5600
UTA 70/30	**Yes**	**0.5700**	3548	928	**0.2600**
UTA 70/30	No	0.4576	3548	1307	0.3700
EG 70/30	**Yes**	**0.2787**	750	481	**0.6400**
EG 70/30	No	0.2700	750	513	0.6800

Table 5.6: Results obtained for evaluation of the user implicit feedback mechanism

Chiller (CH) and 70/30 for Air Treatment Units (UTA) and General Equipments (EG). The effect produced by the three functions can be observed in the table present in Figure 5.2. It is concluded that the performance indicators do not change significantly when using different functions with different reward values in the given reward range (0.7 to 0.05). The function with best results is different for each of the three models tested. The explanation for that might be that each subfamily has a specific scheduling of WOs in its preventive plans and the time that spans between these WOs is more akin to a given reward function. This affinity to a given reward function could be due to the fact that it has the highest rewards in the most frequent periods of time verified in the maintenance operations of the given subfamily.

5.3 Prototype

The network evaluation described in the previous section intended to assess the network's performance by simulating the technician's day-to-day operations over a broad period of time. Furthermore, it is also important to evaluate the network from a daily scope of utilisation, which can be done through the creation of a prototype. The motive



Reward Function	CH		UTA		EG	
	MRR	Miss Rate	MRR	Miss Rate	MRR	Miss Rate
	0.5630	0.3723	0.5725	0.2607	0.2758	0.6413
	0.5666	0.3723	0.5717	0.2615	0.2787	0.6413
	0.5595	0.3723	0.5715	0.2652	0.2799	0.6413

Figure 5.2: Study of reward function values with three distinct functions

behind this prototype is to simulate the tablet that the technicians use and how the BN model would be included in the operation of creating a WO. Figure 5.3 presents the initial screen of the prototype, where there is a text box on which the technician can input the WO's description — field "Descrição". All of the other fields are static due to the fact that the main purpose of the prototype, at the moment, is to evaluate the BN. As future work, it would be interesting to develop it to its full capabilities. This prototype is developed using the *Kivy Python Framework* [32].

The idea is that the technician can input the words to describe the WO and suggestions appear to him as he
she presses the space button. Knowledge from the domain allows to understand that the WOs should be short and simple, with irrelevant information removed. However, technicians tend to write using definite and indefinite articles, speech connectors and pronouns, which are normal to every day writing and speech. To help the technicians with the task of keeping the WOs short and simple, without unnecessary information, the network presents the suggestions with words that are not relevant removed. Figure 5.4 presents an example of a technician that intended to report a belt substitution

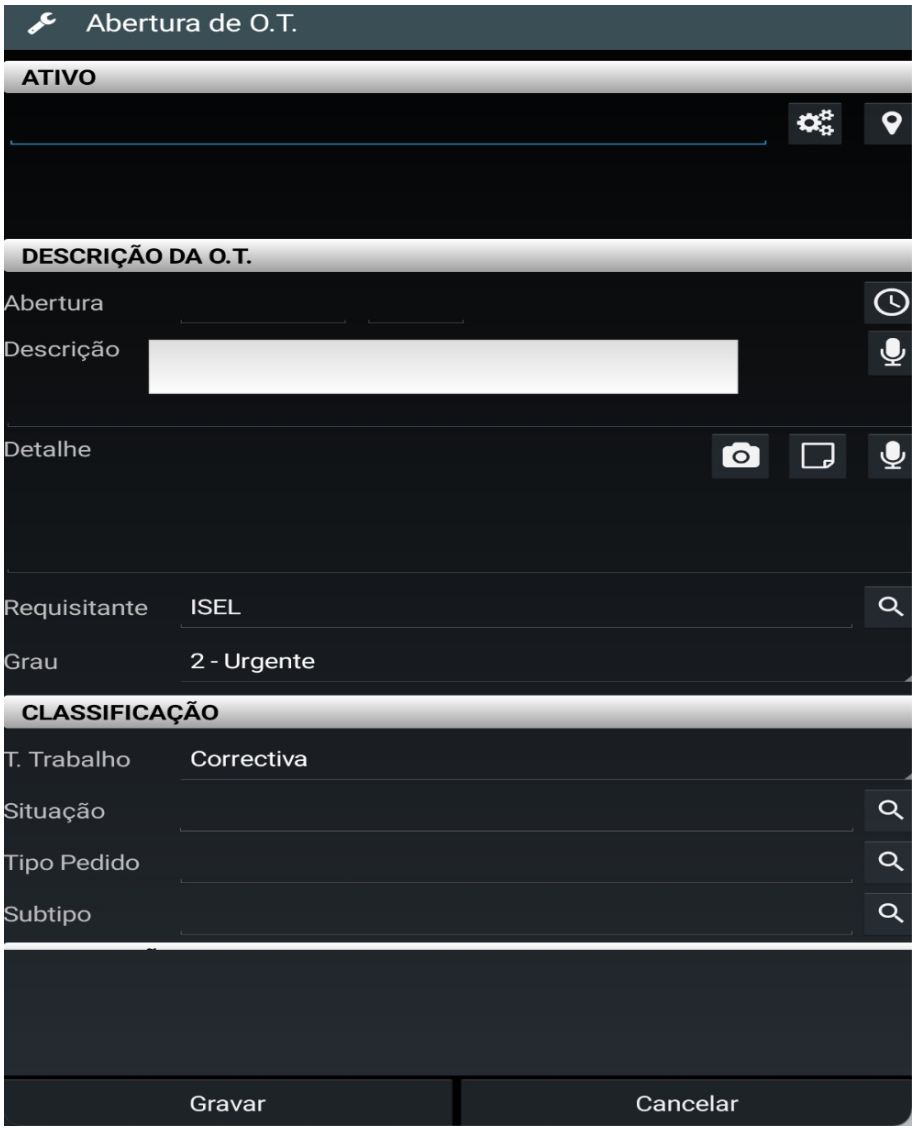


Figure 5.3: Initial Screen of the developed prototype

("substituição da correia"). It can be seen that the irrelevant word "da" is removed from the suggestions. The method that is used to generate the suggestions is the inference of the whole sentence, described in the previous chapter, which allows to suggest whole sentences to the technician. There is also the possibility to use the simpler method, inference of the next word, which would only show the next possible words based on the inputted ones.

The prototype allows the technician to write in his usual way, however it tries to influence the final description of the WO by suggesting sentences that meet the standards of the domain: short and simple. The recommendation system ignores words that do not present any relevant information to the WO by comparing each word written

Figure 5.4: Example of filling out a work order

by the technician with all the words it knows. Since it was trained on carefully processed WOs that do not include irrelevant words, when these terms appear they are not considered by the network, and discarded from the suggestions. Regarding the suggestions given, "spz", "xpz" and "spa" are kinds of belts, "uta" and "utan" refer to assets and the numbers are references to specific kinds of belts.

There is one important consideration that this prototype allows to demonstrate. The domain of asset management is broad and complex, so most of the WOs that are created have a high level of specificity. In order to understand the amount of complexity that a recommendation system in such a domain has to model, Figure 5.5 shows the suggestions given for a specific asset management situation.

From the sentence "limpeza do cilindro humidificador" (cleaning of the humidifying cylinder) the suggestions that are given have a high level of individuality. As seen

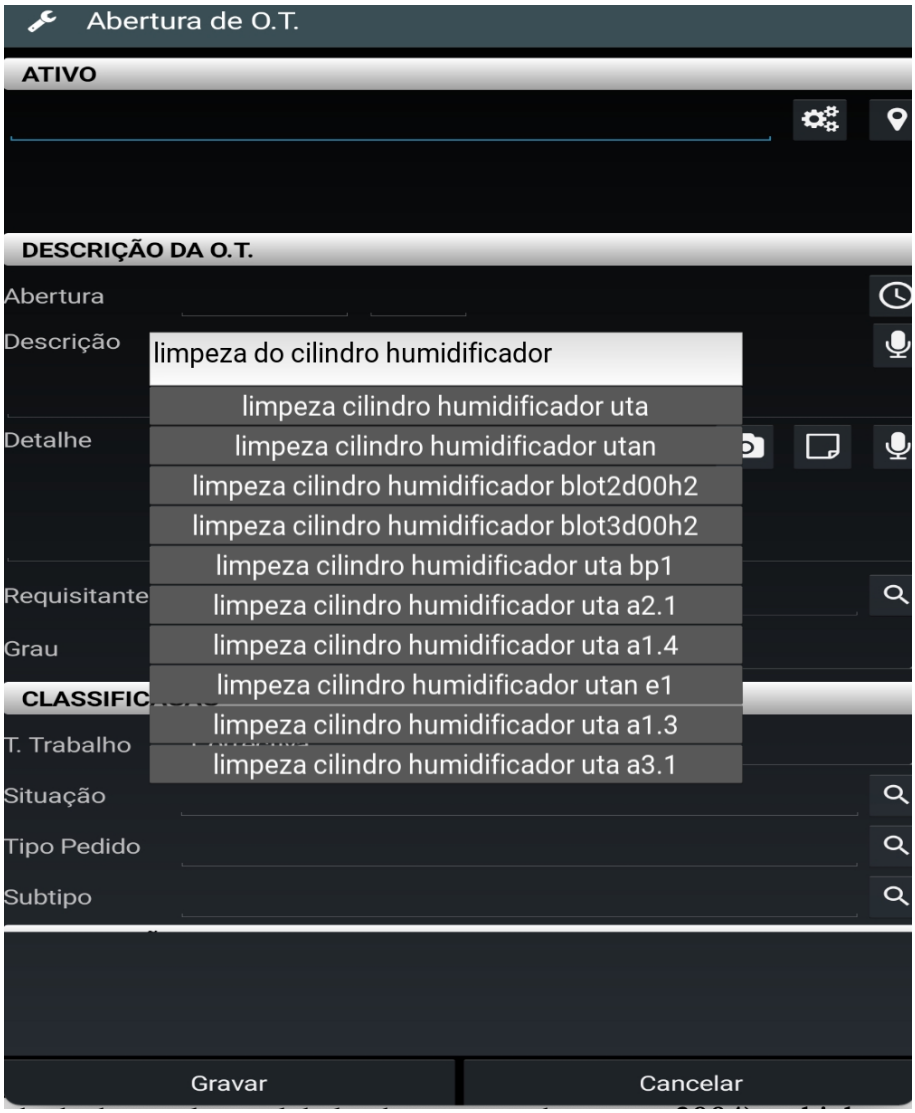


Figure 5.5: Example of how specific can the work orders be

before, "uta" and "utan" refer to the assets where this operation can be made. The particular sequences of numbers and letters in suggestions ranked in third and fourth place consist in references of different kinds of the component (the cylinder). The remaining suggestions are of the assets once again, but this time with individual identifiers attached. This allows to demonstrate that a recommendation system that suggests words cannot be perfect in terms of always recommending correctly what the user intends to write. However, the idea is that it influences the creation of the WOs so that they contain, for the most part, relevant information, and that the asset management situations can be written using the same terms.

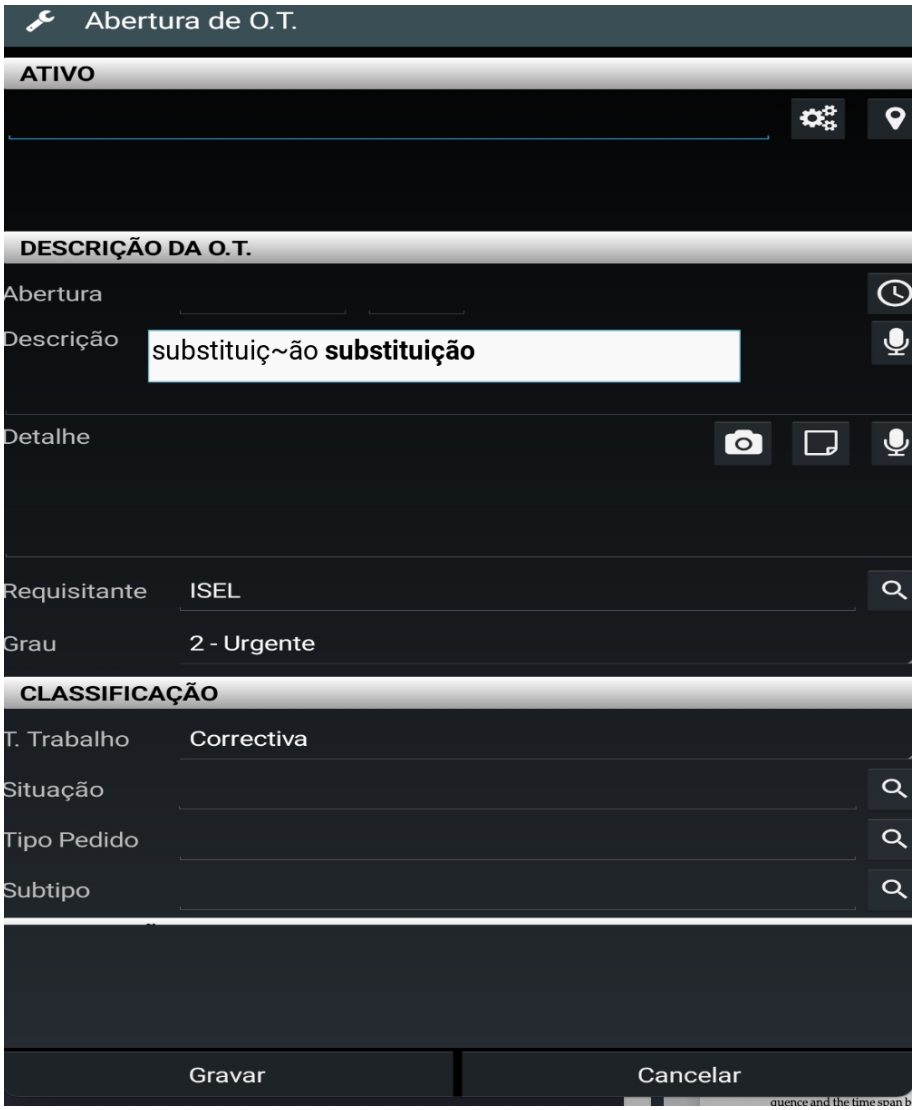


Figure 5.6: Example of the notice of a wrongfully spelled word

5.3.1 Spellchecker

During the exploratory analysis of the WOs conducted in Chapter 3, it was noticeable that there were words written incorrectly. These spelling errors, if verified in large quantities, lead to an imprecision in defining the asset management situation properly. Therefore, the implementation of a spellchecking algorithm that allows to correct the technician while he
she is filling the WO can have the effect of reducing the amount of spelling errors found. Of course, it is important to understand that the spellcheckers used in smartphones and tablets nowadays have industrial strength and are far more complex than the developed one. The idea is to take an already existent spellchecker, using the

SpellChecker Python Module [57], and adapt it to the specific domain of asset management. Since the words used are mainly technical, there isn't even the need to load full language dictionaries, where most of the words of a given language are loaded. In this case, the procedure was to create a new language dictionary that regards only words and sentence constructions present in WOs. It can be understood as a technical language dictionary, that is used by the spellchecker to confirm whether a word is well written or not. This spellchecker was included in the prototype, so that if words with spelling errors were inputted there was a notice to the technician. This way, before querying the network for suggestions, the spellchecker is run on the inputted words. Figure 5.6 presents an example of a technician that clicked on the accent key twice while writing "substituição" (substitution); happens to us all. The prototype promptly suggests the correction. The technician can accept the correction by pressing the enter key.

6

Conclusions and Future Work

This research intended to achieve the normalisation of the words that are used to describe asset management situations when creating WOs. This normalisation is achieved through the recommendation of words while a technician is filling a WO.

A first phase of exploration allowed to understand the domain and how the words are used by the technicians. After studying and applying several Natural Language Processing techniques it was possible to conclude that, globally, the words are used in a very disperse manner. Moreover, this finding indicates that there exist few words, used in a considerable amount of different asset management situations, which leads to a difficulty in identifying patterns in the WOs. In other words, there are few terms, and these few terms are used to describe all the possible situations in the domain. This dissemination of the words across all of the domain makes it difficult for a single ML model to discover the patterns present in the data. The chosen option is, therefore, to use a ML model per subfamily.

The chosen algorithm to implement the recommendation system consists in a BN with implicit user feedback. Each of the network's states models the position of a given word in the sentence, that is, the first state models the first word, second state the second word, and so on. Each of the states encompasses a probability distribution from which the network has the ability to suggest words. It is from the updating of these probability distributions that the adaptation of the network occurs. Asset management is a volatile domain, different WOs are continuously being generated, so the recommendation system adapts to the needs of the technicians through a mechanism of

implicit feedback. This mechanism of implicit feedback uses a reward function which decays over time, so it is given priority to suggestions (asset management situations) that have happened more recently for the last time. The objective of this BN is to influence the technicians while the WO is being created, so that the words that are used are the ones suggested by the network and, consequently, the ones used in the majority of the cases, leading to the normalisation. The mechanism of implicit feedback is to prevent that the suggestions of the network stop being relevant, due to the volatility of the asset management domain.

The evaluation of the BN on three different subfamilies presented slightly distinct outcomes, representative of what happens in other subfamilies, according to the characteristics of each. The Chillers subfamily presented a low quantity of data, so the network had difficulty in conducting appropriate training. As future work, other techniques should be explored, such as synthetic generation of data, with the help of domain experts, so that the amount of data is increased. For the *"Unidades de Tratamento de Ar"* (Air Treatment Units) subfamily, the results were adequate, since this subfamily has a considerable amount of data and the network was able to identify the patterns present in it. For the *"Equipamentos Gerais"* (General Equipment) subfamily, the network produced inaccurate results due to the fact that there exists a high variance in the WOs that this subfamily encompasses. There is a high diversity in the words found in the WOs and the assets that are considered, so the network does not have the capacity to identify patterns that will lead to adequate suggestions. As future work, it would be interesting to study how can the patterns present in these broad-spectrum subfamilies be discovered. One possibility is through drilling-down the hierarchy and study the WOs by asset instead of by subfamily.

The mechanism of implicit user feedback is also evaluated through an ablation study [1]. The conclusion was that the inclusion of the user's feedback improved the network's suggestions considerably. The only registered drawback regarding this mechanism is the fact that the values given for the reward function appear merely out of domain assumptions. However, there is limited literature regarding this aspect, so the recommended way to develop it is through empirical studying. Consequently, a study which aims at understanding the effect of the reward values in the overall solution is conducted, where three different reward functions are used to test different models and their performance indicators are observed. The conclusion is that the performance indicators do not present significant changes when using each of the different reward functions.

Moreover, the implementation of the BN allowed for the creation of a prototype. Its intent is to simulate the tablet that the technicians use and how the BN model would

be included in the operation of creating a WO. In order for the objective of this thesis to be fulfilled without any doubt, there is the need to install the prototype in the tablets that the technicians use. This is the best way to evaluate the network, its capacity of suggestion in real-life scenarios and, more importantly, if the suggestions that are made on a day-to-day basis make the lexicon that is chosen converge towards a normalisation of the words. Since it is not possible to conduct this kind of evaluation due to a lack of useful time, the network is evaluated using "unseen" sequences of WOs and assessed through specific metrics that report its adequacy.

The consideration of using a single ML model per subfamily indicates that there is the need to create and train as many BNs as subfamilies — one hundred and eight (108). This requires that there is an infrastructure that can manage all of these models and their life-cycles. Since the tablets used do not have internet connectivity at all times, due to the fact that some asset management situations occur in places below ground for example, the models have to be stored in the tablets and managed when connectivity is restored. This idea is for future work, but there would be a central entity that would operate by constantly monitoring metrics that evaluate how well a network is suggesting. These metrics are communicated every time there is connectivity, and if the metrics go below a certain threshold, the model is recycled and created with newer WOs. This would allow for the probability distributions to be created from scratch, considering terms used more recently.

The next step in this research, as future work, would be to test the entirety of the one hundred and eight (108) subfamilies one by one, observe the results and understand the next steps that can be done to improve the subfamilies in which the results were not the most adequate — whether because they encompass too many distinct assets or the network could not identify patterns. To conclude, the implementation of a word recommendation system with the objective of normalising the terms used to create a WO was successful. The code written in virtue of all of the work developed in this thesis can be consulted in `https://github.com/PedrcSantos/MasterThesis`, alongside with the recommendation system prototype.

References

[1] Richard Meyes, Melanie Lu, Constantin Waubert de Puiseau, and Tobias Meisen, "Ablation studies in artificial neural networks", *arXiv preprint arXiv:1901.08644*, 2019.

[2] United States Department of Transportation, "Asset management primer", 1999.

[3] Kevin B Korb and Ann E Nicholson, *Bayesian artificial intelligence*. CRC press, 2010.

[4] Fabiano M Belém, Jussara M Almeida, and Marcos A Gonçalves, "A survey on tag recommendation methods", *Journal of the Association for Information Science and Technology*, vol. 68, no. 4, pages 830–844, 2017.

[5] Robbert Anton Kivits and Craig Furneaux, "BIM: Enabling sustainability and asset management through knowledge management", *The Scientific World Journal*, vol. 2013, 2013.

[6] A. K. Parlikad and M. Jafari, "'challenges in infrastructure asset management"", *IFAC*, vol. 2016, 2016.

[7] Wikipedia contributors, *Chiller — Wikipedia, the free encyclopedia*, [Online; accessed 26-November-2021], 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Chiller&oldid=1042126772.

[8] IBM, *What is a computerized maintenance management system?*, [Online; accessed 7-December-2021], 2021. [Online]. Available: https://www.ibm.com/topics/what-is-a-cmms.

[9] Snehal Bhoir, Tushar Ghorpade, and Vanita Mane, "Comparative analysis of different word embedding models", in *2017 International conference on advances in computing, communication and Control (ICAC3)*, IEEE, 2017, pages 1–4.

[10] Yanshan Wang, Sijia Liu, Naveed Afzal, Majid Rastegar-Mojarad, Liwei Wang, Feichen Shen, Paul Kingsbury, and Hongfang Liu, "A comparison of word embeddings for the biomedical natural language processing", *Journal of biomedical informatics*, vol. 87, pages 12–20, 2018.

[11] stevens, *What is civil construction?*, [Online; accessed 28-November-2021], 2021. [Online]. Available: https://www.stevensec.com/blog/what-is-civil-construction.

[12] Wikipedia contributors, *Convolutional neural network — Wikipedia, the free encyclopedia*, [Online; accessed 2-February-2022], 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Convolutional_neural_network&oldid=1068636789.

[13] Fernando Martínez-Plumed, Lidia Contreras-Ochando, Cesar Ferri, José Hernández-Orallo, Meelis Kull, Nicolas Lachiche, María José Ramírez-Quintana, and Peter Flach, "Crisp-dm twenty years later: From data mining processes to data science trajectories", *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 8, pages 3048–3061, 2019.

[14] Payam Refaeilzadeh, Lei Tang, and Huan Liu, "Cross-validation.", *Encyclopedia of database systems*, vol. 5, pages 532–538, 2009.

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding", *arXiv preprint arXiv:1810.04805*, 2018.

[16] Edsger W Dijkstra, "Recursive programming", *Numerische Mathematik*, vol. 2, no. 1, pages 312–318, 1960.

[17] Zhuoye Ding, Xipeng Qiu, Qi Zhang, and Xuanjing Huang, "Learning topical translation model for microblog hashtag suggestion", in *Twenty-third international joint conference on artificial intelligence*, 2013.

[18] Wikipedia contributors, *Evaluation measures (information retrieval) — Wikipedia, the free encyclopedia*, [Online; accessed 26-September-2022], 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Evaluation_measures_(information_retrieval)&oldid=1105990948.

[19] Guy Shani and Asela Gunawardana, "Evaluating recommendation systems", in *Recommender systems handbook*, Springer, 2011, pages 257–297.

[20] Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims, "Evaluation methods for unsupervised word embeddings", in *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015, pages 298–307.

[21] IBM, *Defining facilities management*, [Online; accessed 28-November-2021], 2021. [Online]. Available: https://www.ibm.com/topics/facilities-management.

[22] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov, "Enriching word vectors with subword information", *Transactions of the Association for Computational Linguistics*, vol. 5, pages 135–146, 2017, ISSN: 2307-387X.

[23] Piotr Bojanowski, Onur Celebi, Tomas Mikolov, Edouard Grave, and Armand Joulin, "Updating pre-trained word vectors and text classifiers using monolingual alignment", *arXiv preprint arXiv:1910.06241*, 2019.

[24] Jeffrey Pennington, Richard Socher, and Christopher D Manning, "Glove: Global vectors for word representation", in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pages 1532–1543.

[25] Fréderic Godin, Viktor Slavkovikj, Wesley De Neve, Benjamin Schrauwen, and Rik Van de Walle, "Using topic models for twitter hashtag recommendation", in *Proceedings of the 22nd international conference on World Wide Web*, 2013, pages 593–596.

[26] Henrique X Goulart, Mauro DL Tosi, Daniel Soares Gonçalves, Rodrigo F Maia, and Guilherme A Wachs-Lopes, "Hybrid model for word prediction using naive bayes and latent information", *arXiv preprint arXiv:1803.00985*, 2018.

[27] Rémi Lebret and Ronan Collobert, "Word emdeddings through hellinger PCA", *arXiv preprint arXiv:1312.5542*, 2013.

[28] Sean R Eddy, "Hidden markov models", *Current opinion in structural biology*, vol. 6, no. 3, pages 361–365, 1996.

[29] Xia Hu, Lingyang Chu, Jian Pei, Weiqing Liu, and Jiang Bian, "Model complexity of deep learning: A survey", *Knowledge and Information Systems*, vol. 63, no. 10, pages 2585–2619, 2021.

[30] Heiner Lasi et. al., "Industry 4.0", *Business and Information Systems Engineering*, 2014.

[31] Songrit Maneewongvatana and David M Mount, "Analysis of approximate nearest neighbor searching with clustered point sets", *arXiv preprint cs/9901013*, 1999.

[32] *Kivy framework*, https://kivy.org/.

[33] Ralf Krestel, Peter Fankhauser, and Wolfgang Nejdl, "Latent dirichlet allocation for tag recommendation", in *Proceedings of the third ACM conference on Recommender systems*, 2009, pages 61–68.

[34] Wikipedia contributors, *Latent dirichlet allocation — Wikipedia, the free encyclopedia*, [Online; accessed 14-March-2022], 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Latent_Dirichlet_allocation&oldid=1077006447.

[35] Kai Lei, Qiuai Fu, Min Yang, and Yuzhi Liang, "Tag recommendation by text classification with attention-based capsule network", *Neurocomputing*, vol. 391, pages 65–73, 2020.

[36] Yang Li and Tao Yang, "Word embedding for understanding natural language: A survey", in *Guide to big data applications*, Springer, 2018, pages 83–104.

[37] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov, "Roberta: A robustly optimized bert pretraining approach", *arXiv preprint arXiv:1907.11692*, 2019.

[38] Jason Brownlee, *A gentle introduction to long short-term memory networks by the experts*, 2021. [Online]. Available: https://machinelearningmastery.com/gentle-introduction-long-short-term-memory-networks-experts/.

[39] Qiuchen Lu, Xiang Xie, Ajith Kumar Parlikad, Jennifer Mary Schooling, and Eirini Konstantinou, "Moving from building information models to digital twins for operation and maintenance", *Proceedings of the Institution of Civil Engineers-Smart Infrastructure and Construction*, vol. 174, no. 2, pages 46–56, 2020.

[40] Wikipedia contributors, *Mean reciprocal rank — Wikipedia, the free encyclopedia*, [Online; accessed 26-September-2022], 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Mean_reciprocal_rank&oldid=1107032139.

[41] ——, *Precision and recall — Wikipedia, the free encyclopedia*, [Online; accessed 28-September-2022], 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Precision_and_recall&oldid=1112236752.

[42] Gobinda G. Chowdhury, "Natural language processing", *Annual Review of Information Science and Technology*, vol. 37, no. 1, 51–89, 2005. DOI: 10.1002/aris.1440370103.

[43] Christer Stenström, Mustafa Aljumaili, and Aditya Parida, "Natural language processing of maintenance records data", *International Journal of Condition Monitoring and Diagnostic Engineering Management*, 2015.

[44] Yassine Bouabdallaoui et. al., "Natural language processing model for managing maintenance requests in buildings", *Buildings*, vol. 10, no. 9, 2020.

[45] Ian T Jolliffe and Jorge Cadima, "Principal component analysis: A review and recent developments", *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, page 20 150 202, 2016.

[46] Wikipedia contributors, *Part-of-speech tagging — Wikipedia, the free encyclopedia*, [Online; accessed 9-December-2021], 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Part-of-speech_tagging&oldid=1057308629.

[47] *Pretrained word embeddings: Word embedding nlp*, 2020. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/03/pretrained-word-embeddings-nlp/.

[48] Nathan Hartmann, Erick Fonseca, Christopher Shulby, Marcos Treviso, Jessica Rodrigues, and Sandra Aluisio, "Portuguese word embeddings: Evaluating on word analogies and natural language tasks", *arXiv preprint arXiv:1708.06025*, 2017.

[49] *Pub med central*, https://www.ncbi.nlm.nih.gov/pmc/.

[50] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, *et al.*, "Improving language understanding by generative pre-training", 2018.

[51] Juan Ramos *et al.*, "Using TF-IDF to determine word relevance in document queries", in *Proceedings of the first instructional conference on machine learning*, New Jersey, USA, vol. 242, 2003, pages 29–48.

[52] Pedro Santos, Nuno Datia, Matilde Pato, and José Sobral, "Comparing word embeddings through visualisation", *IV2022, 26th International Conference Information Visualisation*, 2022.

[53] Pedro Santos, Matilde Pato, Nuno Datia, and José Sobral, "Suggesting words using a Bayesian Network", *Inforum-Simpósio de Informática*, pages 1–12, 2022.

[54] Will Kenton, *Silo mentality*, [Online; accessed 1-December-2021], 2020. [Online]. Available: https://www.investopedia.com/terms/s/silo-mentality.asp.

[55] Zoran Latinovic and Sharmila C. Chatterjee, *How AI is helping companies break silos*, 2019. [Online]. Available: https://sloanreview.mit.edu/article/how-ai-is-helping-companies-break-silos/.

[56] Yang Song, Lu Zhang, and C Lee Giles, "Automatic tag recommendation algorithms for social recommender systems", *ACM Transactions on the Web (TWEB)*, vol. 5, no. 1, pages 1–31, 2011.

[57] *Spellchecker framework*, https://norvig.com/spell-correct.html.

[58] *Stanza framework*, https://stanfordnlp.github.io/stanza/.

[59] Jashanjot Kaur and Preetpal Kaur Buttar, "A systematic review on stopword removal algorithms", *International Journal on Future Revolution in Computer Science and Communication Engineering*, vol. 4, no. 4, 2018.

[60] Michael P. Brundage et. al., "Technical language processing: Unlocking maintenance knowledge", *Manufacturing Letters*, 2020.

[61] *What is tokenization: Tokenization in NLP*, [Online; accessed 9-December-2021], 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/.

[62] Ike Vayansky and Sathish AP Kumar, "A review of topic modeling methods", *Information Systems*, vol. 94, page 101 582, 2020.

[63] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He, "A comprehensive survey on transfer learning", *Proceedings of the IEEE*, vol. 109, no. 1, pages 43–76, 2020.

[64] Laurens van der Maaten and Geoffrey Hinton, "Visualizing data using t-SNE", *Journal of Machine Learning Research*, vol. 9, pages 2579–2605, 2008. [Online]. Available: http://www.jmlr.org/papers/v9/vandermaaten08a.html.

[65] D.J. Vanier, "Asset management A to Z", *Innovations in Urban Infrastructure*, pages 1–16, 2001.

[66] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, "Attention is all you need", *Advances in neural information processing systems*, vol. 30, 2017.

[67] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, "Efficient estimation of word representations in vector space", *arXiv preprint arXiv:1301.3781*, 2013.

[68] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang, "A review of recurrent neural networks: LSTM cells and network architectures", *Neural computation*, vol. 31, no. 7, pages 1235–1270, 2019.