Final Degree Project

Biomedical Engineering Degree

KeyCARE: a framework for biomedical Keyword Extraction, term Categorization, and semantic Relation

Barcelona, 5th June 2024

Author: Sergi Marsol Torrent

Directors: Luis Gascó, Martin Krallinger

Tutor: Juan Ignacio Barrios

Abstract

The medical sector generates vast amounts of unstructured data, which, if processed correctly, can significantly enhance medical processes and their outcomes. This thesis presents the development of KeyCARE, a Python library for keyword extraction, term categorization, and relations extraction that tackles this need. Utilizing mainly unsupervised and few-shot methods, KeyCARE efficiently extracts classified keywords from medical records with a recall of up to 98% and an f-score of up to 61%, with partial overlaps considered as correct. While these scores are not comparable to those of supervised Named Entity Recognition systems, they set a high standard for an unsupervised alternative in scenarios of data scarcity. Moreover, the library incorporates relation extractors that identify hierarchical relationships among biomedical keywords and with terminologies, achieving a precision and recall of 93%. This has a clear application in terminology enrichment, data generation and information extraction, particularly in specific domains and low-resource languages such as Catalan. This thesis encompasses the comprehensive development of KeyCARE, including an in-depth evaluation of the implemented models as well as basic use cases demonstrating its practical applications.

Acknowledgements

The successful completion of this bachelor thesis would not have been possible without the invaluable support and guidance of all those who have supported me during its completion.

First and foremost, I am deeply grateful to my thesis director, Dr. Luis Gascó. His expertise and insightful feedback throughout the research process were instrumental in shaping this thesis. His constant encouragement and constructive criticism were particularly valuable, pushing me to refine my work and achieve the highest standard.

I would also like to express my sincere gratitude to Dr. Martin Krallinger, leader of the Natural Language Processing for Biomedical Information Analysis (NLP4BIA) group at the Barcelona Supercomputing Center (BSC), where I conducted my research and continue to work in the present. Dr. Krallinger, along with the entire NLP4BIA team, provided invaluable assistance and advice during the project's development, fostering both my research and professional growth. In addition to that, I wanted to express my gratitude to the whole team for providing me the opportunity of participating in a variety of projects, including TeresIA, a nation-wide project in which I have actively participated with this thesis.

In third place, I would like to thank Dr. Juan Barrios, the tutor of my thesis. His guidance during its development has proven invaluable and his support has enabled me to rise to the occasion.

Finally, I extend my heartfelt gratitude to my friends and family for their unwavering support and encouragement, which has not only been essential for the development of this thesis, but also for my journey through the whole degree.

Glossary of abbreviations[1]

KeyCARE: Keyword Extraction, term Categorization and semantic Relation

AI: Artificial Intelligence

DL: Deep Learning

ML: Machine Learning

BSC: Barcelona Supercomputing Center

NLP4BIA: Natural Language Processing for Biomedical Information Analysis

SOTA: State-of-the-art

NLP: Natural Language Processing

SNOMED-CT: Systematized Nomenclature of Medicine – Clinical Terms

UMLS: Unified Medical Language Systems

DeCS: Health Sciences Descriptors ("Descriptores de Ciencias de la Salud")

IE: Information Extraction

NER: Named Entity Recognition

RE: Relation Extraction

LM: Language Model

PLM: Pre-trained Language Model

LLM: Large Language Model

PoS: Part-of-Speech

BERT: Bidirectional Encoder Representation from Transformers

SapBERT: Self-Alignment Pre-training Bidirectional Encoder Representation from Transformers

CRF: Convolutional Random Fields

CAGR: Compound Annual Growth Rate

TF-IDF: Term Frequency-Inverse Document Frequency

RAKE: Rapid Automatic Keyword Extraction

YAKE: Yet Another Keyword Extractor

SetFit: Sentence Transformers Fine-Tunning

Mesinesp: Medical Semantic Indexing in Spanish

PyPI: Python Package Index

[1] In addition to the glossary of abbreviations, a glossary of terms has been created and can be found in section 12.1. Glossary. The terms that are described there are marked in blue the first time they appear in the document.

List of figures

List of tables

List of equations

Table of contents

1. Introduction

This first introductory section presents the motivation and aim behind the development of KeyCARE: a framework for biomedical Keyword Extraction, term Categorization and semantic Relation. This first section introduces both the general and specific objectives that it wants to tackle, as well as the structure and followed methodology of the report and the limitations and scope of the project.

1.1. Motivation and aim of the project

From the beginning of the 21st century, the medical sector has witnessed a complete digital transformation that has led to a significant increase in the volume of medical data generated [1]. This data can be broadly categorized into two main formats: structured and unstructured. Structured data, typically found in organized tables and databases, provides well-defined and machine-readable information like patient records or genomic data. Unstructured data, on the other hand, encompasses the vast amount of information contained within regular text documents, ranging from patients' Electronic Health Records (EHRs) to research papers. Though less structured, this data holds immense value, capturing the richness of patient experiences and clinical observations that complement the analytical power of structured data. Being able to automatically extract information from these documents is the key to improve patient care, accelerate research, and generate valuable insights. This is why this project focuses on the processing of unstructured data from medical documents through Natural Language Processing (NLP) techniques. NLP is stablished as a research field in Artificial Intelligence (AI) that tackles the issue of textual data processing, which has a clear application in the healthcare sector.

Information Extraction (IE) is the NLP task that refers specifically to automatically extracting specific, relevant information from unstructured or semi-structured data sources [2]. However, IE systems rely on structured data resources like terminologies to be able to extract relevant information from unstructured data. A terminology is a controlled vocabulary that defines and categorizes specific concepts within a particular domain through a standard lexis and structure. In the biomedical field, several prominent terminologies exist, such as SNOMED-CT (Systematized Nomenclature of Medicine - Clinical Terms) [3], UMLS (Unified Medical Language System) [4], and DeCS (Health Sciences Descriptors, "Descriptores de Ciencias de la Salud") [5]. These terminologies provide standardized terms for diseases, procedures, medications, and other biomedical entities, ensuring consistency and facilitating data exchange across different healthcare systems. This also enables structuring the content of hospital medical records to provide a faster access to them and facilitate the economic management of this type of healthcare infrastructure through clinical coding.

Moreover, the main **Information Extraction (IE)** tools that are currently in use require manually annotated data, the generation of which is often very time consuming. In English, efforts have led to the generation of a wide variety of resources for the training and evaluation of NLP models, but that is not the case for other languages. While Spanish has more resources than most languages, it still lacks the creation of terminologies and corpora that would enable the application of some

NLP tools. In Catalan, like in many other low-resource languages, this situation is even more exaggerated, despite efforts from the Catalan government to produce a structured terminology in Catalan as SNOMED-CT [6].

An NLP tool for IE that requires amounts of manually annotated training data is **Named Entity Recognition (NER)** [2]. NER is a useful NLP method for information extraction and text analysis, based on the extraction of named entities from a text. A named entity is a key subject from that text that belongs to a specific semantic category or class, as are locations, names, or events. In the medical domain, some of the most relevant semantic categories in NER tasks are medical procedures, diseases, symptoms, species, and drugs. NER systems work in a supervised manner, thus requiring extensive training examples to learn from. And while there are studies on the performance of few-shot NER systems [7][8], they still present difficult scalability, especially in low-resource languages and multilingual scenarios.

Another key technique in information extraction is **Relation Extraction (RE)**, which refers to the classification of semantic relationships among entities [2]. The type of information provided through RE is essential to construct semantic knowledge bases as well as to enrich terminologies. Enriched terminologies enable a much swifter application of text processing tools in specific domains, such as the detection of interaction between biological agents as can be drugs and proteins [9]. However, the generation and enrichment of such structures is very time consuming, which has led to a scarcity of terminologies in a variety of languages. Therefore, there is a need for semiautomatic tools that use RE for terminology enrichment that could accelerate the process of creating a structured terminology.

This project aims to produce a tool that tackles these issues by retrieving relevant information from medical texts without requiring vast amounts of manually annotated training data. This encompasses the extraction of classified terms from texts as well as the extraction of semantic relations among those terms, all of which play a key role in information extraction and terminology enrichment, especially for terminologies of the medical domain. The main goal of this project is to implement this system using unsupervised and few-shot models, as well as supervised models for which there is accessible multilingual data. In addition to that, KeyCARE has been developed as part of wider projects and initiatives of the Barcelona Supercomputing Center (BSC), so its development has further goals and applications that are not included in the scope of the project.

1.2. Objectives

Recognizing the need to enhance the process harmonizing IE techniques for the medical sector, and especially in low-resource languages, this project aims to develop a framework for digital tools based on AI and NLP that stablishes a common access point for IE resources. This will enable the generation, expansion, reuse, and application of language technology resources with greater efficiency than current methods, and with many practical applications. Thus, this project introduces KeyCARE, a fully functional Python library that functions as a framework for information extraction from biomedical documents using models that do not require vast amounts of manually annotated data. The final objective of such frameworks is to provide the healthcare system with tools for the management of the vast amounts of data stored in medical documents, leading to an enhanced efficiency in healthcare management and to the creation of tools for assisted diagnosis, for instance.

The project has been structured according to the following specific objectives:

- Implementation of different unsupervised keyword extraction methods that achieve a recall comparable to that of supervised NER systems in biomedical-domain texts.
- Implementation of unsupervised or few-shot term categorizers that are able to classify medical terms into main semantic categories with a high precision.
- Implementation of a supervised term pairs classifier to extract precise "*is a*" relations among medical terms. In this case, the model is trained with automatically extracted data from previously existing medical terminologies.
- Creation of an easy-to-use Python library with integrated classes and functions, as well as the corresponding documentation and tutorials.
- Complete evaluation of each of the tasks performed by the library and of the library as a whole. While the library is not expected to perform information extraction tasks with an accuracy such as that of its supervised equivalents, a comparison would be appropriate.
- Implementation of the library as a wholly functional tool ready to apply to different use cases. These include the generation of NER candidates in an unsupervised manner, a study of keyword prevalence in medical documents against a set of parameters, a semiautomatic validation tool for medical terminology enrichment, and the generation of a knowledge graph. While this bachelor final thesis does not encompass the complete implementation and evaluation of these use cases, a preliminary analysis of or evaluation is encouraged.
- Development of a complete project memory including the following sections: introduction, background, market analysis, concept and detail engineering, execution schedule, technical viability, economic viability, laws and regulations, conclusions and future steps.

1.3. Limitations and scope

The scope of this project includes all the aforementioned objectives, but it does not specifically include all the complementary steps that might be necessary for their completion. For instance, this project does not include the gathering of data for the training of some models, since this data has already been provided by the BSC. It does not encompass the full implementation and evaluation of the use cases of KeyCARE either, although that is currently being developed by the author at the BSC with the intention of publishing research papers. However, examples of application of the library have been performed and are included in this document as a preliminary analysis of the use cases.

The project also presents some intrinsic limitations, which mostly stem from either a lack of resources or time, or from the nature of the proposed approach. The main limitations are:

- Time, since KeyCARE has been mostly developed in the short span of seven months (plus some additional months for the writing of the memory). Since this project includes the training of various NLP models based on Deep Learning (DL), time should be considered a factor of the utmost importance. This is because the training of some of these models can take days to complete, even when ran in the BSC's infrastructure.

- The fact that the training of some NLP models can be very computationally expensive along with a sometimes-scarce availability of computational resources at the BSC. While the provided servers had great computational power, their high demand made them difficult to access at times, what inevitably slowed down the project.
- Another important limitation is the lack of Gold Standard data for keyword extraction in Spanish. This means that the keyword extractors cannot be evaluated on the task they perform but need to be evaluated with data corresponding to similar tasks, as are NER Gold Standard Corpora.
- Also, the fact that the data used for training the relation classifiers has been obtained automatically from the SNOMED-CT structure, rather than from manually annotated relations. This means that the training data might not represent fairly the general understanding of a term, but rather how the term is understood as part of the SNOMED-CT hierarchical structure.
- The fact that unsupervised methods are used for tasks usually performed by supervised models, which have a greater performance the unsupervised alternative. This is because this project targets low-resource languages where no manually annotated data is available, and it has the potential to help produce the resources with which a supervised model could later be trained.
- Additionally, there is the fact that the models implemented in the library have not been directly trained to perform the tasks that are performed in some of the use cases. One clear example of this is when using the library to generate NER candidates in an unsupervised manner. Consider that the library does not exactly perform a NER, but it rather uses keywords extractors and then classifies the extracted keywords into classes, which is not actually the same process.

1.4. Location of the project

This project has been conducted throughout the duration of a 7-months internship in the Natural Language Processing for Biomedical Information Analysis (NLP4BIA) research group at the Barcelona Supercomputing Center. Spanning from July 2023 to January 2024, the project has been developed under the supervision of Dr Luis Gascó and Dr Martin Krallinger. Additionally, the author of this thesis currently maintains a permanent position at the BSC to continue to develop KeyCARE, as well as to work on two research papers on its development and on analyses performed through its application. Additionally, KeyCARE has been developed as part of the project TeresIA, a nationwide CSIC initiative for the accumulation of NLP technologies for the creation of a Spanish terminology [10][11]. Thus, the use cases of KeyCARE and some updates are still under development and the obtained results are soon to be published.

2. Background

This section introduces the background to the developed project, including some NLP general concepts, the State-of-the-Art (SOTA) in the tasks performed by KeyCARE, and the current state of similar technologies to the developed library. All this favours a better understanding of the motivation of KeyCARE and of the project as a whole.

2.1. General concepts

Some general concepts that are key for the understanding of this project are here explained.

Language Models (LMs): as mentioned, NLP is a field that uses statistical, Machine Learning (ML) and DL-based models for the processing of human language. Language Models are models trained on massive amounts of text data, what allows them to process and understand language and perform tasks like generating text, translating languages, and writing different kinds of creative content. Current trends have a tendency towards LM trained through DL to better represent the deep meaning of words in a vector representation.

Transformers: network architecture based solely on attention mechanisms that is positioned as State-of-the-Art for sequence modelling and translation problems, such as language modelling and machine translation [12]. This architecture constituted a breakthrough in NLP, outperforming previous approaches based on Convolutional and Recurrent Neural Networks by far. The use of Transformers for specific tasks usually includes the use of pre-trained models that have already been pre-trained with big sets of data for other tasks. This leveraging of pre-trained models also revolutionized the field by introducing Transfer Learning, which helped reduce training times to a fraction. As shown in Figure 1, there are three main blocks in the Transformer architecture [13]:

- **Encoder:** neural-network-based model that processes an input sequence of text and condenses it into a fixed-length representation, capturing the essential information. Encoders are useful for tasks where there is a textual input, but no text needs to be generated, such as in Text Understanding.
- **Decoder:** neural-network-based model that takes an encoded representation and generates an output sequence based on the encoded information. Decoders are useful for tasks where an encoded input is given and text is generated, such as in Text Prediction.
- **Transformer architecture (Encoder-Decoder):** combined architecture where an encoder processes an input sequence, and a decoder uses that encoded information to generate an output sequence. It is the base architecture for any transformer model, and it allows for complex tasks like Machine Translation, Text Summarization or Question Answering.

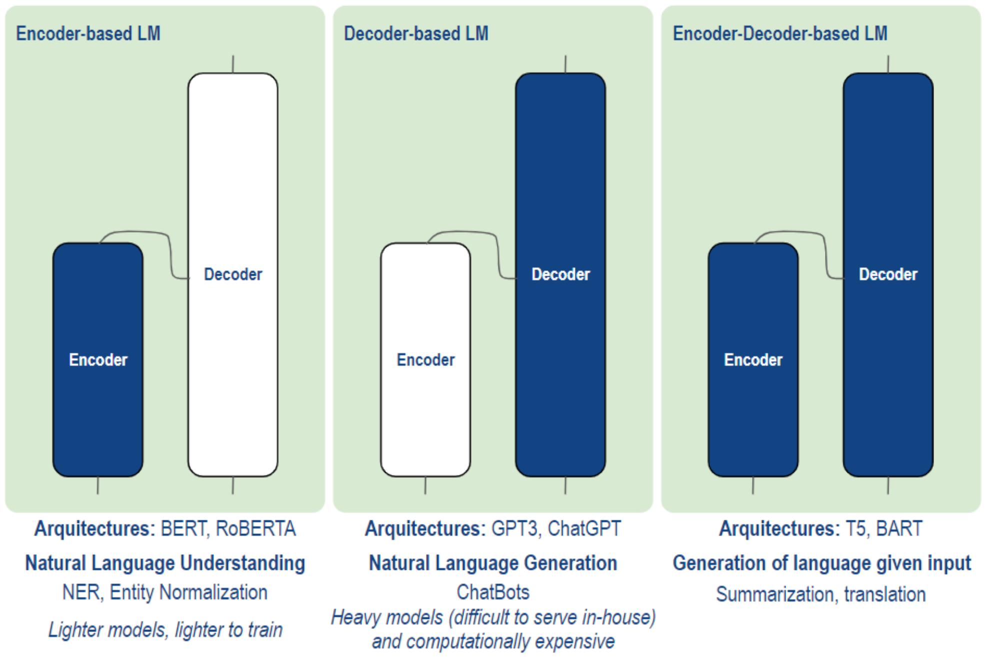

Encoder-based LM
Encoder
Decoder
Arquitectures: BERT, RoBERTA
Natural Language Understanding
NER, Entity Normalization
Lighter models, lighter to train
Decoder-based LM
Encoder
Decoder
Arquitectures: GPT3, ChatGPT
Natural Language Generation
ChatBots
Heavy models (difficult to serve in-house) and computationally expensive
Encoder-Decoder-based LM
Encoder
Decoder
Arquitectures: T5, BART
Generation of language given input
Summarization, translation

Figure 1. Three main architectures of Transformers-based Language Models [14].

Sentence Transformers: Sentence Transformers consists of a library of Deep Learning models based on the Transformer architecture that specialize in generating embeddings for entire sentences. This makes them very useful for many NLP tasks, such as Question Answering, Text Summarization or Text Prediction. Sentence Transformers is a library built on top of Huggingface, a library that allows to use pre-trained models based on the Transformer architecture, a powerful Deep Learning architecture used for various NLP tasks [12].

Basic NLP methods: there are a variety of NLP methods and techniques with different purpose, among which there are some that are commonly used as steps of complex pipelines. The most relevant ones for this project are introduced here:

- **Tokenization:** tokenization is the process of breaking down text into tokens, to prepare them for further analysis using NLP techniques. A token is defined as the smallest unit of text, typically a word, character, or sub-word, used for analysis by NLP models. In addition to that, the tokenization adds special characters that enable the model to perform further processing. An example would be the tokenization of the term “acute gastroenteritis”, which can be tokenized as “acute-gastroenteritis”, “acute-gastro-enter-itis”, or “ac-ute-gas-tro-en-ter-itis”, depending on the task to be performed.
- **Text embeddings generation:** a word embedding is defined as the vector representation of that word in a high-dimensional numerical space. The generation of adequate word embeddings leads to a representation of terms where more semantically similar terms are represented by similar vectors and are far away from semantically different terms. Figure 2 shows an example of the representation of embeddings for medical terms using two Language Models, PubMedBERT [15] and SapBERT [16]. This example shows how related terms like “Coronavirus infection”, “quarantine” and “high fever” are represented closely, while less related terms like “antimalarials” or “Hydroxychloroquine” are represented further apart in the vector space. The generation

of embeddings for words, terms or whole documents is used to provide algorithms and Machine Learning methods with a numerical input so they can grasp the relationships between words.

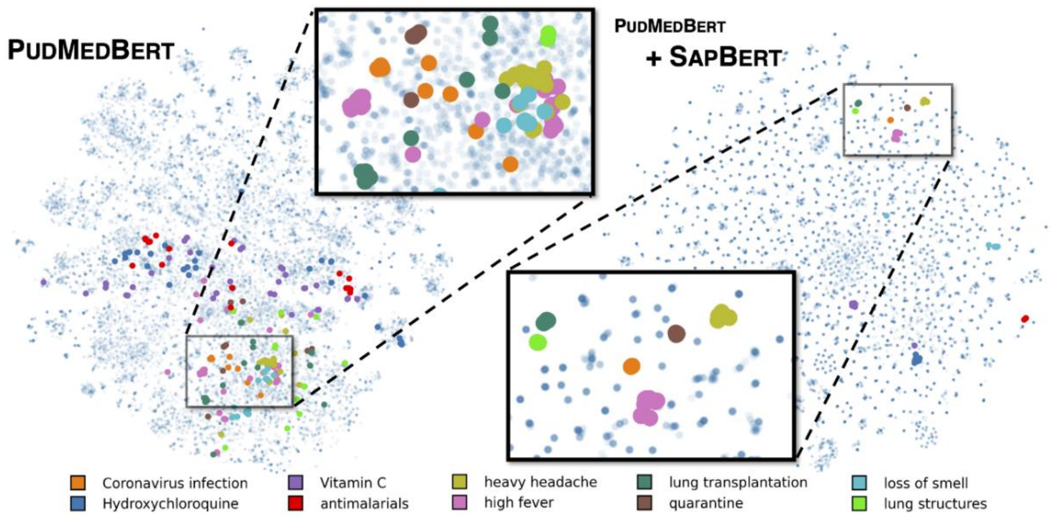


Figure 2. Scheme of the embeddings generation by SapBERT+PubMedBERT when compared to PubMedBERT [16].

- **Data collation:** data collation is the process in which data from different sources or in different formats is standardized into a unified system. In this project, data collation is mainly used in sentences and texts with different lengths, which are zero-padded to the same length before their embeddings are generated.

Part-of-Speech (PoS) tags: Part-of-Speech tags are syntactic labels assigned to words in a text to indicate their grammatic function in a sentence. For example, in the sentence “Resected the malignant tumor” the corresponding PoS tags would be “Verb-article-adjective-noun”. This can be very helpful when identifying entities in a text, and many NLP methods for keyword extraction include parameters to select only keyword with specific PoS tags.

Named Entity Recognition (NER): NER is an NLP task that is used to identify named entities within a text and is one of the key tasks for Information Extraction [2]. As explained, an entity corresponds to a key subject from that text that belongs to a specific class, such as a location, a name, or an object. In a medical context, NER can be used to identify and extract diseases, medical procedures, or symptoms from a medical document. NER systems require huge manually annotated datasets for their training and are very dependent on the embeddings of the text tokens, since the main task they perform is token classification. Regarding the evaluation of NER systems or equivalent methods, it is important to consider overlaps among the predicted entities and the true entities. For instance, if the true entity is “benign neoplasia” and only an exact overlap is considered correct, that same entity must be extracted. However, when also considering a partial overlap as correct, “neoplasia” would also be accepted.

Relation Extraction (RE): RE is another key task involved in IE, and it is used to classify the semantic relations between entities or groups of entities [2]. This is useful for a variety of NLP tasks, as well as for the construction of semantic knowledge databases and the generation of terminologies, ontologies and thesauri. The extraction of relations among a specific term with other concepts can be used for the enrichment of the lexical variants of the term and the detection of neologisms. An example of task based on Relation Extraction is the detection of types of interaction between proteins and drugs, as shown in the Drugprot paper [9]. In the mark of this project, RE is intended as a tool for the identification of existing relations among extracted terms and terminology concepts. Finally, it is of note that recent studies have begun to use NER and RE in a joint manner to tackle IE tasks.

***Is a* Relations:** an *is a* type of relation between terms describes a hierarchy relation where the generality of each term is assessed with respect to the other. Knowing these relationships between terms can be very useful for the creation and assessment of structured hierarchical terminologies. When referring to *is* a relations, the first term – the one that exerts the relation – is called the source term, and the second term – the one that receives the relation – is called the target term. There are four possible *is a* relations between terms: exact, when the two terms mean exactly the same; broad, when the source term includes the target term; narrow, when the source term is included in the target term; and no relation, when none of the previous happens. For example, "Shortness of breath" and "Dyspnea" have an exact relation, "Acute dyspnea" and "Dyspnea" are narrow, "Shortness of breath" and "Acute dyspnea" are broad, and "Dyspnea" and "Hypercapnia" do not have an *is a* relation. For another example, refer to Figure 11 in section 5.3.3. Relations classification.

2.2. Historical background

According to the literature, the first attempts at creating NLP machines was developed by the Germans during World War II, when they developed Enigma, a machine to safely encrypt their messages [17]. In response, the British created an establishment, where Alan Turing, along with a team of intelligence agents, managed to decrypt the messages encrypted by Enigma. That led to Turing publishing a paper in 1950 describing the Turing Test, also known as the Imitation Game, which was the first article to propose a method to determine whether a computer can think like a human or not. This was the first step in the long way that Artificial Intelligence (AI), NLP, and computers have come to the present moment.

After that, NLP started developing exponentially, with the first translation machines being developed in 1954 through a joint project between Georgetown University and IBM Company [18]. In 1969, Roger Schank introduced the concept of tokens, which provided better insights to the machine, but was still dependent on sets of rules that the machine would follow to process text. In 1970, William Woods developed the concept of Augmented Transition Networks (ATN), which used recursion to extract some meaning of a text even when there was not enough information available. But it was not until the 1980s that Machine Learning emerged, providing more advanced ways to interpret ambiguity and provide acceptable evidence for decision-making, therefore greatly advancing the field of NLP. Currently, the focus has been shifted to Deep Learning based models, as they do not require a rule or a fixed criteria and outperform the other models when addressing complex or ambiguous problems. However, the more these models advance, the more data and processing power they require, which has parallelized the advancement of NLP with that of the hardware that supports Deep Learning techniques.

With the advancement of DL, many NLP techniques were developed over time. Named Entities were first introduced in 1995 at the Message Understanding Conferences (MUC), which were of great influence on IE research in the US at that time [19]. Since then, there have been various international efforts to enhance NER systems, with some of the first being IREX (Information Retrieval and Extraction Exercise) in 2000 in Japan, and the shared task CoNLL in 2002 and 2003 for English, German, Dutch and Spanish. However, it was not until 2013 with the introduction of the word2vec models that DL-based systems started dominating the field. Word2vec introduced neural network based LMs that enabled storing deep semantic information of the text in its vector representation [20]. After that, in 2017 this field was revolutionized again with the introduction of the Transformer architecture [12]. The Transformer architecture stablished as a new neural network architecture for NLP tasks that relied solely on attention mechanisms, abandoning previous approaches such as recurrent neural networks and convolutions. This new architecture also popularized Transfer Learning for Transformer architectures. This means that Transformer models that had been trained in one task, such as the generation of embeddings, are leveraged to improve performance on new tasks with fine-tuning, such as text classification.

Finally, current trends point towards the use of generative models as drivers of the improvement in the performance of many NLP tasks. However promising this recent line of research might be, there are still efforts focused on Named Entity Recognition (NER) systems that can handle the complexities of specific domains. In parallel, terminologies, ontologies and thesauri are being developed to procure NER and IE systems with as rich a lexical guide as possible.

2.2. State of the art

This section introduces the State-of-the-art for IE methods, both supervised, like NER and RE, and unsupervised, like keyword extractors. It also includes generalist methods as well as domain-specific methods that are of use for IE in the medical domain.

2.2.1. Information Extraction with NER and RE

According to various studies, NER and RE are defined as the two main tasks for Information Extraction from unstructured textual data since they jointly allow the identification of named entities and the existing relations among them [2]. And both these techniques are currently dominated by Deep Learning-based hybrid and joint models that are consolidated as SOTA.

NER state-of-the-art in English and for a general domain is currently held by supervised and semi-supervised Deep Learning-based approaches, as well as by hybrid models such as Neuro-CRF. This is a model that combines a Deep Learning approach with Conditional Random Fields (CRF) to label text sequences with distributed work representations [21]. In other languages and multilingual scenarios, Deep Learning-based models and hybrid models like Neuro-CRF are faced with limitations in their training due to data scarcity. In these cases, Transfer Learning is employed for cross-lingual NER, which is outperformed by traditional models but can deal with lack of the target language data and shows promising results. Finally, NER suffers with two limitations when facing specific domain scenarios: the fact that only local information is considered and the model's tendency to overfitting due to the lack of labeled data. In data scarcity scenarios, hybrid models might be preferred to DL traditional ones.

Additionally, since the development of ChatGPT by OpenAI, NER trends have veered towards the use of Large Language Models (LLMs). GoLLIE, a LLM trained to follow annotation guidelines, is one of the latest implementations of LLMs for NER tasks [22]. It outperforms previous approaches on zero-shot IE and is able to follow detailed definitions by not only relying on the knowledge already encoded in the LLM. Another enhancement of LLMs for NER is UniversalNER, which uses targeted distillation to train Student models from LLMs [23]. This enables the generation of smaller and more cost-effective models that can excel in a broad application such as open information extraction, even outperforming the original LLMs, such as ChatGPT, and other SOTA multi-task instruction-tuned systems. Finally, one of the most recent NER implementations that is not based on LLMs is GLiNER [7], a generalist model for NER that uses a bidirectional Transformer architecture and enables parallel entity extraction. GLiNER has showed promising results, outperforming both ChatGPT and fine-tuned LLMs in zero-shot evaluations on various NER benchmarks.

RE state-of-the-art is also held by supervised and semi-supervised Deep Learning-based and hybrid models. Particularly, approaches based on Convolutional Neural Networks (CNNs) are widely used for IE problems based on RE, with different implementations for extracting semantic relations among terms, such as attention-based methodologies [24]. Other common DL approaches are Joint Modeling Approaches and Transfer Learning, of which the later has proved useful in low-resource languages and domain-specific scenarios. Recent advances in IE haver also shown that joint NER-RE can enhance the depth of the extracted relations by considering their context. An example of this is SMAN, a Span-based Multi-Modal Attention Network that simultaneously extracts the context and span position information, and jointly models the pan and the label in the RE stage [25]. Another recent approach consists of using networks featured by transformers with non-regressive parallel decoding, which directly output relational triples in one shot while maintaining their intrinsic order [26].

As seen, the state-of-the-art for Information Extraction relies on DL-based implementations of NER and RE systems. However, these systems require huge amounts of training data and only some variations of them manage in low-resource languages and domain-specific scenarios, such as those based on Transfer Learning. This is why this project proposes an approach that is not based on NER systems, but rather on unsupervised keyword extraction and unsupervised or few-shot term classification. Relation Extraction is still conducted in a supervised manner but trained with data obtained from previously existing ontologies. Therefore, the presented library does not perform the State-of-the-art techniques for IE, but rather IE techniques that do not need manually annotated training data and present high scalability, even if they do not outperform NER systems.

2.2.2. Information Extraction in the medical domain

The medical field is constantly evolving to try and optimize medical processes, which leads to enhanced patient care and outcomes. Examples of this include the development of Information Extraction techniques that are specifically designed for biomedical-domain texts. These would allow doctors to understand a patient's previous clinical history in a fraction of the time it takes them currently. The pipeline for IE from electronic medical documents is also mainly based on NER followed by RE.

Both NER and RE SOTA for biomedical documents include rule or dictionary-based methods, as well as ML or DL-based methods [27]. While rule-based methods might achieve higher scores when dealing with specific datasets, they require manual construction by a medical expert and present an extremely low-scalability. This has led to the current tendency to prioritize IE methods based on either ML or DL. Like for NER models for general domains, CRF-based models are considered State-of-the-Art for NER in the biomedical domain, along with Structural Support Vector Machine (SSVM), among others. Regarding RE, the some of the most used ML-based methods are also CRF, SVM, Long Short-Term Memory (LSTM) and Graph Convolutional Networks (GCNs). For both NER and RE, current tendencies are veering towards LLMs, even if Pre-trained Language Models (PLMs) still outperform them in IE tasks in the biomedical domain [28].

An example of the enhancement of NER systems for biomedical documentation in languages other than English is CMF-NERD [28]. This approach uses a random sampling algorithm based on recent SOTA few-shot methods and a LLM for NER in Chinese. While the results did not reach the expected scores, this method shows promise for the development of a few-shot NER that does not require huge amounts of manually annotated data. Another example of recent advancements in NER and RE in the biomedical field is presented in the Rule-enhanced Drug Description Information Extraction, which combines rule-based and DL methods for instruction retrieval in from drug manuals [28]. This method has shown remarkable results for knowledge extraction from pharmaceutical instructions in Chinese.

While there are many innovative approaches for IE in the medical domain, one of the current challenges in this field is the accurate representation of biomedical entities through embeddings in a vector space [29]. This has led to the development of different strategies to tackle text embeddings. One of the main frameworks in NLP for sentence and text embeddings generation is Sentence Transformers [30]. Sentence Transformers is a Pytorch-based framework that provides State-of-the-Art pre-trained models for sentence and text embeddings in over 100 languages and in different domains. For instance, a Transformers pre-trained model that is specific for the biomedical domain is SapBERT (Self-Alignment Pre-training for Biomedical Entity Representation) [16]. SAPBERT has been recognized as SOTA for medical entity linking as well as in the scientific domain. It has outperformed various domain-specific pre-trained language models such as BioBERT, sciBERT and PubMedBERT, and it has already been developed in Spanish, as well as in English.

2.3. State of the situation

KeyCARE has been developed as part of **TeresIA**, a nation-wide initiative of the Spanish National Research Council (CSIC) that aims to develop terminologies, data platforms, and NLP technological infrastructures to support AI and drive the country's digital transformation [10]. Figure 3 shows the main functionalities of the TeresIA project, which include the extraction of new terminologies and their relation among them and with pre-existing ones, the detection of new terms, the annotation of documents, and evaluation of the quality of the implemented systems. This project is aligned with Spain's National Artificial Intelligence Strategy (ENIA) and is part of the 2024 National Strategy for AI [31]. It involves the participation of many public institutions, including the BSC, the Ontology Engineering Group-Universidad Politécnica de Madrid (OEG-UPM), the Cervantes Institute, the Translation General Direction of the European Commission, and the Spanish Association for Terminology (AETER). TeresIA was recently presented before the European Commission, receiving 1.4 million euros in funding from the Digitalization Secretariat [32], as well as an award in the field of innovation and entrepreneurship by the 2024 Internet Awards [33] One of the pillars of the participation of the BSC in this project has been the implementation and assessment of automatic methods for extracting terms as well as the extraction of semantic relations for terminology enrichment, all of which has been developed within the KeyCARE library. In addition to that, TeresIA focuses on tools and terminologies belonging the specific domains of medicine, research and the juridical field, of which the BSC has focused on the medical domain.

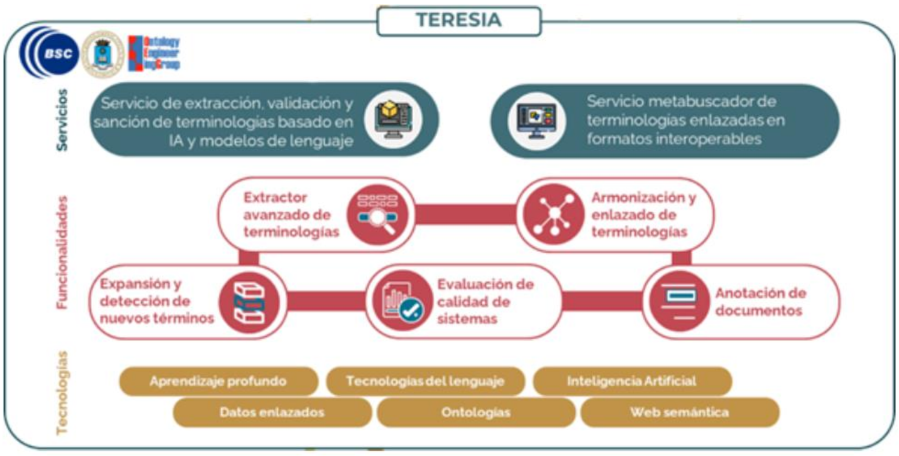

TERESIA
Servicios
Servicio de extracción, validación y sanción de terminologías basado en IA y modelos de lenguaje
Servicio metabuscador de terminologías enlazadas en formatos interoperables
Funcionalidades
Extractor avanzado de terminologías
Armonización y enlazado de terminologías
Expansión y detección de nuevos términos
Evaluación de calidad de sistemas
Anotación de documentos
Tecnologías
Aprendizaje profundo
Tecnologías del lenguaje
Inteligencia Artificial
Datos enlazados
Ontologías
Web semántica

Figure 3. Main goals, functionalities and technologies of the TeresIA project [10].

3. Market analysis

After introducing the SOTA for the main Information Extraction techniques, this section delves into the current situation of these techniques in the market. This includes an overview to the market sector in which it stands, as well as of both private and public initiatives that have a similar goal as this project. This helps identify both opportunities and threats for the project as well as establish the future perspectives of the sector in which it is located.

3.1. Market sector

Since the release of ChatGPT and other publicly available LLMs, interest in AI has skyrocketed, pushing continuous research and innovation in the filed both by tech giants and public organizations. The global AI market size in 2023 reached an estimated 196.63 billion USD and is projected to grow past this with a Compound Annual Growth Rate (CAGR) of 36.6% from 2024 to 2030 [34]. This term represents the mean annualized growth rate for this market sector in the indicated period. Within the AI market, as much as much as 27.73 billion USD corresponded to the development of NLP tools in 2022, which is predicted to grow with a CAGR of 40.4% between 2023 and 2030 [35]. However, with an increase in the size of LLMs also comes an increase in the size of the data that they need to be trained on, and therefore a need to generate vast amounts of structured data. When referring to specific domains, as is the medical field, there is a latent need to generate and enrich terminologies where medical concepts are standardized in a set structure. Enriching current terminologies and generating new ones will enable way of uncovering medical knowledge, as well as the training of much more specific DL-based tools than the current ones. However, there is still a need for automatic NLP tools that are able to generate the adequate information for terminology enrichment.

3.2. Competitive analysis

In Spain there have been numerous initiatives and organizations that have tried to tackle those issues for decades. An example is the Spanish Society for NLP (SEPLN), which was founded in 1983 and promotes collaboration and innovation among different national and international organizations for the advancement of NLP tools in Spanish [36]. The SEPLN collaborates with many research groups that focus on NLP, such as the OEG-UPM, the UNED's research group in NLP, or the Language and Speech Technologies group at the Polytechnic University of Catalonia (TALP-UPC). Many of these groups are developing research lines for the improvement of terminologies and IE techniques, such as the OEG's efforts for collaborative ontology generation [37] or the TALP's initiative for Spanish open domain Information Retrieval [38]. In addition to that, many research groups are focusing on the developing of NLP techniques for the medical domain, like the UNED's NLP research group with its EDHER-MED project [39]. This project aims to provide a tool for the early detection of likely health risks in patients from the analysis of medical documents.

Two other pioneering groups in the application of NLP to the medical domain are the BSC's Language Technologies Unit (LTU) and the NLP4BIA group, within which this project has been developed. Both these groups participate in nation-wide initiatives to improve the quality of NLP resources in Spanish both in the biomedical domain and in general. For instance, both NLP4BIA and LTU are part of the Language Technologies Development Plan (planTL) of the Spanish Government [40]. This plan seeks to promote the development of NLP, machine translation, and conversational systems in Spanish and co-official languages, increasing the Language infrastructure in Spanish and coordinating NLP development efforts. Another example of this is the development of the AINA project [41], funded by the Catalan Government and conducted at the BSC to generate datasets and develop computer models to facilitate the development of AI-based applications in Catalan, such as voice assistants, search engines, translators, spell checkers, and chatbots.

As seen, there are many initiatives at a Catalan, Spanish and even European level that target the creation and enrichment of terminologies in general and specific domains through the implementation of different NLP techniques, some of these being specific for Information and Relations Extraction. However, there is a need for the integration of these technologies and the generated ontologies in frameworks that allow their coordination. This is why KeyCARE and TeresIA, the parent project within which KeyCARE has been developed, are tackling this need for the Spanish language and with special attention to the medical domain.

3.3. Future perspectives

As mentioned, the market sizes of NLP and AI are expected to continue to grow exponentially, producing every time more complex and accurate systems. The state of the development of NLP technologies can be observed through a Gartner curve as the one in Figure 4 [42]. This figure shows how different technologies advance through the different phases of their development in a graph of expectations against time. As seen, NLP is currently located in the Trough of Disillusionment, which means that the current state of the technology has not met the initial expectations due to real-world challenges and still needs some time to be established as a completely productive and mainstream technology. However, this figure also shows that many subtasks of NLP, such as Text Classification, NER, Question Answering and Document Summarization are already entering the phase of the Slope of Enlightenment. In this phase, technologies have already overcome the Trough of Disillusionment and are on its way to the Plateau of Productivity, which indicates when these technologies will reach mainstream adoption and deliver their practical benefits. In general, Figure 4 shows that most NLP-related tools will reach this last phase in 5 to 10 years.

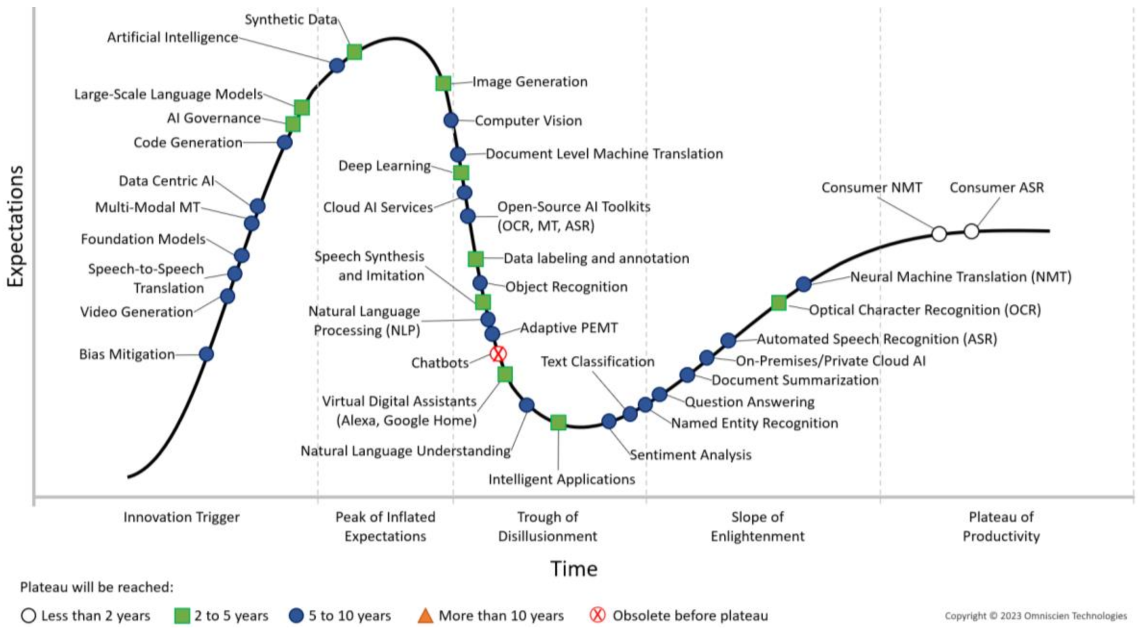


Figure 4. Gartner technology curve showing the future perspectives of NLP techologies [42].

Regarding the future perspectives of the KeyCARE project within, it will continue to develop within the mark of the TeresIA Project, which is funded and set to last for at least until the second half of 2025. This will ensure that more IE and RE methods are implemented to improve the framework as a whole and generate more valuable results. It is also the hope that, thanks to initiatives like this one, structured information and terminologies will have been produced for the medical domain both in Spanish and Catalan. This will allow for better analysis of the medical data as well as for the implementation of more complex technologies for IE in the medical sector, improving the quality of the medical processes.

4. Concept engineering

This section shows a general overview of the pipeline that is implemented in KeyCARE, while outlining the possible implementations of each of its parts. This requires the definition of the needs that the library must meet, as well as a clear definition of its applications and what they entail. The definition of the needs for each part of the pipeline allows for a complete comparison of different implementation alternatives, leading to the selection of the optimal alternative.

4.1. Functional requirements of the library and user needs

KeyCARE is a Python library defined with three main functions: keyword extraction, term categorization and relation extraction. And since its aim is to be of use in different contexts and with a variety of applications, its implementation must meet some functional requirements to comply with prospective user needs. When using the KeyCARE main classes with all the default parameters, it must provide a framework that is functional when working with texts in the biomedical domain and in Spanish. Therefore, all the default models must use base models and training data obtained from medical records and literature in Spanish. Nevertheless, the library must be flexible enough to be usable in domains other than the medical one and in languages other than Spanish. It must be customizable to include as many specifications and parameters as the user might require, and easy to integrate in language processing pipelines. Additionally, it must be scalable and efficient, so it can be applied efficiently to large datasets without performance degradation. The specific requirements of the library are the following:

- **Unsupervised keyword extraction:** provide a set of different methods and models for unsupervised keyword extraction, while allowing the complete tunning of the parameter configuration of each method, as well as extra parameters. The extracted terms shall be provided with additional information such as their span within the text and processed with adequate tools for stopword removal, among other functions. Additionally, any method using a language model must give the option to choose the used base model other than the default one.
- **Classification and clustering of keywords:** provide both supervised and unsupervised methods for the classification or clustering of the extracted keywords, also allowing the complete tunning of the parameter configuration of each method, as well as any extra parameters. Since a clustering algorithm might not successfully identify groups among the provided terms, the library shall provide a classification model that functions in a few-shot manner. For the methods using a language model, an already trained classification model must be set as default, but the option for training and storing a new model from scratch must be provided, as well as the option to import an already trained model. When training a model from scratch, the library must provide tools for an exhaustive evaluation of the trained model.
- **Classification of hierarchical relations:** provide supervised methods for the classification of hierarchical relations among terms or phrases, also allowing the complete tunning of the parameter configuration of each method, as well as any extra parameters. It shall be able to handle inputted term pairs in different formats and it must return the corresponding relation type. Additionally, it must give the option to choose an

already trained model rather than the default one, which might handle other semantic relationships rather than hierarchical ones.

- **Multilingual support:** all the methods and models implemented in the library must be set by default in Spanish, but they must provide multilingual support. This might entail changing the base models of some methods as well as training models from scratch in the desired language.
- **Domain independence:** all the implemented methods and models must be designed to work in texts from the biomedical domain by default, but they must provide domain adaptation. This might entail changing the base models of some methods as well as training models from scratch with data from the desired domain.
- **Structure of the library:** the library shall be implemented with a modular structure that enhances its performance and interpretability while enabling its easy customization and extension. For that, it must be designed with an optimized class structure with integrated functions. This should also enable the library's easy application to a variety of use cases that might only need some part of the library's pipeline.

4.2. Pipeline and structure of the library

Once the functional requirements of the library have been defined, its structure and the implementation of its pipeline have been discussed. The library has been developed in Python since it constitutes the most extended open-source programming language for NLP, providing a wide variety of modules and libraries for language processing tasks. The mainly used modules for the implementation of the main three NLP tasks implemented in KeyCARE are **Transformers**, **Sentence Transformers**, **NLTK**, **Spacy**, and **PyTorch**. Regarding the library architecture, it is designed to ensure the workflow of the three main functions of the library as part of a pipeline: keyword extraction, their classification, and the extraction of their relations among them and with other terms. For that, the library is implemented with two main classes, one for the extraction and classification of keywords and another one for the extraction of hierarchical relations. These classes shall call on other classes where the keyword extractors, classifiers and relation extractors are implemented.

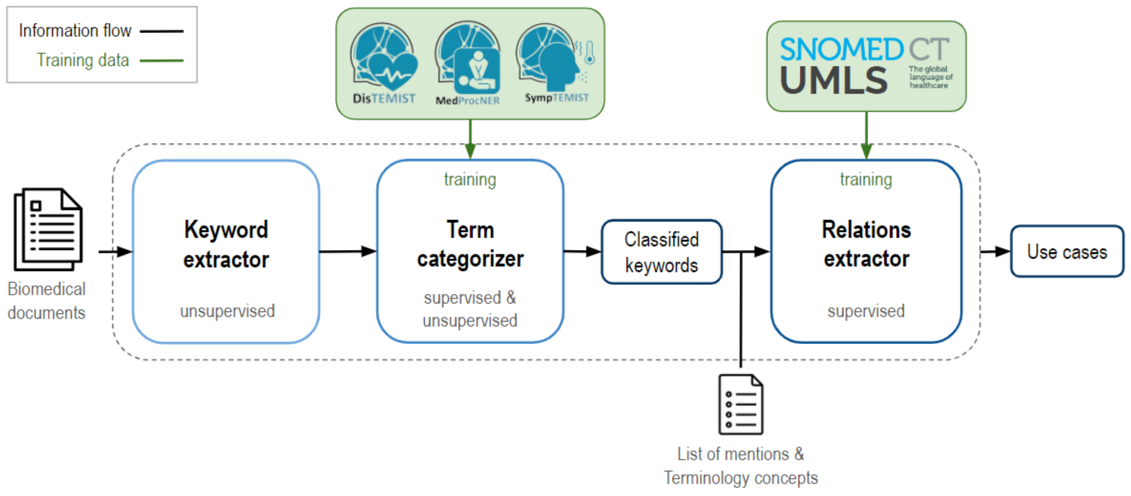


Figure 5. General elements and workflow of KeyCARE. Own creation.

The general pipeline of the library is shown in Figure 5. As seen, documents of the biomedical domain are first introduced and processed by the keyword extractors, which are unsupervised methods and therefore do not require training data. The extracted keywords are then passed to the term categorizers, which give place to classified keywords. The term categorizers can be unsupervised clustering models or supervised classifiers, in which case they require training data obtained from NER Gold Standard Corpora. Finally, the classified keywords are passed to the relations extractors jointly with other lists of mentions or terminology concepts, so that relations might be extracted among them. The relations extractors function in a supervised manner, for which they have been trained on structured data from SNOMED-CT and UMLS (Unified Medical Language Systems). All these lead to the application of the library to a variety of use cases.

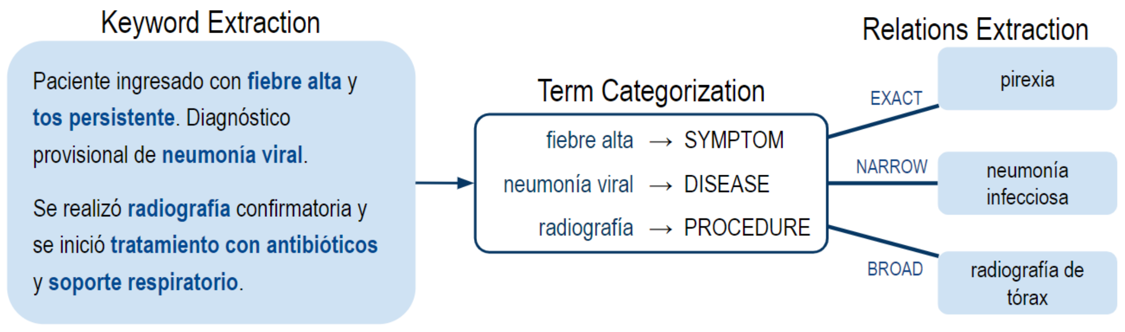

Keyword Extraction
Paciente ingresado con fiebre alta y tos persistente. Diagnóstico provisional de neumonía viral.
Se realizó radiografía confirmatoria y se inició tratamiento con antibióticos y soporte respiratorio.
Term Categorization
fiebre alta → SYMPTOM
neumonía viral → DISEASE
radiografía → PROCEDURE
Relations Extraction
EXACT
pirexia
NARROW
neumonía infecciosa
BROAD
radiografía de tórax

Figure 6. Example of the ideal functioning of the three main functions of KeyCARE. Own creation.

The implementation of the pipeline seen in Figure 5 should give place to results as the one in Figure 6. This figure shows how keyword extraction should be performed from a medical record, and how the extracted keywords should be classified into different categories, such as symptom, disease, and procedure. Finally, it also shows how hierarchical relations among the extracted keywords and with other terms might be extracted.

4.2.1. Unsupervised keyword extraction

Since one of the library's goals is to perform an unsupervised equivalent of a NER that requires as little training data as possible, the first logical step is keyword extraction. Keyword or keyphrase extraction methods are mainly used for terminological extraction to retrieve relevant information from texts in a structured manner. These methods use different metrics to assess which terms hold a more similar meaning to that of the whole document and are therefore more relevant in describing it. An example of how keyword extraction should function in a medical document in Spanish is shown in the first step of Figure 6.

A wide variety of unsupervised keyword extractors have been considered for the implementation of this first step of the library. These include methods based on statistical descriptors, graph-based methods, and methods based on LMs, among others. A short description of the proposed methods with their respective advantages and disadvantages is shown below:

- **TF-IDF (Term Frequency-Inverse Document Frequency):** method to assess word importance in a document with relation to the entire corpus based only on word and document frequency [43]. It is language-independent, fast, and simple. It might favor terms that are frequent in a document but not in others, it requires many documents to function correctly, and it does not capture meaning or context.

- **YAKE! (Yet Another Keyword Extractor):** method to assess term importance based on five statistical features of the candidate terms. It is efficient, language-independent, and works on single documents, but does not capture meaning and it might produce redundant phrases [44].
- **RAKE (Rapid Automatic Keyword Extraction):** Graph-based method that assess word co-occurrences to select keywords. It is efficient, language-independent, and operates on single documents, but relies heavily on stopwords, and might not perform well with long documents [45].
- **TextRank:** graph-based method that focuses on word importance within the document's co-occurrence network. It is language-independent, scalable, and efficient for large datasets, but requires efficient parameter tuning and might miss frequently used terms [46].
- **EmbedRank:** DL-based method that captures semantic similarity using word and document embeddings [47]. It works on single documents and it is highly accurate and customizable, but it requires a pre-trained model, it is computationally expensive and has difficult interpretability. This and the other DL-based models generally need to be trained on GPU, since they require more computational resources than traditional approaches.
- **KeyBERT (Key Bidirectional Encoder Representations from Transformers):** DL-based model that uses Transformers-based word embeddings to assess the terms that represent the document better [48]. It offers customization and easy integration, as well as high precision, but shares the limitations of EmbedRank.
- **PhaseFormer:** multimodal method that combines word embeddings and graphs for deep semantic understanding [49]. It is highly accurate and customizable, but it is very computationally expensive, it has low interpretability, and it is scarcely documented.
- **AttentionRank:** DL-based method that focuses on word attention to assess which keywords are of more importance to the whole document [50]. It is highly customizable and captures deep context relations, but requires a pre-trained model, is computationally expensive, and is not yet implemented as a module.

Since all the proposed methods have their own advantages and might perform better in different scenarios, it has been decided to implement at least four different alternatives as part of KeyCARE. All the implemented alternatives have been chosen to procure a variety of options while maintaining multilingual support and domain independence. Additionally, methods that operate on single documents have been prioritized. Firstly, YAKE has been implemented as a method based on statistical descriptors that usually outperforms TF-IDF. Both RAKE and TextRank have been implemented as graph-based methods that might function differently in different scenarios. Finally, KeyBERT has been implemented as the LM-based method that presents a highest proven cost-efficiency and that supposedly will achieve the highest accuracy. This will enable users of the library to choose between the provided models (or many of them at once, if needed), depending on their priority: whether it's the final performance, the scalability of the project or the computational cost. Additionally, some of the other models might be added to the library in a near future. For instance, AttentionRank will be implemented to the library as soon as there is a fully functional library of the method.

4.2.2. Term categorization

The step that follows keyword extraction in the defined pipeline is the categorization of the keywords, which can be performed in a supervised or unsupervised manner. The supervised version of term categorization consists of a task of multilabel sentence classification, while its unsupervised version corresponds to a clustering of the sentences into different groups. It is important that the classification is performed in a multilabel manner, since medical terms can often belong to more than one semantic category, as is the case for diarrhoea, which can be intended as a symptom or as a disease. Since the goal of KeyCARE is to provide a framework that functions with as little training data as possible, unsupervised clustering models and few-shot multilabel classifiers are highly valued. An example of how the classification of the extracted keywords should function can be seen in the second step of Figure 6. As seen, the keywords are classified into classes according to the semantic group to which they belong in the biomedical domain, as are diseases, symptoms, and medical procedures, among others.

For the implementation of the term categorizer, both unsupervised and supervised approaches have been considered. The unsupervised approach would consist of a clustering algorithm, while the supervised alternatives would consist of DL-based sequence classifiers based on the Transformer architecture. A short description of the proposed methods with their respective advantages and disadvantages is shown below:

- **Clustering:** for the unsupervised implementation of the term categorizer, a clustering model based on the vector representation of the terms has been considered. This would consist of applying a clustering algorithm such as K-means to the embedding representation of the terms, previously generated using a pre-trained LM. This implementation would also allow for other unsupervised methods such as kNN or decision trees with simple modifications. The main advantage of this method is that it does not require labelled training data and thus can be used in low-resource languages or specific domains where there is no annotated data. However, its performance is significantly low, and it might produce clusters that do not correspond to the target classes.
- **Huggingface's AutoModelForSequenceClassification:** the AutoModel for Sequence Classification of the Huggingface library is a generic AutoClass that leverages the Transformer architecture to accurately represent textual data and classify it into groups [51]. It is simple to implement and specifically designed for sequence classification, achieving high accuracy in that task. However, it is a generic class with low configurability, it requires large amounts of labelled training data, and is computationally expensive (requires GPU).
- **SetFit (Sentence Transformers' Fine-Tunning):** SetFit is a few-shot classifier based on the Transformer architecture that is trained using contrastive learning from sentence pairs [52]. Due to its few-shot nature, it only requires small amounts of labelled data for its training and is still able to achieve results comparable to those of the

AutoModelForSequenceClassification. It is also easy to implement and performs especially well with short terms. However, it might be prone to overfitting, and it is computationally expensive (requires GPU).

Consider that the clustering algorithm requires a LM for the generation of the embeddings, and both classifiers require a pre-trained LM as a base model. In all these cases, the Spanish version of **SapBERT** has been chosen as default.

Like for the keyword extractors, each proposed method has its own advantages and disadvantages, so all the provided possibilities have been implemented as part of KeyCARE. A K-means clustering could be used in a situation where there is no labelled data whatsoever and there is no possibility to generate it swiftly. In case a small dataset could be generated, SetFit would represent the best option, largely outperforming the unsupervised alternative. Finally, the Transformer's AutoModel for Sequence Classification would only be used in scenarios where SetFit might not perform well and there is a huge availability of training resources.

4.2.3. Classification of semantic relations

The third step of the pipeline implemented in KeyCARE is the extraction of semantic relations between keywords and with other terms. In this case, the aim is to classify term pairs according to their hierarchical relationship (*is a* relationship type) between a source mention and a target mention. An example of how this should function is also shown in the third step of Figure 6. As shown, *EXACT* relations are found between synonyms, like "pirexia" and "fiebre alta"; *NARROW* relations indicate when a term is contained in another one, like "neumonía viral" with "neumonía infecciosa"; and *BROAD* relations indicate when a term contains another one, like "radiografía" with "radiografía de tórax".

For the implementation of this relation classifier, only supervised methods have been considered since unsupervised approaches generally do not have enough data to identify the types of relations among term pairs. The two proposed supervised approaches for term pair classification are the Transformer's AutoModel for Sequence Classification and SetFit, same as for the term classification. A short comparison of their advantages and limitations for the proposed task is shown below:

- **Huggingface's AutoModelForSequenceClassification:** this Huggingface's autoclass is also designed for sequence pairs classification, not only for single sequences [51]. Therefore, it provides an easy implementation for the classification of hierarchical relations, and it performs greatly. Nevertheless, it still presents with low configurability, a need for large amounts of labelled data for training, and it is computationally expensive, requiring a long training time.
- **SetFit:** the SetFit classifier is solely designed for single sequence classification, but it can be used for sequence pairs classification by classifying strings of the form "source mention </s> target mention". The main advantage of SetFit is also its few-shot nature while maintaining a similar accuracy to regular classifiers [52]. On the other hand, it is not designed for sequence pairs classification, which might lead to a misrepresentation of the

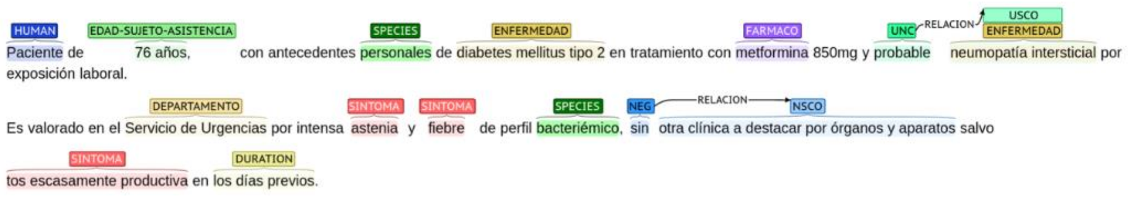
data in this task. This adds up with its tendency to overfitting and with its long training time and need for expensive computational resources.

Consider that both sequence pairs classifiers require a pre-trained LM as a base model. In all these cases, the Spanish version of **SapBERT** has been chosen as default.

In this last case, both alternatives have also been implemented in the KeyCARE library to provide flexibility on the chosen approach. While the Transformer's AutoModel for Sequence Classification is directly intended for the classification of term pairs and is therefore the most adequate method, it does require huge amounts of labelled training data. Therefore, in scenarios where only small sets of data can be generated SetFit can be used as a solid alternative.

4.3. Training and evaluation of the models

Keyword extraction: since the keyword extractors function in an unsupervised manner, there is no need for labelled training data. Regarding their evaluation, since there are no corpora available to evaluate Spanish keyword extraction systems, this model will be evaluated using datasets for evaluating NER systems. Figure 7 shows an example of a text with annotated entities of different classes such as the ones used for evaluation of the keyword extractors. Although keyword extraction is not the same task as entity recognition, evaluation corpora for this type of task are a good way to measure whether unsupervised systems are capable of extracting most of the biomedical entities of interest from a text. NER Corpora for Medical entities such as medical procedures, diseases or symptoms are used.



Paciente de 76 años, con antecedentes personales de diabetes mellitus tipo 2 en tratamiento con metformina 850mg y probable neumopatía intersticial por exposición laboral.

Es valorado en el Servicio de Urgencias por intensa astenia y fiebre de perfil bacteriémico, sin otra clínica a destacar por órganos y aparatos salvo tos escasamente productiva en los días previos.

Figure 7. NER annotated dataset example. Generated by the NLP4BIA group at the BSC. Own creation.

Term categorization: for the training process of the term classifiers, data from annotated NER Corpora has been used as well. In this case the entities have been used as classified terms where their class is the assigned NER label. The same data has been used for evaluation.

Relations extraction: for the training process of the relation classifiers, hierarchical relations among pairs of terms are needed. Thus, data from structured terminologies such as SNOMED-CT and UMLS has been used to generate the training pairs with different types of hierarchical relations. The same data has been used for evaluation.

4.4. Implementation of the use cases

KeyCARE is proposed as an easy-to-use framework that aims to provide the necessary set of tools to perform keyword extraction, their categorization and semantic relation. While the objective of this project is to develop the library, a few practical applications with goals related to information extraction, terminology enrichment and text processing within the biomedical domain have also been proposed. These are meant as a way of qualitatively validating the library, rather than truly assessing its functionality. Thus, the main use cases that have been implemented as part of this project are defined here:

1. **The generation of NER candidates in an unsupervised manner:** using the keyword extractors and the term classifiers, keywords and keyphrases belonging to a specific class can be extracted. These classified keywords could constitute some kind of Silver Standard for Named Entities in texts of the biomedical domain. This would be extremely useful in low-resource languages, where there is no annotated data to implement actual NER systems. This use case has been implemented and evaluated in Spanish, since it is the language in which there are the adequate manually annotated corpora to evaluate the results. However, an implementation in a low-resource language such as Catalan has also been produced, even if it cannot be evaluated.
2. **The creation of a semiautomatic tool for relations validation:** using the keyword extractors to extract relevant terms from medical documents or bibliography, a large quantity of terms within the domain can be obtained. These terms can then be classified into categories using the classifier and compared to terms from standardized terminologies using the relations extractor. This can help identify new terms that are related to pre-existing terminologies, as well as find new relationships among terminologies and ontologies, which is also of use for cross-ontology mapping. This use case has been implemented on a set of sample documents, even if it cannot be evaluated. Note that the use of this tool is proposed in a semiautomatic manner, since it is not intended to perform as well as a human annotator, but rather to generate candidates that could be validated by an expert.
3. **The study of keyword prevalence and co-occurrence against different parameters:** using the keyword extractors and the classifiers, the keywords that best represent each document of a corpus of the biomedical domain can be extracted. Then, a study of the prevalence of these keywords against different parameters associated with each document can be performed. For example, the study of the keyword prevalence against the publication year of the papers can lead to the identification of medical neologisms and to the assessment of the popularity of different procedures over time. Additionally, the study of term prevalence against the DeCS Codes (from the DeCS terminology) associated with each document can lead to identifying the keywords that better represent papers on a particular topic. In this project, only the analysis of the prevalence of a few terms against the publication year of the papers is presented. On the other hand, the analysis of the co-occurrence of the extracted keywords in the same documents can lead to the generation of a knowledge graph for medical terms. For instance, such a graph would allow to build relationships between diseases and symptoms that occur together frequently. Thus, it would be of use to help create a structured terminology in which different relation types could be represented. In this project, a graph generated from some of the most common keywords in a corpus of research papers has been generated.

5. Detail engineering

This section focuses on the definitive implementation of KeyCARE, including examples of its functioning as well as the scores for the methods and models used in each of the steps that conform the library. It also entails the definition of the library's architecture as well as its corresponding documentation, future maintenance, and applications.

5.1. Architecture of the library, modules and classes

KeyCARE has been published in GitHub [53] and is accessible through PyPI [54]. PyPI (Python Package Index) is the official software repository for Python, where software packages for various functionalities are stored and new packages can be uploaded. Its structure and the files that conform it can be accessed through https://github.com/nlp4bia-bsc/KeyCARE [53], and it can be imported into a python script simply using the command import. Refer to the provided link and access the library for a better understanding of its structure[2].

Figure 8 shows the class structure of KeyCARE, along with the relations among the implemented classes. These include relations of inheritance and information flow as well as the main classes of the pipeline calling on other classes with integrated functions. The main pipeline of the library is implemented through two main classes - TermExtractor and RelExtractor – which call on the other classes with built-in functions. TermExtractor is responsible for executing the unsupervised keyword extraction pipeline as well as their classification or clustering, for which it calls on other classes. These classes include the extractors - RakeExtractor, YakeExtractor, TextRankExtractor

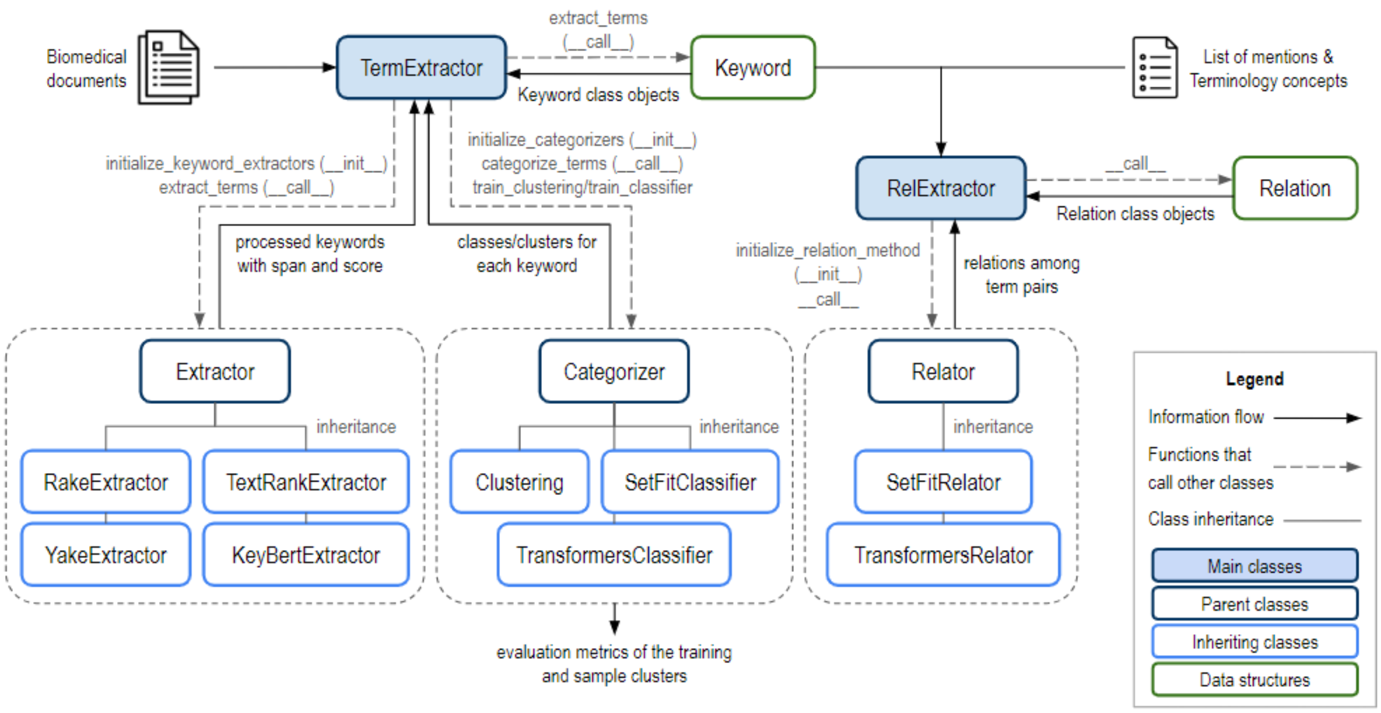


Figure 8. Structure of the main classes of KeyCARE. This figure identifies the classes as Main classes, Parent classes, Inheriting classes, and classes used to create data structures. It includes relations among classes describing information flow, calling of one class to another, and relations of inheritance among classes. Own creation.

[2] It is strongly recommended to look at the tutorials in the /nbs folder of the repository to understand the full functionalities of KeyCARE [53].

and KeyBertExtractor – which contain the methods and models for keyword extraction, and all inherit from the Extractor parent class. Other called upon classes are the categorizers - Clustering, TransformersClassifier and SetFitClassifier – which contain the models for keyword clustering and classification, and all inherit from the Categorizer parent class. Finally, TermExtractor also calls upon the Keyword class, which is a data structure used to store the extracted and classified keywords. On the other hand, RelExtractor is responsible for executing the classification of relations among terms and keywords, for which it calls on other classes. These include the relators – TransformersRelator and SetFitRelator – which contain the models for sequence pairs classification and inherit from the Relator parent class. Additionally, RelExtractor also stores the relations in a specific data structure provided by the Relation class.

The architecture shown in Figure 8 has been designed specifically for the pipelines associated with the two main classes. TermExtractor is implemented to outline the pipeline of keyword extraction and their posterior classification or clustering. On the other hand, RelExtractor is responsible for the term pairs relation pipeline.

Keyword extraction: regarding the keyword extraction task, TermExtractor first uses the *initialize_keyword_extractors* function to initialize the keyword extractors with the specified parameters. These parameters include the extractors to be used, the language, the number of keywords to be extracted, and the maximum number of tokens of the extracted keywords. They also include the option to extract keywords based on their Part of Speech (PoS) tags and to specify the PoS pattern. In the case more than one extractor is selected, it provides the option to join keywords from the different extraction methods and remove complete overlaps among them. Additionally, a parameter can be specified to postprocess the extracted keywords. This postprocessing includes removing meaningless terms (single numbers, single characters and stopwords), removing stopwords at the beginning and end of keyphrases, and removing non-alphanumeric characters[3]. Finally, additional parameters can be specified to the extractors thanks to the implementation of kwargs to the library, as long as they are valid parameters.

Then, when the TermExtractor class is called upon a text, it calls onto the initialized extractor(s) through the *extract_terms* function, so that the extractors generate the keywords according to the already specified parameters using built-in functions. The generated keywords are passed back to the TermExtractor object, where they are stored in the attribute *keywords* as a list of Keyword class objects with their span, score, and extraction method.

Keyword categorization: for the classification and clustering tasks, TermExtractor first uses the *initialize_categorizers* function to initialize the categorizer with the specified parameters. The parameters include the categorization method to be used, be it clustering or any of the two implemented multilabel classifiers. In the case of clustering, the number of clusters can be specified, as well as the path to a pre-trained LM for the embedding generation before the clustering, the path to an already trained clustering model and a path to store the trained model. In the case of multilabel classification, the maximum number of predicted labels per keyword can be specified, as well as the threshold for the prediction of the labels. Same as for the clustering, a

[3] For a better understanding of this process, please refer to the tutorial of the TermExtractor class or to the source code of the library.

pre-trained LM for the base models and an already trained model for classification can be specified, as well as an output path to store the trained model. Finally, additional parameters can be specified to the categorizers thanks to the implementation of kwargs to the library, as long as they are valid parameters.

Then, when the TermExtractor class is called upon a text, not only does it perform keyword extraction, but it also automatically categorizes those keywords into clusters or classes, depending on the specified categorizer. To do so, TermExtractor calls onto the initialized categorizer using the *categorize_terms* function, so that it categorizes the keywords according to the previously specified parameters using its integrated functions. The label corresponding to each keyword is passed back to the TermExtractor object, which adds it altogether with the categorization method as attributes of the Keyword objects.

Additionally, the TermExtractor class can be used to train both clustering and classification models using any of the implemented categorizers. To do so, the TermExtractor object can be called after its initialization using the *train_clustering* and *train_classifier* functions, respectively. When calling the *train_clustering* function, a list of mentions can be provided as unlabelled training data, to train the pre-trained LM specified in the initialization with the other specified parameters. When calling the *train_classifier* function for either the SetFit classifier or the Transformer's AutoModel, lists of mentions and their respective labels have to be provided both for the training and the testing of the previously specified pre-trained LM. In addition to that, the user can specify the evaluation metrics of the model, which include the generation of a classification report and of a multilabel confusion matrix heatmap. Finally, both functions include the option to specify additional parameters through a kwargs dictionary, as long as they are valid.

Term pairs relation: for the classification of hierarchical relations between groups or pairs of terms and keywords, RelExtractor first uses the function *initialize_relation_method* to initialize the relation classifier with the specified parameters. These parameters include the relation classification method to be used, the language, and the path to an already trained model, if provided. The maximum number of labels per term pair can also be specified, along with the threshold for the classification of the relations. Additionally, it also includes a parameter that indicates whether to compute the relations of each source mention only with its corresponding target mention or to compute the relations with all the combinations between source and target. Finally, it also includes the specification of extra parameters, if applicable, using the kwargs dictionary.

Then, when the RelExtractor class is called on two lists of strings or keywords, it calls on the initialized relation classifier to compute the relation types with the specified parameters using its built-in functions. The relation labels are passed back to the RelExtractor object, where they are stored in the attribute *relations* as a list of Relation class objects with the relation type, the source and target mentions, and the relation classification method. Note that, opposite to the term classifiers, the relation classifiers cannot be trained with built-in functions from the library. This is because relation classification is a more subtle task than sentence classification, so it is recommended to train such models with libraries built specifically with that purpose and then pass the trained model as a parameter to the RelExtractor object.

5.2. Implementation of the pipeline

The KeyCARE library has been implemented to conduct a pipeline including unsupervised keyword extraction, the categorization of the keywords, and the classification of relations among them and with terms. This section focuses on the implemented models and methods for each of the steps of the pipeline. This includes the data on which they have been trained and evaluated, the base LMs used, the pipeline of each method itself, and an exhaustive evaluation of their performance.

5.2.1. Unsupervised keyword extraction

The keyword extraction implemented in TermExtractor is performed in an unsupervised manner. The four extraction algorithms are **YAKE!, TextRank, RAKE, and KeyBERT**; and they can be used individually or together. This section delves into the pipelines of each of the implemented extractors, as well as into their evaluation and final implementation.

An example of the actual functioning of the library on the topic of keyword extraction can be seen below in Figure 9. In the example, TextRank with default parameters has been used to extract keywords from a text of the DisTEMIST Corpus, of which only some relevant parts are shown. As seen, most of the extracted keywords correspond to symptoms, procedures, drugs, and diseases, among others, but it also extracts as keywords non-relevant terms, such as “Mediante”. In addition to that, it does not always capture the whole keyphrases, for instance including “Gleason 4” but not “Gleason 4+4=8”, which has a completely different meaning.

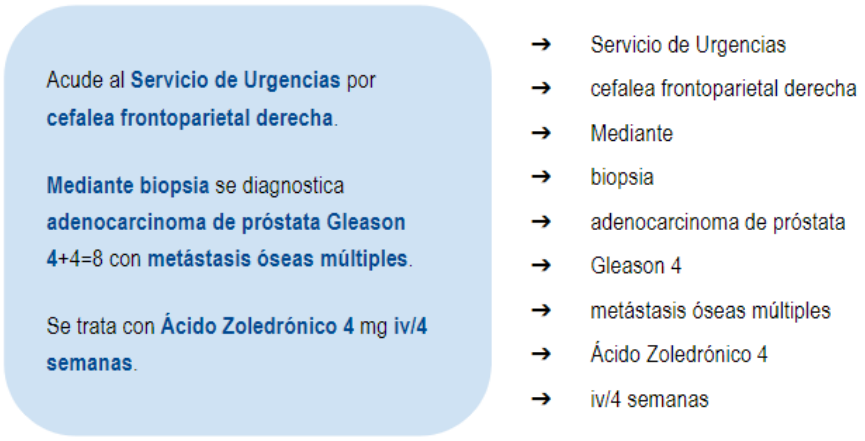

Acude al Servicio de Urgencias por cefalea frontoparietal derecha.
Mediante biopsia se diagnostica adenocarcinoma de próstata Gleason 4+4=8 con metástasis óseas múltiples.
Se trata con Ácido Zoledrónico 4 mg iv/4 semanas.
→ Servicio de Urgencias
→ cefalea frontoparietal derecha
→ Mediante
→ biopsia
→ adenocarcinoma de próstata
→ Gleason 4
→ metástasis óseas múltiples
→ Ácido Zoledrónico 4
→ iv/4 semanas

Figure 9. Example of functioning of the keyword extractor on a text of DisTEMIST. Own creation.

5.2.1.1. Keyword extractor pipelines

YAKE! (Yet Another Keyword Extractor): this unsupervised keyword extraction algorithm is based on statistical descriptors regarding word frequency and its relationship with the context [44]. It sets an approach that can be applied on single documents, without requiring a large corpus to function correctly. Additionally, it presents scalability and can therefore be used on different document sizes and domains. Since it is based on basic statistical descriptors, it is also a language-independent model, meaning that it can function correctly with a variety of languages without any further specification. The algorithm’s pipeline is based on five main steps:

1. Preprocessing and candidate term identification: text is split based on empty spaces and special characters, yielding a list of tokens that are considered as candidate keywords.
2. Five statistical features: for each of the tokens, five statistical features are computed:
    - Casing: count of the times a term is uppercase or represented as an acronym, which is generally related to a higher word importance
    - Term position: indicates the median position of the term within a sentence, with positions at the beginning being related to higher word significance.
    - Term frequency normalization: refers to the normalized frequency of the term in the whole document.
    - Term relatedness to context: refers to the co-occurrence of the term with other terms that constitute its context.
    - Term difference: counts of the times the term appears in different sentences.
3. Computing the term score: computing each individual term's score from the five computed statistical features. The equation for a sliding window of 3-grams is shown (1):

$$S(kw) = \frac{\prod_{w \in kw} S(w)}{KF(kw) * (1 + \sum_{w \in kw} S(w))} \qquad (1)$$

Equation 1. Individual term score for YAKE! Where S(kw) represents the score associated to a keyword kw, KF(kw) represents keyword frequency, and S(w) represents the score for the terms w that form a keyword kw.

4. Generating n-grams and computing keywords scores: all valid n-grams are generated, and their score is computed as the product of the scores of all the members but normalized to the total number of members.
5. Data deduplication and ranking: removal of similar words, while keeping the most relevant ones and sorting the keywords based on the computed scores.

TextRank: TextRank is a graph-based method for term extraction that uses graphs of related terms (also known as graphs of co-occurrence) [46]. It is based on PageRank, an algorithm used to compute the weight of different web pages, which are represented as nodes. TextRank applies this to text ranking by using equation (2):

$$S(V_i) = (1-d) + d * \sum_{j \in \ln(V_i)} \frac{1}{|Out(V_j)|} S(V_j) \qquad (2)$$

Equation 2. Weight of each node in the TextRank algorithm. Where $S(V_i)$ represents the score of a vertex V_i, d represents a damping factor between 0 and 1, $In(V_j)$ are the predecessor nodes to a vertex V_j, $Out(V_j)$ are the successor nodes to a vertex V_j, and $S(V_j)$ the score for that given vertex V_j.

This formula yields the weight of each node of the graph, which depends on the weight of the inboud (previous) nodes. It is applied in an iterative manner, by initializing all weights to one and iterate with the formula until the weights are optimized. The steps of the TextRank pipeline are:

1. Text tokenization and annotation with part of speech (PoS) tags: words are stored with specific PoS tagging, which indicates whether each word is a noun, verb, adjective or other. This has been done with spaCy for PoS tagging.

2. Word co-occurrence graph construction: words are represented in nodes, and they are connected by an undirected edge if they occur within a window of words in the text.
3. Graph ranking: all nodes are given a score based on their connections using the given formula and then they are ranked according to that score.
4. Top-scoring words selection: after sorting the words by score, the top ones are selected.
5. Keywords extraction: multiword keywords are formed from selected words.

RAKE (Rapid Automatic Keyword Extraction): RAKE constitutes a graph-based unsupervised method, which is based on the generation of a graph of related terms or co-occurrence to assess the importance of each term [45]. This algorithm operates on individual documents that can be of different types and with specific grammar conventions. This also means that RAKE is domain-independent as well as language-independent. The pipeline of steps that RAKE follows is shown:

1. Candidate keywords extraction: parsing the document to find content bearing words and phrases. This is carried out by first splitting the text to an array of words and then splitting this array to sequences of contiguous words using phrase delimiters and stopwords. Words within the same sequence are considered together as a candidate key.
2. Keyword co-occurrence graph construction: graph representation where nodes are words that appear interconnected and that are weighted based on their co-occurrences.
3. Word scoring: words are scored using three parameters:
    - Freq(w) or word frequency: number of appearances of a term in candidate words. It favors words that occur often independently of their co-occurrence.
    - Deg(w) or word degree: number of times that a word co-occurs with another one. It favors words that occur often and in long candidates.
    - Deg(w)/Freq(w) or ratio of the degree of frequency: the quotient of the two previous measures. It favors words in long candidates.
4. Candidate keywords total score: sum of the scores of the member words.
5. Adjoining keywords: adding stopwords to candidates to achieve some lexical coherence.
6. Keyword extraction: selection of the top T candidates based on their score. Generally, the best scoring third of the candidates is selected.

KeyBERT: KeyBERT is built as a Bidirectional Transformer Encoder model based on the semantic similarity of words [47]. Therefore, it uses BERT embeddings to transform words, phrases and documents to a vector representation. Its steps are the following:

1. Keyphrase extraction: keyphrases are extracted using a given keyphrase length and stopwords, as well as using PoS tags. When using a given keyphrase length, only n-grams of up to the desired length are extracted. On the other hand, when using PoS tags, a KeyphraseCountVectorizer object is used with a spacy pipeline in Spanish to extract keywords with the defined PoS patterns.
2. Vector representations: BERT embeddings obtained from pre-trained models are applied to generate a vector representation of the candidates. In this case, since the documents belong to the medical domain and contained medical keywords, the chosen PLM as base LM is the Spanish version of SapBERT. For further information on the training process of SapBERT, refer to the end of this subsection.

3. Similarity between candidates: cosine similarity is performed to assess which candidates are similar, and candidates with generally high similarities are assumed to be relevant. This performs well in high dimensionality, where similarity is interpreted as inverse to the distance in a high dimensional vector space. This step yields the keyphrases that individually represent the document and the other candidates more accurately.
4. Diversification of the result: select the keyphrases that best represent the document jointly, instead of individually. This has been achieved using Maximal Marginal Relevance (MMR), which is designed to minimize redundancy and maximize the diversity of the results. It does so by selecting first the best result and then iteratively selecting others that are both similar to the document and not similar to the already selected keyphrases. The diversity of this method can be tuned for different results.

For the models that require a pre-trained base LM, such as KeyBERT, the mainly used LMs have been SapBERT-based architectures. **SapBERT (Self-Alignment Pre-training for Biomedical Entity Representations)** is a LM that constitutes a pre-training scheme that self-aligns the representation space of biomedical entities [16]. The fact that SapBERT performs self-alignment as part of its training process allows it to better distinguish different types of biomedical entities within text. Furthermore, SapBERT is specifically designed for biomedical entity representation, being trained on top of PubMedBERT using contrastive learning [55]. PubMedBERT is a PLM that is specific for biomedical data and has been trained on large sets of biomedical literature from PubMed [15]. Contrastive learning is a learning strategy for LMs where first positive pairs of entities – from the same type – and negative pairs of entities – from different types – are generated. Then, the model is taught to push positive pairs together and pull negative pairs apart to ensure a better representation of different entity types. SapBERT trained on top of PubMedBERT has shown a clearly superior performance for biomedical term representation than PubMedBERT on its own and is considered SOTA for this task. An example of how SapBERT + PubMedBERT outperform PubMedBERT in the representation of biomedical entities in a vector space is shown in Figure 2 from section 2.1. General concepts. For KeyBERT and all other models requiring a LM as base model, either for keyword extraction or for other tasks, the Spanish version of SapBERT has been used. The Spanish version of SapBERT has been developed by NLP4BIA at the BSC and provides to pre-trained versions that have been trained slightly differently [55]. One of the versions - SapBERT with no parents - has been trained using as positive pairs only synonyms, while the other alternative - SapBERT with parents – has been trained using as positive pairs both synonyms and their parents.

5.2.2. Term categorization

The term categorization developed within TermExtractor has been implemented in both a supervised and unsupervised manner. The supervised multilabel classification is conducted with the SetFit classifier and the AutoModelForSequenceClassification, both from the SentenceTransformers library, while the unsupervised categorization is conducted with a K-Means Clustering algorithm. This section focuses on the pipelines of each of the implemented categorizers, as well as on their training and evaluation.

An example of the functioning of the term categorizers implemented in the library is shown in Figure 10. which showcases how the extracted keywords from Figure 9 are classified using the SetFit classifier with default parameters. As seen, the extracted keywords, even when incomplete, are generally correctly classified. It is also of note that the class NO_CATEGORY is used to discard those extracted terms that are not keywords or that do not correspond to any of the classes implemented in the classifier, as is the case for "Mediante". This provides an additional filter to the keyword extraction pipeline.

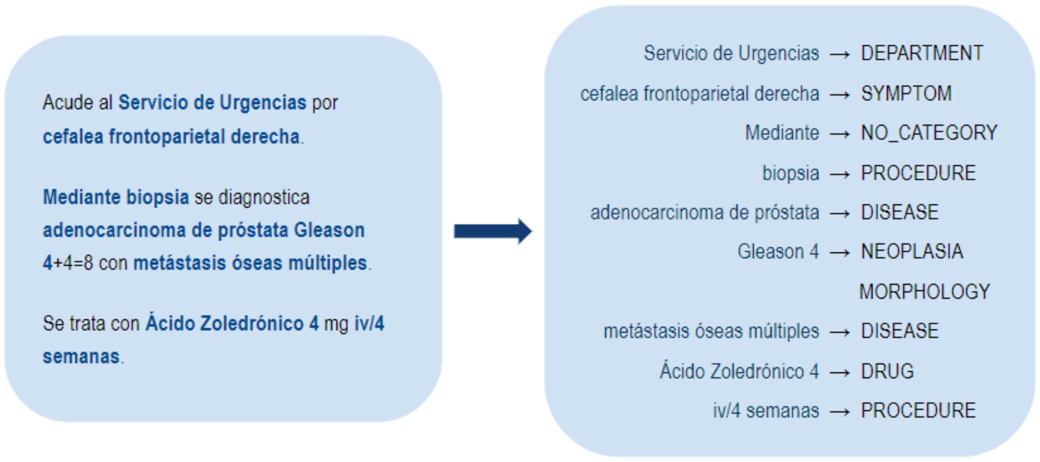

Acude al Servicio de Urgencias por cefalea frontoparietal derecha.
Mediante biopsia se diagnostica adenocarcinoma de próstata Gleason 4+4=8 con metástasis óseas múltiples.
Se trata con Ácido Zoledrónico 4 mg iv/4 semanas.
Servicio de Urgencias → DEPARTMENT
cefalea frontoparietal derecha → SYMPTOM
Mediante → NO_CATEGORY
biopsia → PROCEDURE
adenocarcinoma de próstata → DISEASE
Gleason 4 → NEOPLASIA MORPHOLOGY
metástasis óseas múltiples → DISEASE
Ácido Zoledrónico 4 → DRUG
iv/4 semanas → PROCEDURE

Figure 10. Example of functioning of the term classifier on the keywords from a DisTEMIST text. Own creation.

5.2.2.1. Term categorizer training pipelines

Unsupervised K-Means Clustering: this unsupervised clustering method is implemented through two main phases: the generation of the embeddings of the terms and the clustering of those embeddings.

1. Embeddings generation: BERT embeddings are generated using a pre-trained LM specific for biomedical terms in Spanish. In this case the Spanish SapBERT generated by the NLP4BIA group has been used[55]. This results in a vector representation of the terms in a high dimensional space that allows for a posterior clustering of the embeddings.
2. K-Means clustering: this unsupervised method uses a recursive algorithm to generate optimal clusters in which the different data points are distributed. To do this, it first places k centroids randomly within the data space. Then each data point is assigned to the cluster with the closest centroid and the centroids are recalculated based on the average of the data points assigned to each cluster. These last steps are repeated until the stopping criteria are met and the final clusters are generated.

Transformer's AutoModel for Sequence Classification: the Transformer's AutoModels are generic classes that are generally based on two main components - a pre-trained Transformer LM and a classification head specifically designed for the corresponding task [51]. In this case, the classification head has been specifically designed for a sequence classification task, which takes the encoded representation of the input and transforms it into a probability distribution over class labels. Its pipeline consists of the following steps:

1. Preprocessing: this includes any processing of the input text, such as tokenization and padding.
2. Embedding layer: this layer of the model leverages the pre-trained LM that is provided as a base model to generate the embeddings of the input sequences. In this case the used base LM is the Spanish version of the SapBERT model trained with no parents by NLP4BIA at the BSC [55].
3. Transformer Encoder Layers: these layers further process the embeddings to extract higher-level representations of the input text that capture meaning and context better.
4. Classification head: this classification head consists of fully connected layers with activation functions like softmax that take the output of the encoder layers and output a vector of probabilities, one for each class label. Since this model is implemented in a multilabel manner within KeyCARE, a decision threshold is set to select the valid labels based on the output probabilities.

SetFit: SetFit is also a model based on the Transformer architecture that uses a pre-trained LM for a sequence classification task [52]. However, it differs from the AutoModel because it includes an embedding fine-tuning phase before the classification head that allows it to function in a few-shot manner, therefore needing smaller sets of training data than a regular model. Its pipeline consists of the following phases:

1. Preprocessing and Embeddings generation: these work similarly to those from the AutoModel. The pre-trained base LM used for the generation of the embeddings is in this case also the Spanish version of SapBERT that has been trained with no parents [55]. In addition to that, sentence pairs are generated from the input data for the following phase.
2. Embedding Fine-tuning Phase: this step uses Contrastive Learning to fine-tune the embeddings that represent the input sequences. The pairs generated by the last step are either positive pairs – if the two terms belong to the same class – or negative pairs – if the two terms are of different classes. The model is fine-tuned to learn to push positive pairs together and pull negative pairs apart, increasing interclass variance and decreasing intraclass variance. This step allows SetFit to function in a few-shot manner, since with n terms it can generated n^2 term pairs on which the embedding is fine-tuned.
3. Classifier Training Phase: this step is also consistent with last step of the AutoModel's training, where a classification head is trained to produce a vector of probabilities for the class labels. In this case a Logistic Regression has been used for the classification head while keeping the model's multilabel nature through a decision threshold.

5.2.2.2. Training data

Both the SetFit classifier and the Transformer's AutoModel are supervised methods, and therefore they require a labelled training dataset. This data has been obtained from NER Gold Standard Corpora produced by the NLP4BIA group at the BSC as part of the PlanTL, which are designed for Named Entity Recognition tasks [40]. By taking the entities of each of these corpuses as terms and the name of the entities as their classes, a corpus of 98,484 classified terms has been generated. Of these, 73,863 terms of 21 different classes have been used for the training of the classifiers and 24,621 terms for their evaluation. The classes are the following:

- Procedure – from the MedProcNER corpus, for NER of medical procedures [56].
- Disease – from the DisTEMIST corpus, for NER of diseases [57].
- Symptom – from the SympTEMIST corpus, for NER of symptoms [58].
- Drug – from the DrugTEMIST corpus, for NER of pharmaceutical products [59].
- Neoplasia morphology – from the CanTEMIST corpus, for NER of cancer features [60].
- Human, Species – from the LivingNER corpus, for NER of living beings [61].
- Profession, Work situation, Activity – from the MeddoProf corpus, for NER of professional activities [62].
- Department, Community, Transportation, Language, FAC_GEN, GPE_GEN, GEO_GEN, FAC_NOM, GPE_NOM, GEO_NOM – from the MeddoPlace corpus, for NER of locations and associated entities [63]. The last six entity names refer to the generic and specific names of spaces, geographical and geological entities, respectively.
- NO_CATEGORY – manually generated class that includes terms that should not be classified as keywords, such as stopwords or words with little meaning. This differs from terms that should are not assigned any class, which might be keywords that do not correspond to any of the classes with which the classifier has been trained.

5.2.3. Classification of semantic relations

The classification of hierarchical relations implemented in RelExtractor has only been developed in a supervised manner, using both the AutoModelForSequenceClassification and the SetFit classifier. This section focuses on the implementation and training of these classifiers, as well as on their evaluation for the task of relation classification.

An example of the functioning of the relation classifiers implemented in the library is shown in Figure 11. This example shows how four classified keywords obtained from Figure 10 have been paired with four terminology concepts and the hierarchical relations among them have been extracted using the AutoModelForSequenceClassification with default parameters. As seen, in all four cases the model classified the hierarchical relation correctly, for instance understanding that "cefalea frontoparietal derecha" is a type of headache.

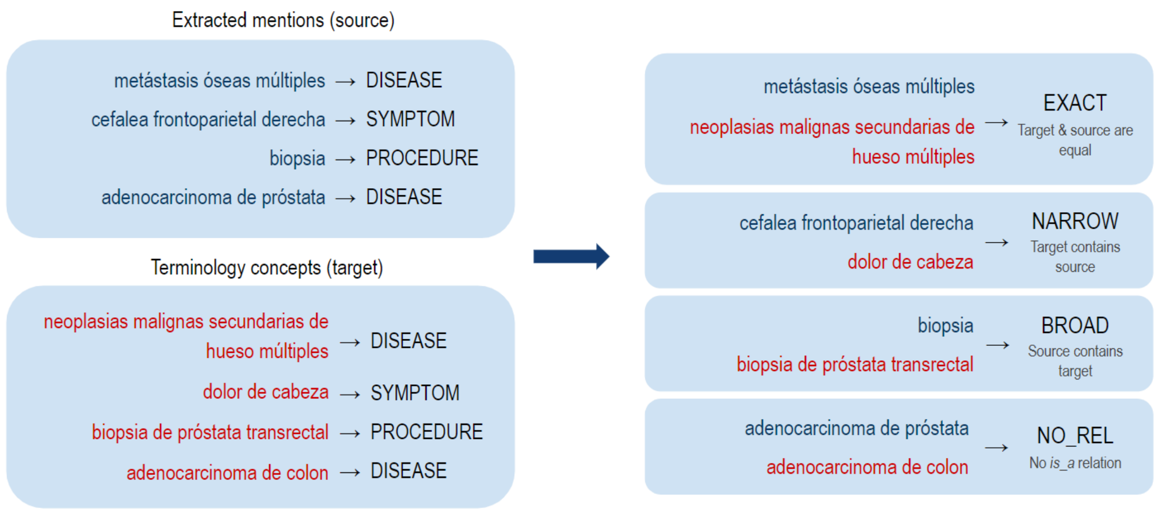

Extracted mentions (source)
metástasis óseas múltiples → DISEASE
cefalea frontoparietal derecha → SYMPTOM
biopsia → PROCEDURE
adenocarcinoma de próstata → DISEASE
Terminology concepts (target)
neoplasias malignas secundarias de hueso múltiples → DISEASE
dolor de cabeza → SYMPTOM
biopsia de próstata transrectal → PROCEDURE
adenocarcinoma de colon → DISEASE
metástasis óseas múltiples
neoplasias malignas secundarias de hueso múltiples → EXACT
Target & source are equal
cefalea frontoparietal derecha
dolor de cabeza → NARROW
Target contains source
biopsia
biopsia de próstata transrectal → BROAD
Source contains target
adenocarcinoma de próstata
adenocarcinoma de colon → NO_REL
No is_a relation

Figure 11. Example of functioning of the relations extractor on keywords extracted from a DisTEMIST document with terminology concepts. Own creation.

5.2.3.1. Relation classifier training pipelines

The models used for the relation classification pipeline are precisely the same ones that have been used form term classification previously: SetFit and the Transformer's AutoModel for Sequence Classification. Thus, for further information about the implementation of their pipelines refer to section 5.2.2. Term categorization. Additionally, for both models the Spanish version of SapBERT that has been trained with no parents has been used as base model. It is of note, however, that for this task the two models have been implemented for sentence pair classification, instead of a mere sequence classification. In the case of the Transformer's AutoModel this is an already built-in function, but for SetFit the source and target mentions have been introduced as a single sequence of the form "source mention </s> target mention".

5.2.3.2. Training data

Both SetFit and the Transformer's AutoModel are supervised models and therefore require a labelled training dataset. However, there was no available dataset containing hierarchical relations between medical terms in Spanish, so it was decided to generate such a dataset from structured terminologies as are SNOMED-CT and UMLS. Since the goal of the relations is to showcase if one of the terms includes the other, if they are equal, or if they have no hierarchical relation whatsoever, the SNOMED-CT hierarchical structure has been used to extract term pairs with such relations [3]. Nevertheless, the terms themselves have not been extracted from SNOMED-CT directly, but they have been extracted as they are written in UMLS through a mapping across terminologies [4]. This means that, while the extracted relations correspond to those from the SNOMED-CT tree structure, the terms themselves are written as they are in UMLS, since it constitutes a more standardized terminology. Figure 12 shows an example of how term pairs have been extracted from the SNOMED-CT structure. This example focuses on the target term "tumor maligno de pulmón", for which source terms with EXACT, BROAD, NARROW and NO_REL relations have been extracted:

- **EXACT** - synonyms from UMLS have been selected, as are "cancer de pulmón" and "tumor maligno de pulmón".
- **NARROW** - terms that are descendants of "tumor maligno de pulmón" have been selected, such as "adenocarcinoma de pulmón".
- **BROAD** - terms that are parents of "tumor maligno de pulmón" have been selected, such as "neoplasia maligna" or "tumor maligno de tórax".
- **NO_REL** - in this case, two different types of not hierarchically related terms have been considered: close NO_RELs - those from the same branch as the target term - and distant NO_RELs - those from other branches. The first ones are terms like "sarcoma de omoplato" which is close to "tumor maligno de pulmón" but is neither a descendant nor a parent term. The second ones are terms like "endoscopia de esófago", which belongs to a completely different branch within SNOMED-CT. For both types of NO_REL, a Jaccard Index has been used to compute similarity between the source and target terms, so as to ensure that both not related source terms that are lexically similar to the target term, and terms that are lexically diverse are well represented in the dataset.

Using an iterative algorithm to extract these relations from the SNOMED-CT database, over 10 million triplets of the form *source mention* – *target mention* – *relation type* have been generated. Both the SetFit classifier and the Transformer's AutoModel have been trained on increasingly bigger amounts of triplets and with different class distributions. For the moment, they both have been trained on up to 1.5 million triplets, with EXACT, NARROW and BROAD having 350,000 triplets each, and 450,000 triplets for NO_REL. Within the NO_REL class, 100,000 triplets were close NO_RELs and 350,000 triplets were distant NO_RELs. The training of both models has been performed with different training datasets and with the tunning of some parameters, such as the number of training epochs.

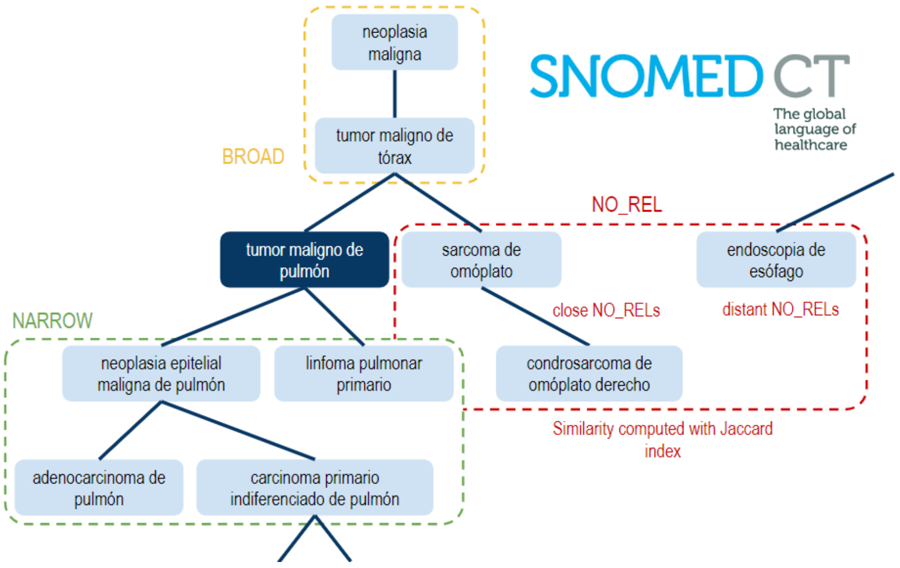

neoplasia maligna
tumor maligno de tórax
BROAD
SNOMED CT
The global language of healthcare
NO_REL
tumor maligno de pulmón
sarcoma de omóplato
endoscopia de esófago
close NO_RELs
distant NO_RELs
NARROW
neoplasia epitelial maligna de pulmón
linfoma pulmonar primario
condrosarcoma de omóplato derecho
Similarity computed with Jaccard index
adenocarcinoma de pulmón
carcinoma primario indiferenciado de pulmón

Figure 12. Scheme of the extracted relations from SNOMED-CT's hierarchical structure. Own creation.

5.3. Results and discussion

This section explains and discusses the obtained results for each of the steps of the implemented pipeline, as well as comment on the suitability of the implemented methods and models.

5.3.1. Keyword extraction

Each of the four extraction methods has been evaluated on two biomedical NER Gold Standard Corpora developed by the NLP4BIA group at the BSC. These are MedProcNER, a NER Gold Standard manually annotated corpus for medical procedures [56], and DisTEMIST, a NER Gold Standard manually annotated corpus for diseases [57]. Both these corpuses contain the same 750 documents, formed by medical records and other biomedical texts, with the corresponding manually annotated named entities. The precision, recall and f-score of the extraction methods has been assessed by comparing the extracted keywords from these corpora to the actual NER entities in them. The evaluation has both been performed using exact overlaps, meaning that the keywords must be exactly the same as the NER entities to be counted as correct, and relaxed overlaps, which means that as long as there is any overlap between the extracted mention and the NER entity it is counted as valid. This has been done using the MeddoPlace scoring script, a script accessible through GitHub that provides the code for evaluation with exact and relaxed overlaps [64]. For this evaluation, each of the four methods has been tested extracting keywords of up to one, three and five tokens of length, which have been evaluated on all the entities from the corresponding corpus. In addition to that, KeyBERT has been tested using a set of different PoS tags and values for *top_n* (the number of keywords to be extracted from each document) ranging from 5 to 50. KeyBERT has also been tested using different base pre-trained LMs, those being the versions of the Spanish SapBERT that have been pre-trained with parents and without them [55]. Finally, keyword extraction with combinations of extractors have also been assessed.

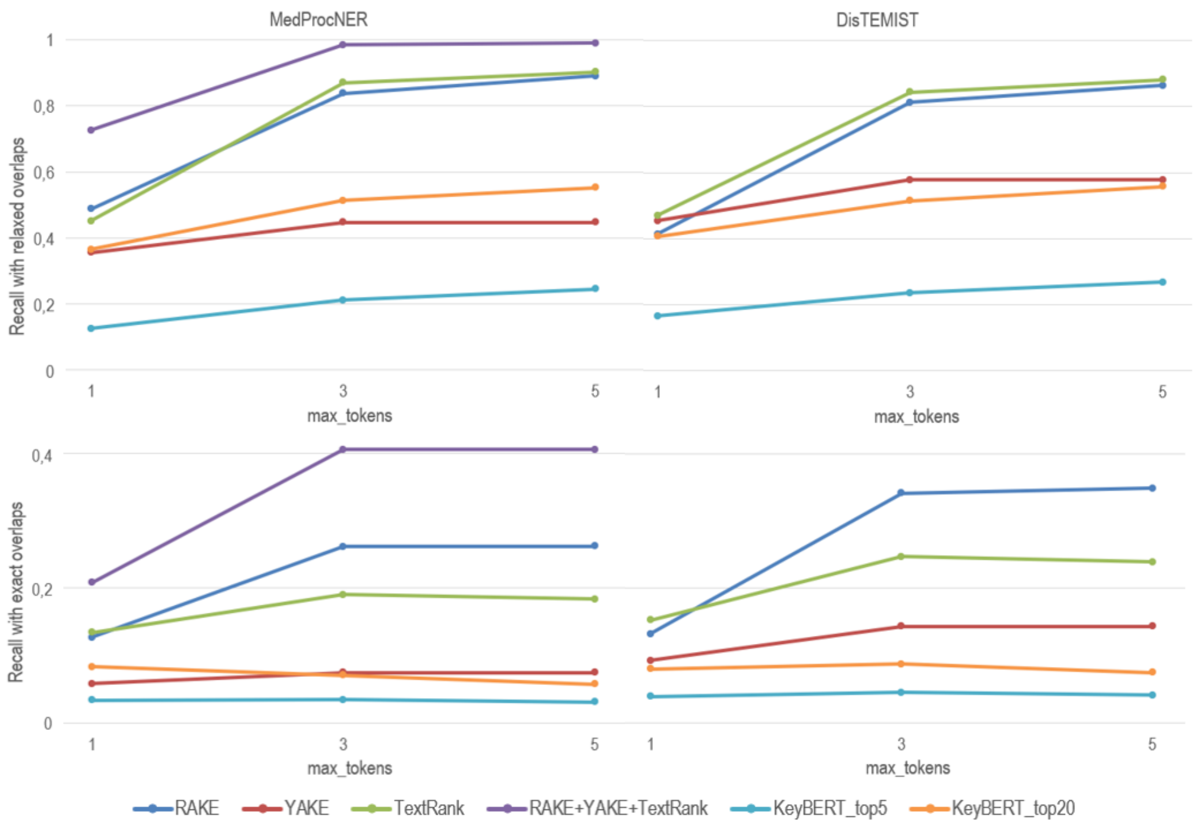


Figure 13. Graphs of the recalls of different keyword extractors against the maximum allowed number of tokens per keyword. On the upper left evaluated on MedProcNER with relaxed overlaps, on the lower left with exact overlaps. On the upper right evaluated on DisTEMIST with relaxed overlaps, on the lower right with exact overlaps. Own creation.

Figure 13 shows four graphs with the evaluation for different extractors on both MedProcNER and DisTEMIST, both with exact and relaxed overlaps. Each of the graphs shows the recall for the following extractors: RAKE, YAKE, TextRank, KeyBERT with *top_n*=5, and KeyBERT with *top_n*=20. Additionally, the evaluation on MedProcNER also shows the combination of the RAKE, YAKE and TextRank extractors at the same time. The precision, recall and f-scores for these extractors and for all the other ones that have been tested can be found in Tables 8 and 9 from section 12.2. Additional material. Only the recall is shown - both for exact and relaxed overlaps – because it is the main metric of interest in this evaluation. The implemented extraction methods are meant for the extraction any term that is considered relevant to the meaning of the document, which might include diseases, medical procedures, symptoms, drugs, as well as any other class. However, the MedProcNER and DisTEMIST corpuses only contain medical procedures and diseases, respectively. Therefore, a good outcome for this task is achieved when the extractors identify as many entities as possible from both MedProcNER and DisTEMIST – achieving a high recall -, regardless of their precision when detecting specifically diseases or medical procedures.

The precision is provided by the next step of the pipeline, in which the extracted keywords are classified. The evaluation of the whole pipeline, including both keyword extraction and their classification, can be seen in section 5.4.1. Unsupervised generation of NER candidates.

As seen in Figure 13, the methods that consistently provide a highest recall are RAKE and TextRank, with both achieving roughly a **90% recall** with relaxed overlaps. With exact overlaps, RAKE outperforms TextRank, obtaining up to **35%** in recall over **24%** of TextRank. However, an even higher recall is achieved when using RAKE, YAKE and TextRank together, achieving up to **99% recall** with relaxed overlaps and **41%** with exact ones. It must be noted that the combination of methods also generates significant noise along with the keywords and its precision is extremely low. It is also interesting to mention that YAKE obtains decent results when extracting keywords with one token of length but does not perform well with longer terms. On the other hand, KeyBERT generally performs well in terms of precision but fails at obtaining a high enough recall, even when extracting the top 20 best keywords. The version of KeyBERT that is based on PoS tags is not shown in Figure 13 due to its low performance, for it refer to Tables 8 and 9 from section 12.2. Additional material. Finally, it is also of note that there is a notable increase in recall when increasing the maximum number of tokens per keyword from 1 to 3, but not when increasing it to 5. This indicates that most entities have lengths of up to 3 tokens.

Considering the results of each of the extractors with keywords of different tokens of length, the default configuration for KeyCARE's keyword extraction pipeline sets TextRank as the default extractor and the parameter *max_tokens* at 4 tokens per keyword. Nevertheless, depending on the application or type of keywords for which it is intended, these parameters should be modified.

5.3.2. Term categorization

Both the SetFit classifier and the Transformer's AutoModel have been evaluated on a test dataset of 24,621 entities of the same source as their training data[4]. The training and evaluation have been performed with different sets of parameters, such as the number of epochs, whether the SapBERT base model has been pre-trained with or without parents, and whether the classification is performed in a multilabel manner or not. With each of these combinations, a classification report indicating the precision, recall and f-score for all classes has been generated, as well as a multilabel confusion matrix plotted as a heatmap. The precision measures the proportion of positive predictions that are truly correct, recall indicates the proportion of actual positive cases that were identified correctly, and f-score is combination of both metrics. The classification report also includes the precision, recall and f-score for the weighted average and the macro average (average considering all classes equally) of all the classes. Both SetFit and the AutoModel with different parameter combinations have obtained similar results, always attaining **average f-scores over 90%**. However, the optimal combination considering the model's training time and the obtained evaluation metrics is the SetFit classifier functioning in a multilabel manner, and with either of the Spanish SapBERT versions as a base model[5]. In addition to that,

[4] The K-Means clustering approach has not been evaluated because it does not generate clusters that can be approximated to the classes of the provided data. In any case, it underperforms vastly when compared to the term classifiers.

[5] The best scoring trained models for both SetFit and the AutoModel have been stored and uploaded to Huggingface, from where KeyCARE imports them. Access them from sources [68] and [69], respectively.

the optimal number of epochs to train this model has been set at three, since for longer trainings the model tends to overfit the training data. Table 1 shows the classification report for this combination of parameters and with Spanish SapBERT with no parents as base model. On the other hand, Figure 14 shows the multilabel confusion matrix heatmap for that same combination of parameters.

Table 1. Classification report of SetFit as a term classifier. Only showing the main medical classes.

Category	Precision	Recall	F1-score	Quantity
DISEASE	0.83	0.82	0.83	2598
DRUG	0.94	0.70	0.81	416
HUMAN	0.99	0.94	0.96	3150
NEOPLASIA MORPHOLOGY	0.92	0.96	0.94	3633
NO_CATEGORY	0.96	0.99	0.97	88
PROCEDURE	0.96	0.97	0.97	3619
SYMPTOM	0.91	0.88	0.90	3103
SPECIES	0.98	0.99	0.98	4251
Macro avg	0.84	0.82	0.83	24621
Weighted avg	0.93	0.93	0.93	24621

As seen in Table 1, there is a clear imbalance between classes both for training and evaluation (the trainset and testset contain roughly the same proportions of each class). Despite being trained on imbalanced data, both the AutoModel and SetFit have obtained **weighted averages for precision, recall and f-score of around 93%**. Table 1 only shows the scores for the main medical classes, which are generally well represented in the training data and have obtained promising f-scores as: 83% for diseases, 94% for drugs, 99% for human, 92% for neoplasia morphology, 96% for procedures, 91% for symptoms, and 98% for species. On the other hand, more underrepresented classes have still obtained decent f-scores, despite having less than a hundred occurrences in the training data. Some examples of this are activities with a 62% f-score and GEO_GEN with a 86% f-score, with the exception of languages, which has achieved a f-score of 0% in all scenarios. For the scores of these classes refer to Table 10 from section 12.2. Additional material. Finally, the NO_CATEGORY class has achieved a f-score of 97% despite also having a small amount of manually generated examples as training data. All of this speaks for SetFit's few-shot capacities, achieving impressive scores even for classes with very sparse datasets.

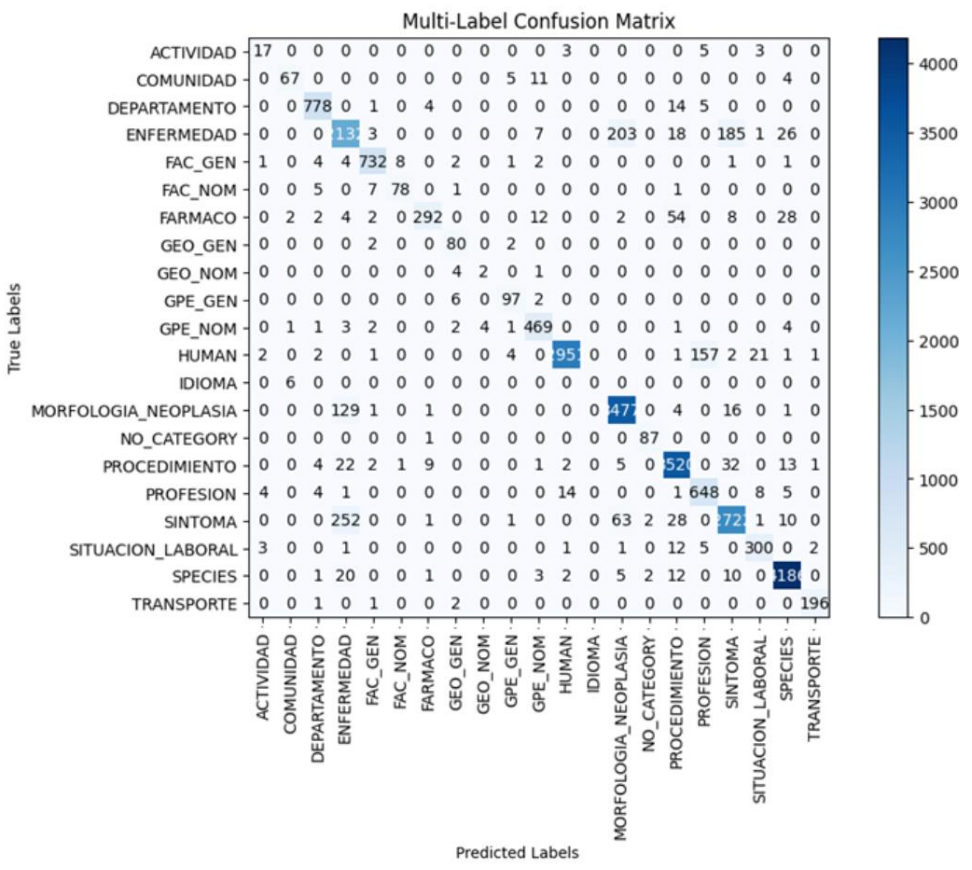


Figure 14. Multi-label Confusion Matrix of the classification performed by SetFit. Includes all classes. Own creation.

Figure 14 also shows a clear imbalance between classes and further confirms SetFit's impressive performance as a classifier even with sparse training data. The existing misclassified entities are mainly found among classes with a huge representation in the data, as are diseases. There is some misclassification of entities between diseases and symptoms, part of which might correspond to those entities that are polysemic and might refer to either a pathology or to a symptom of one. There is also misclassification of entities between diseases and the morphology of neoplasia, but this is mostly since neoplasia are diseases and thus can also be classified as such. This can be solved by adjusting the parameters of the classifier to enable a multilabel term classification that assigns both disease and symptom labels to the same entity. It can also be solved by training the classifier separately for diseases and neoplasia morphology, therefore allowing some overlap among the representation of the two classes. Finally, the evaluation code has been implemented to also count the occurrences within each class that have been assigned no label. The ones with most occurrences are work situation with 33, diseases and symptoms both with 23, and language with 17. For well-represented classes like diseases or symptoms this does not pose an issue. However, for an underrepresented class like languages it shows that the generated embeddings are not able to capture the nature of these entities well enough.

5.3.3. Relations classification

Both relation classifiers have been evaluated on a test dataset generated the same way as the train dataset and containing 200,000 triplets, equally distributed among classes. As mentioned, the training has been performed with increasingly large amounts of data and different sets of parameters, such as the number of epochs and whether the SapBERT base model has been pre-trained with or without parents. The number of epochs and the number of triplets for training have been two limiting factors due to their needs for extensive computational resources. With each of these combinations, a classification report indicating the precision, recall and f-score for all classes has been generated, as well as a multilabel confusion matrix plotted as a heatmap. The classification report also includes the precision, recall and f-score for the the macro average and the weighted average of all the classes. Both SetFit and the AutoModel with different parameter combinations have obtained similar results, attaining increasingly high scores when trained with more data. The best overall scores have been obtained for the Transformer's AutoModel with SapBERT with no parents as a base model, trained for two epochs and with 1.5 million triplets[6]. Table 2 shows the classification report in this scenario and Figure 15 the heatmap of its multilabel confusion matrix. In addition to that, the evaluation of other configurations of the model has also proven insightful: for instance, when training the model with 100,000 triplets and up to 10 epochs it has been seen that the smallest validation loss was obtained at 2 epochs. This was because the model overfitted the training data after seeing it more than two times, so after two epochs the validation loss increased. The same conclusion has been reached when training the model with 750,000 triplets and up to 4 epochs. However, the scores improved when the size of the training data increased to 750,000 triplets and improved even further when 1.5 million triplets were used. This shows that training over 2 epochs does not affect performance, while increasing the training data size and tunning the balance of classes can improve the performance greatly.

As seen in Table 2 and Figure 15, the evaluation of the relations classifier yields impressive scores on the dataset that has been generated the same way as the train dataset. All the classes are classified with precision, recall and f-score over 85%, and the **overall f-score using both macro and weighted averages is 93%**. The highest scoring class is NARROW, attaining a f-score of 98%. The class that achieves lowest scores is NO_REL, which still maintains an 88% f-score. These results confirm the model's strong performance on the provided data set.

Table 2. Classification report of the Transformer's AutoModel for the relation classification task. On the left, evaluated on the dataset generated from SNOMED-CT and UMLS. On the right, evaluated on the manually annotated data set.

Relation type	Precision	Recall	F1-score	Quantity
BROAD	0.89	0.93	0.91	50000
EXACT	0.95	0.94	0.94	50000
NARROW	0.97	0.98	0.98	50000
NO REL	0.89	0.86	0.88	50000
Macro avg	0.93	0.93	0.93	200000
Weighted avg	0.93	0.93	0.93	200000

Relation type	Precision	Recall	F1-score	Quantity
BROAD	0.50	0.82	0.62	620
EXACT	0.31	0.95	0.46	367
NARROW	0.49	0.64	0.55	358
NO_REL	0.98	0.64	0.78	3655
Macro Avg	0.57	0.76	0.60	5000
Weighted Avg	0.84	0.69	0.72	5000

[6] The best scoring trained models for both SetFit and the AutoModel have been stored and uploaded to Huggingface, from where KeyCARE imports them. Access them from sources [70] and [71], respectively.

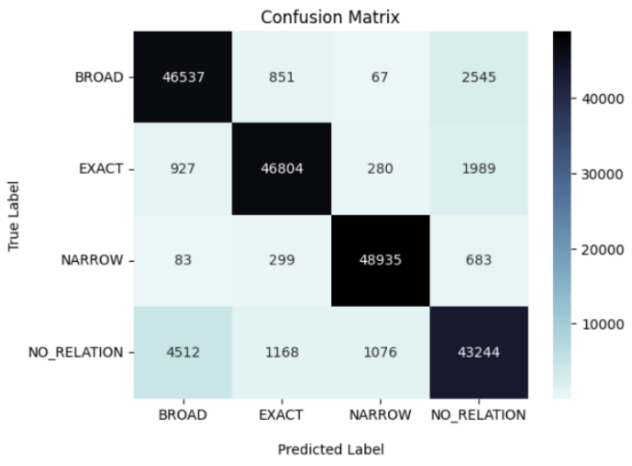


Figure 15. Confusion matrix of the Transformer's AutoModel for the relation classification task evaluated on the dataset generated from SNOMED-CT and UMLS. Own creation.

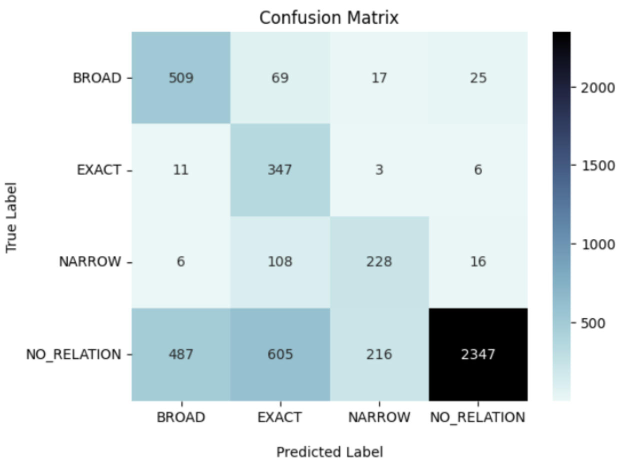


Figure 16. Confusion matrix of the Transformer's AutoModel for the relation classification task evaluated on the manually annotated dataset. Own creation.

In order to conduct a more complete and unbiased evaluation, the trained models for relation classification have also been evaluated on another dataset with data distributions unseen in the previous evaluation. This dataset has been manually annotated by NL4BIA at the BSC and contains a total of 5,000 triplets, of which 620 are BROAD, 367 are EXACT, 358 are NARROW and 3655 are NO_REL. This dataset has proven useful because it contains many NO_REL triplets that are difficult to identify. This is because most term pairs from this dataset have a very similar meaning even if they are not hierarchically related. An example of this is the term pair “abordaje de articulación de hombro" with “artrotomía de articulación de hombro”, which is not related hierarchically but the model predicts as EXACT. This has helped identify which relation types were not well represented in the training data and adapt it to better train the model. For instance, at the start of training the model only distant NO_RELs were being considered, and the evaluation on this dataset helped reach the conclusion that close NO_RELs were also needed.

The evaluation on this second test dataset of the AutoModel trained on 1.5 million triplets for 2 epochs has resulted in the second classification report shown in Table 2. Its results are also shown through the confusion matrix heatmap of Figure 16. As seen, the overall scores are significantly lower than in the previous evaluation, what indicates that this dataset likely contains term pairs that are not correctly represented in the first test dataset. The overall f-score using a weighted average reaches 72%, while its macro average is of only 60%. When referring to the scores of the specific classes, BROAD, EXACT and NARROW relations have very low recalls – between 31% and 50% - but obtain significantly higher precisions – between 64% and 95%. On the other hand, the NO_REL class obtains a precision of 98% but a recall of 64%. This is because, as seen in the confusion matrix, the NO_REL class is underpredicted, and many NO_RELS are predicted as BROAD, EXACT and NARROW. This observation is hugely magnified due to the severe class imbalance of this dataset, where NO_RELS have a representation of over ten times as much as some other classes.

Evaluation of results across both datasets demonstrates promising performance of the implemented relation classification model, even when applied to challenging data distributions. However, the second dataset has shown that those NO_RELS that have a high semantic similarity are often recognized as BROAD, EXACT or NARROW relations. It has also shown that term pairs including a very specific source term are usually classified as NARROW, even if there is no relation among the terms. Thus, training data that better represents these specific types of relations shall be generated to further train the model. This could be achieved by including NO_RELS among pairs of very specific terms with very general terms, as well as including NO_RELS that are semantically similar but belong to different branches within SNOMED-CT.

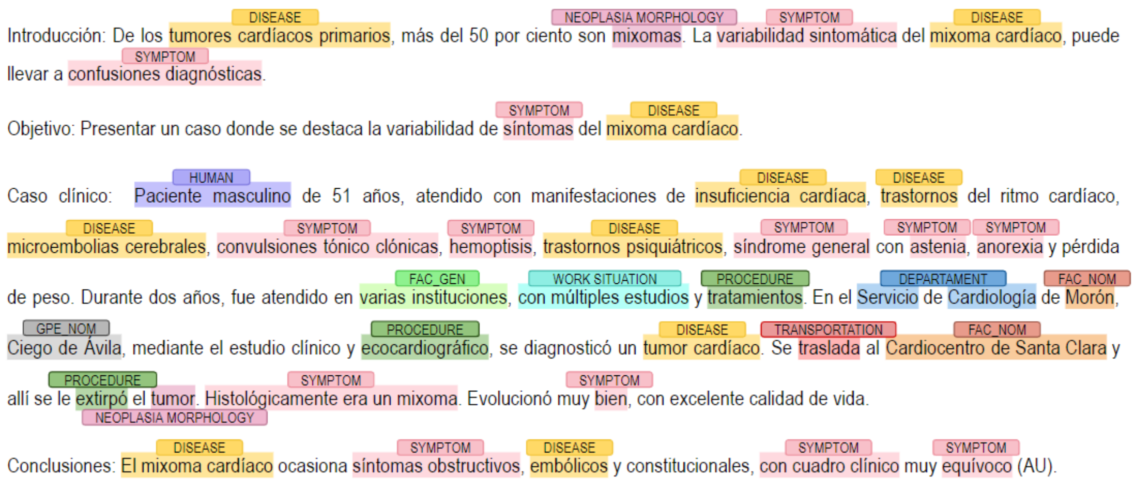
5.4. Evaluation of the use cases

This section discusses some applications of KeyCARE, some of which are presented as a future step and some of which have been implemented or evaluated on annotated data. This also helps assess how the library conducts the functions for which it has been designed, rather than just evaluating each of the models implemented in it. The use cases that are discussed in this section include the unsupervised generation of NER candidates, the generation of a knowledge graph, the analysis of term prevalence and co-occurrence against different parameters, and the creation of a semiautomatic tool for relations validation.

5.4.1. Unsupervised generation of NER candidates

The first use case for which KeyCARE is intended is the generation of NER candidates in an unsupervised manner. By leveraging the unsupervised keyword extractors and the SetFit classifier implemented in the TermExtractor class, this allows the user of the library to conduct an equivalent to Named Entity Recognition without the need for manually annotated training data. While the library is not intended as an alternative that performs as well as NER, which is the SOTA for this IE tasks, it provides a viable alternative in scenarios where there is no training data available. In fact, the use of KeyCARE for the generation of NER candidates that are validated manually can help create corpora for the training of NER systems in a much more efficient way than by manually annotating them from a document.

First, this use case has been executed on documents from Mesinesp (Medical Semantic Indexing in Spanish), a medical dataset with documents from different domains within medicine [65]. This has produced examples of the extraction of classified keywords for documents of specific domains, as the one shown in Figure 17, which corresponds to a document of Mesinesp of the


Introducción: De los tumores cardíacos primarios, más del 50 por ciento son mixomas. La variabilidad sintomática del mixoma cardíaco, puede llevar a confusiones diagnósticas.

Objetivo: Presentar un caso donde se destaca la variabilidad de síntomas del mixoma cardíaco.

Caso clínico: Paciente masculino de 51 años, atendido con manifestaciones de insuficiencia cardíaca, trastornos del ritmo cardíaco, microembolias cerebrales, convulsiones tónico clónicas, hemoptisis, trastornos psiquiátricos, síndrome general con astenia, anorexia y pérdida de peso. Durante dos años, fue atendido en varias instituciones, con múltiples estudios y tratamientos. En el Servicio de Cardiología de Morón, Ciego de Ávila, mediante el estudio clínico y ecocardiográfico, se diagnosticó un tumor cardíaco. Se traslada al Cardiocentro de Santa Clara y allí se le extirpó el tumor. Histológicamente era un mixoma. Evolucionó muy bien, con excelente calidad de vida.

Conclusiones: El mixoma cardíaco ocasiona síntomas obstructivos, embólicos y constitucionales, con cuadro clínico muy equívoco (AU).

Figure 15. Example of functioning of the pipeline for the unsupervised generation of NER candidates on a document of the domain of cardiology of Mesinesp. Own creation.

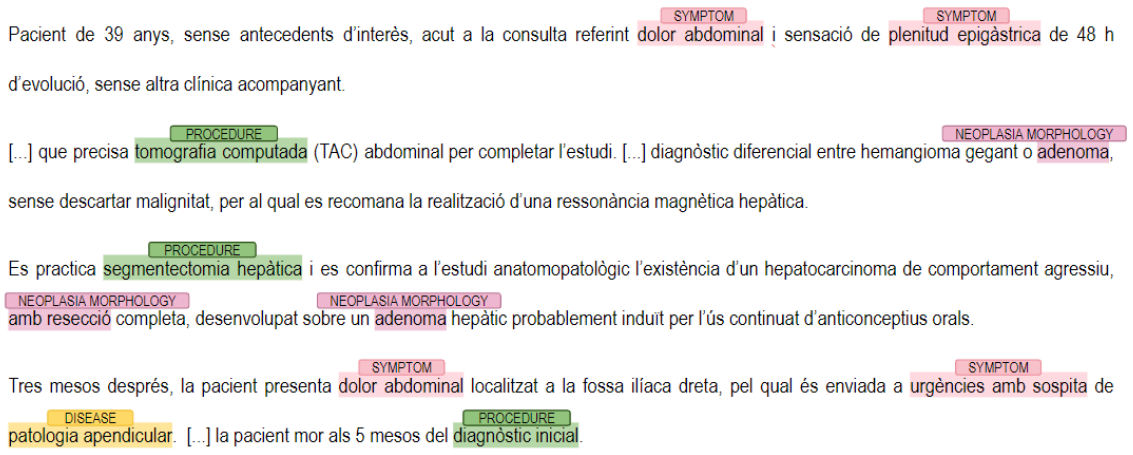
domain of cardiology. As seen, entities of different classes are extracted and classified, including diseases, symptoms, locations, and procedures among others. Most entities are correctly classified, especially those of classes with many occurrences, such as TRANSPORTATION, FAC_NOM or HUMAN. It is also of note that the extraction of some main classes such as diseases or the morphology of a neoplasia also perform great, with just a few diseases being misrecognized, such as the word "trastornos" on its own. On the other hand, the class SYMPTOM performs worse on a general basis, extracting entities like "equívoco", "bien" or the word "síntomas". While it does recognize most symptoms, achieving a high recall, it also generates many false positives.

In addition to that, this application of KeyCARE has been evaluated on the same NER Gold Standard Corpora as the keyword extractors, MedProcNER and DisTEMIST. Thus, the evaluation has been performed for the classes PROCEDURE and DISEASE, obtaining the precision, recall and f-score both with exact and with relaxed overlaps. The best performing configuration of KeyCARE for these corpuses has been the selection of RAKE as an extractor with up to 5 tokens per keyword and SetFit as a classifier with default parameters. As shown in Table 3, this configuration achieves **f-scores of up to 65%** with relaxed overlap and of up to **30%** with exact overlap for DisTEMIST, and of **around 60%** with relaxed overlaps and **20%** with exact overlaps for MedProcNER. According to the literature, supervised NER systems that have been evaluated on the MedProcNER corpus by different teams have obtained f-scores ranging from 20% up to 80% approximately [56]. Thus, while the obtained results for the proposed approach are not comparable to the scores obtained by a supervised NER model, KeyCARE still obtains decent scores that make it fit for applications where it is implemented with a posterior manual revision. On the other hand, if the objective of the application is to recognize all possible entities, even if that means reducing the precision, it is possible to define a different configuration to increase the overall recall. An example of this is also shown in Table 3, including RAKE, TextRank and KeyBERT with up to three tokens per keyword as extractors, and SetFit with default parameters as the classifier. In this case, up to 94% of the keywords from MedProcNER are at least partially detected (with relaxed overlaps) and nearly the 40% are exactly detected.

Table 3. Classification report of the pipeline for the unsupervised generation of NER candidates. Includes the evaluation of the combinations with the best f-score and best recall when evaluated on MedProcNER and DisTEMIST with exact and relaxed overlaps. Own creation.

		MedProcNER			DisTEMIST		
		precision	recall	f1-score	precision	recall	f1-score
RAKE with up to 5 tokens	Relaxed overlap	50.26%	75.97%	60.50%	64.32%	66.84%	65.56%
	Exact overlap	17.23%	26.06%	20.75%	30.16%	31.35%	30.74%
RAKE + TextRank + KeyBERT with up to 3 tokens	Relaxed overlap	28.61%	94.26%	43.90%	38.45%	86.27%	53.20%
	Exact overlap	12.03%	39.63%	18.46%	19.20%	43.08%	26.56%

Finally, in order to provide a preliminary overview of how KeyCARE might function for low-resource languages when there is no training data whatsoever, its pipeline for keyword extraction and categorization has been tested on medical records in Catalan. This has been achieved using the default configuration of TermExtractor, that is TextRank as the extractor and SetFit as classifier. Note that SetFit has been trained on mentions in Spanish and with a base model in Spanish. The obtained result for a specific medical record obtained from the Primary Care Bulletin of Catalonia [66] is shown in Figure 18. As seen, the extracted keywords in this figure are much less numerous than those from Figure 17. That is because the keyword extraction in Catalan seemed to produce more irrelevant and meaningless keywords than in Spanish, and thus the keywords have been filtered using the *top_n* parameter to extract only the ten best keywords. This way the results present a high enough precision, extracting mostly correct keywords for the four extracted classes. However, this has also meant a decrease in the pipeline's recall, where many more keywords from the document have been missed than in Spanish. To achieve a comparable performance to that in Spanish, a domain-specific base model in Catalan should be produced and the classifier should be trained on entities in Catalan, which can be retrieved from the future SNOMED-CT expansion in Catalan [6].


Pacient de 39 anys, sense antecedents d'interès, acut a la consulta referint dolor abdominal i sensació de plenitud epigàstrica de 48 h d'evolució, sense altra clínica acompanyant.

[...] que precisa tomografia computada (TAC) abdominal per completar l'estudi. [...] diagnòstic diferencial entre hemangioma gegant o adenoma, sense descartar malignitat, per al qual es recomana la realització d'una ressonància magnètica hepàtica.

Es practica segmentectomia hepàtica i es confirma a l'estudi anatomopatològic l'existència d'un hepatocarcinoma de comportament agressiu, amb resecció completa, desenvolupat sobre un adenoma hepàtic probablement induït per l'ús continuat d'anticonceptius orals.

Tres mesos després, la pacient presenta dolor abdominal localitzat a la fossa ilíaca dreta, pel qual és enviada a urgències amb sospita de patologia apendicular. [...] la pacient mor als 5 mesos del diagnòstic inicial.

Figure 16. Example of functioning of the pipeline for the unsupervised generation of NER candidates on a text in Catalan. Own creation.

In order to produce more accurate results than the ones shown in Figure 18, the whole pipeline would have to be implemented in Catalan. This would entail using either a base model for the biomedical domain that understands Catalan and training the SetFit classifier with labelled data in Catalan. Regarding the base model, since there is no Catalan version of SapBERT, the multilingual version of SapBERT could be used instead. Regarding the training data, the classifier could be trained on data from the Catalan version of SNOMED-CT, which is under development and set to be released soon [6]. Another option would be to train the model on artificially generated data from Spanish terms with machine translation. As an alternative, since SetFit functions in a few-shot manner, it could be trained on manually produced small sets of data.

5.4.2. Analysis of term prevalence and co-occurrence against different parameters

This second use case of KeyCARE consists of its use for the extraction of classified keywords from medical documents corpora to analyze their prevalence against different parameters and their co-occurrence with other terms. This analysis has been conducted by running the TermExtractor class of KeyCARE with default parameters over the Mesinesp Corpus, as in the last use case, extracting the keywords from 249,462 biomedical scientific papers. In addition to that, the extracted keywords have been postprocessed using a filtering based on PoS tags, sentence similarity, and a clustering algorithm that have allowed the identification of the most prevalent keywords and group them with similar terms based on their meaning. This has resulted in a reduction of the number of extracted keywords from roughly 3 million to 1,000 keywords, on which further analysis has been conducted.

An analysis that can be conducted with the produced data is he **study the prevalence of different terms against the publication year** of the research papers from which they have been extracted. Figure 19 shows an example of this for the keywords “coronavirus”, “enfermedad pulmonar obstructiva crónica”, and “embarazo adolescente” over the time period ranging from 1981 to 2020. As can be seen, “enfermedad pulmonar obstructiva crónica” has a steadily increasing prevalence, which is probably due to the general increase in biomedical research over the last decades. On the other hand, the term “coronavirus” is barely mentioned before 2020. a year in which it appears 860 times as a result of the COVID-19 pandemic. Finally, the term “embarazo adolescente” has a significative increase in prevalence starting around 2010. which might be due to the decriminalization of abortion in Spain in 2010 [67]. While these are only three simple examples, KeyCARE has a clear application in the study of term prevalence that can help identify medical neologisms and track the use of different procedures or drugs over time.

Another analysis that can be conducted with the extracted data is the study of term co-occurrence, in other words, studying how often different terms appear in the same document. Figure 20 shows an example of a **knowledge graph**[7] created by computing the co-occurrences of a subset of 200 extracted keywords from the Mesinesp corpus. This graph shows only the relations among keywords that have co-occurred at least 50 times in the whole corpus, thus discarding trivial or random co-occurrences. The generation of this type of graph can help identify

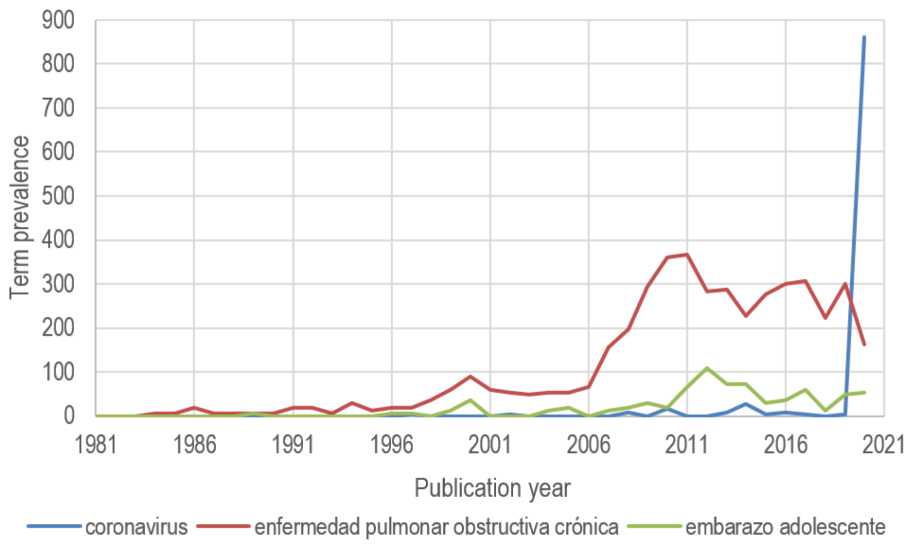


Figure 17. Evolution of the prevalence of "coronavirus", "enfermedad pulmonar obstructiva crónica", and “embarazo adolescente” over time in the scientific literature contained in the Mesinesp corpus. Own creation.

[7] The asterisks indicate that a keyword represents a cluster of terms similar to it, not just the term itself.

an existing correlation among medical terms, what provides insight on the relations among terms and is useful for terminology enrichment. An example from the figure is the relation of “sobrepeso” as a symptom with the diseases “diabetes gestacional” and “diabetes mellitus”, with which it co-occurred 84 and 103 times, respectively. Another noteworthy example is the co-occurrence of “fiebre” as a symptom with other symptoms such as “disnea”, “tos”, “vómitos”, “dificultad respiratoria”, which indicates that they usually occur together. Note that the keywords have been classified by the SetFit classifier, which does not include yet classes for body parts, risk factors or other entities, all of which might be misclassified.

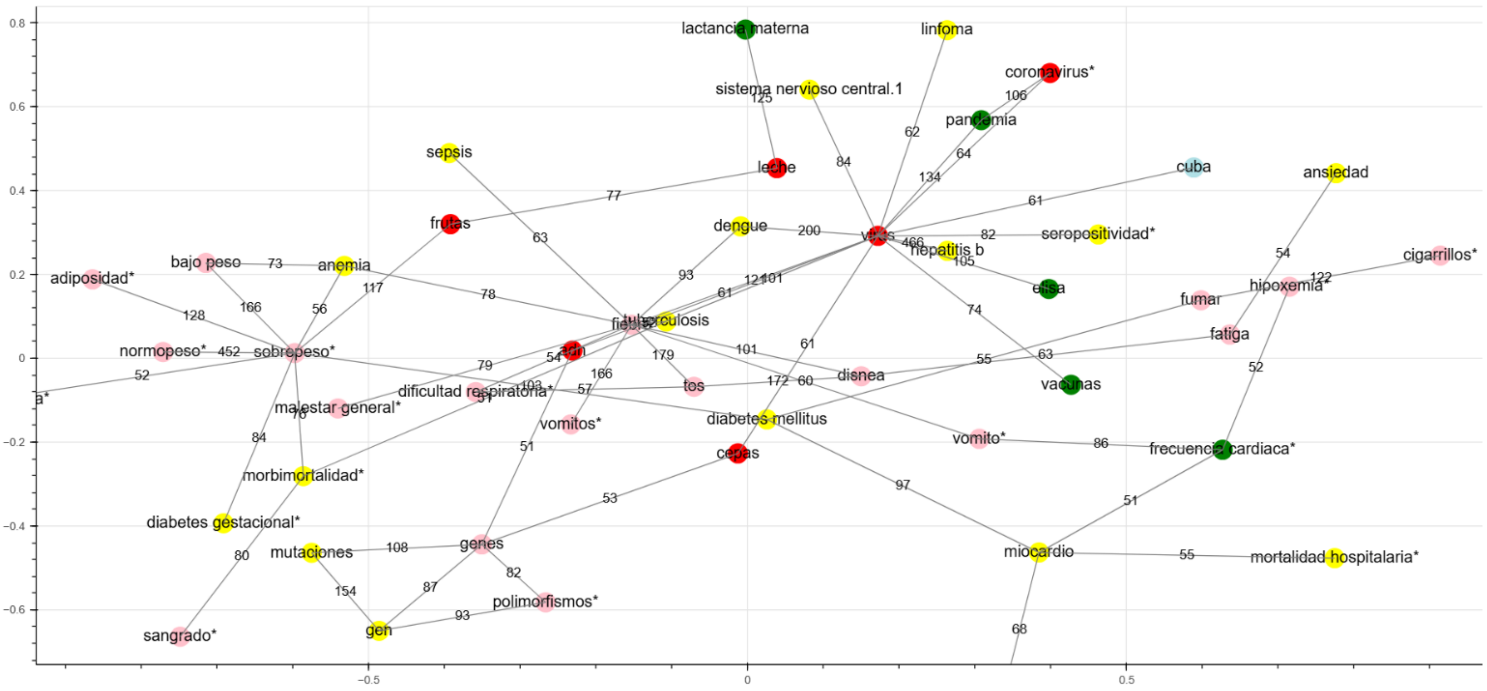


Figure 18. Graph of co-occurrence of extracted keywords from Mesinesp. The colors indicate the classes associated to each keyword: pink for symptoms, yellow for diseases, green for procedures, red for species, blue for GPE_NOM (location). Own creation.

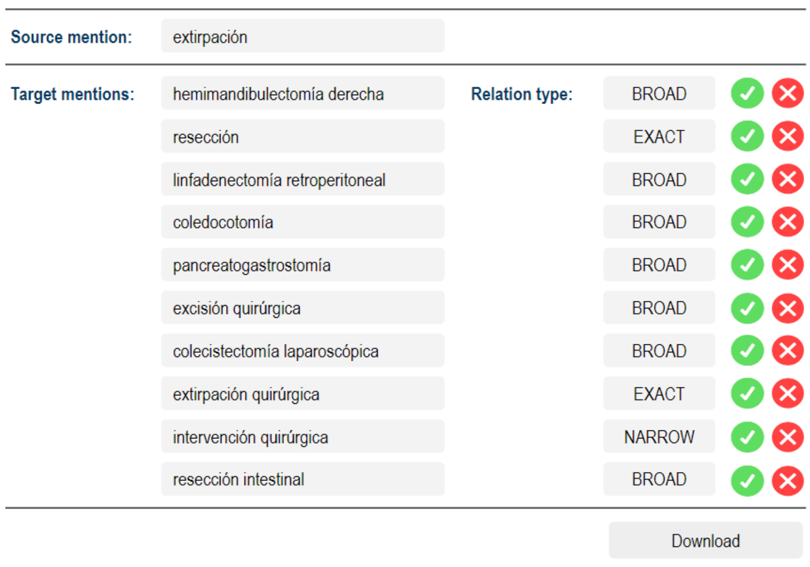
5.4.3. Semiautomatic tool for relations validation

The last use case considered in this project is the creation of a semiautomatic tool to generate candidate relations between terms that would need to be validated manually. Instead of having a field expert define terminology branches or mappings from one terminology to another from scratch, this would allow to generate to generate candidates for those relations that could be validated much more easily. Not only would this allow for a swifter terminology enrichment, but also for a more comprehensive one and with terms from real medical records. For instance, consider that the SNOMED-CT structure of the term “extirpación” is to be enriched. The pipeline for this use case would be as follows:

1. Use TermExtractor to extract all the classified keywords from a corpus of medical records or biomedical literature.
2. Since “extirpación” describes a medical procedure, filter medical procedures from the extracted keywords from their given label.
3. Once only keywords that refer to medical procedures remain, introduce the list of procedure keywords along with the term “extirpación” into the RelExtractor object.

4. From the extracted relations filter out the NO_RELS to keep only relevant relations.
5. Manually validate these relations through a user interface that accelerates the process by allowing the user to indicate whether a relation is valid or not.

Although this use case has not yet been implemented with a full interface, it has been tested on the part of the MedProcNER Corpus previously used for testing the term classifiers. Figure 21 shows 10 extracted relations for the term “extirpación” from the documents of this corpus as they would be shown in an interface. Note that following the described pipeline, up to 59 relations have been extracted for the term “extirpación” from a Corpus of 250 documents, so the shown relationships in the figure are only part of those extracted. Among the 59 extracted relations, most have been correctly identified, as the ones shown in the table, but there are also incorrectly extracted relations such as “márgenes”, “facoemulsificación” or “DP” (peritoneal dialysis), which have been identified as narrow concepts for “extirpación”. It is the hope that with further training of the relations classifier with data that better represents real-life scenarios these errors will be mostly avoided. In any case, since this use case is intended as a tool that requires manual validation, the performance of the pipeline is still remarkable.



Source mention:	extirpación			
Target mentions:	hemimandibulectomía derecha	**Relation type:**	BROAD	✓ ✗
	resección		EXACT	✓ ✗
	linfadenectomía retroperitoneal		BROAD	✓ ✗
	coledocotomía		BROAD	✓ ✗
	pancreatogastrostomía		BROAD	✓ ✗
	excisión quirúrgica		BROAD	✓ ✗
	colecistectomía laparoscópica		BROAD	✓ ✗
	extirpación quirúrgica		EXACT	✓ ✗
	intervención quirúrgica		NARROW	✓ ✗
	resección intestinal		BROAD	✓ ✗

Figure 19. Example of functioning of the pipeline for the semiautomatic relations validation tool, showing extracted relations from the MedProcNER corpus on the term "extirpación". Own creation.

5.5. Publication of the library

This section refers to the details of the publication of the library to online platforms that enable it to be imported to any script, as well as to the documentation and conditions accompanying this process. This means that this section also includes the documentation that has been uploaded within the library, as well as the plan for the library’s maintenance and updates in a near future.

The KeyCARE library has been uploaded to both GitHub and Pypi, accessible through references [53] and [54], respectively. Through GitHub the folder structure and the implementation of each of the classes can be viewed, and PyPI allows it to be imported and run over documents with some simple commands as shown below. Note that before importing KeyCARE to your script, the installation of KeyCARE with the command “pip install keycare” is required.

```
# import the two main classes

from keycare.TermExtractor import TermExtractor
from keycare RelExtractor import RelExtractor

# extract classified keywords from a text

text = “...”
termextractor = TermExtractor()
termextractor(text)
print(termextractor.keywords)

# extract hierarchical relations among terms

terms1 = []
terms2 = []
relextractor = RelExtractor()
relextractor(terms1, terms2)
print(relextractor.relations)
```

For the publication of the library to PyPI and GitHub as a functional module that can be imported from elsewhere, a specific set of configuration files has been produced. These include:

- **.env_keycare** (hidden): this file stores sensitive variables of the virtual environment using a secret management tool.
- **.git** (hidden): this folder stores Git version control information for the project.
- **.gitignore** (hidden): this file specifies files that Git ignores when committing changes.
- **LICENSE**: this file contains the license agreement for the code.
- **package-lock.json**: this file is generated by a package manager and lists the exact versions of dependencies used in the project.
- **package.json**: this file contains metadata about KeyCARE, including its name, version, dependencies, and scripts.
- **pyproject.toml**: this file has been used to configure various development tools for KeyCARE, including build tools and testing frameworks. It is the file that indicates the project’s structure and dependencies to PyPI, allowing it to function as a module.
- **README.md**: this file serves as a project introduction, containing a brief description of the project, installation instructions, usage examples, and other relevant information.
- **requirements.txt**: this file lists the Python dependencies required for KeyCARE.
- **nbs/ folder**: This folder contains Jupyter notebooks mainly showing how to use the main functionalities of KeyCARE as tutorials.
- **data/ folder**: this folder stores text files and toy data that are used as performance examples of KeyCARE such as those from the tutorials.
- **dist/ folder**: This folder contains built distribution files of the library, which are necessary for the correct upload of the module to PyPI.
- **src/keycare/ folder**: this folder contains all the source Python files of the KeyCARE library in the structure defined in previous sections.
- **www/ folder**: this folder contains the logo of KeyCARE, also shown in Figure 22.

Figure 20. Official logo of the KeyCARE library. Own creation.

5.5.1. Documentation

The KeyCARE library has been implemented as a framework that aims to be as understandable as possible. For that, all the code for class and function definition has been commented using a set notation. With that notation, each function and static method that is defined within a class of the library has been commented using a docstring that includes the overall description of the function; the name, type, and description of all the input parameters; and the returns or prints that the function generates.

In addition to that, two self-explanatory tutorials on the use of the main classes TermExtractor and RelExtractor have been generated. These tutorials can be accessed at the /nbs folder of the GitHub page of KeyCARE [53]. It is recommended to **carefully read the tutorials to better understand all the functionalities implemented in the KeyCARE library**. In addition to that, for a better understanding of the trained models that are implemented in KeyCARE, they have been uploaded to Huggingface and are accessible through the following links: SetFit for classification [68], the AutoModel for classification [69], SetFit for relation classification [70], and the AutoModel for relation classification [71].

5.5.2. Maintenance and updates

As mentioned, this Project is located within the mark of TeresIA a much bigger and more funded project. Thus, KeyCARE will continue to be developed at least for the duration of TeresIA. The updates on KeyCARE will mainly correspond to adjusts in dependencies, should those be updated and cause compatibility issues, and to the enhancement of the currently implemented methods as well as the implementation of more methods. As commented in section 10. Conclusions and future steps, there are already some planned updates that aim to enhance the overall performance of the library. For instance, more complex keyword extractors shall be implemented as alternatives to RAKE, YAKE, TextRank and KeyBERT. The implementation of AttentionRank is already in course, with the goal of achieving a more accurate keyword extraction process. In addition to that, the term classification model shall soon be trained with data including more categories, such as body parts or risk factors, to better represent extracted keywords. Finally, the model for relations classification will also be trained with more triplets and a training set that better represents corner cases that the current model is still not able to detect. In addition to all of that, it is the objective of the author to implement KeyCARE in a multilingual manner, so that it is able to function with languages different than Spanish. Further detail to these updates is given in the section 10. Conclusions and future steps.

6. Execution schedule

This section focuses on the elaboration of the diagrams necessary for the correct planning of the execution schedule of the project over time. This includes the definition of the main steps of the project, the definition of the work packages and the tasks that compose them, and the planning of these tasks over a timeline.

6.1. Phases and Milestones

The first step of the schedule planning is the development of the Phases and Milestones Table, where the main phases of the project are defined, and a due date is set for each of them. This table is shown in Table 11 from section 12.2. Additional material, which includes the description of each phase, its responsible, its supervisor or evaluator, its due date, and its deliverables. As seen in this table, the project has been divided into nine main tasks: its planning, a basic formation in NLP and Sentence Transformers, the implementation of each of the three main parts of the library, the publication of the library, the evaluation of the use cases, the development of this document, and its latter presentation.

6.2. Work Breakdown Structure (WBS)

Once the main phases of the project have been defined, the Work Breakdown Structure (WBS) is used to structure the project into the main work packages, each of which includes a set of tasks to be performed. The WBS can be seen in Figure 23. In this case, the main work packages of the project are the following: project management, basic NLP formation, Keyword Extractors implementation, Term Categorizers implementation, Relation Classifiers implementation, publication of the library, and use cases evaluation. Note that these do not correspond exactly to those defined in Table 11, since the planification of the project and the elaboration of the written document and the presentation have been grouped into the project management work package. Also note that the WBS diagram includes the tasks developed within each of these work packages, but not their details nor their deliverables. This is all shown as part of the WBS dictionary, which can be found in Table 12 from section 12.2. Additional material.

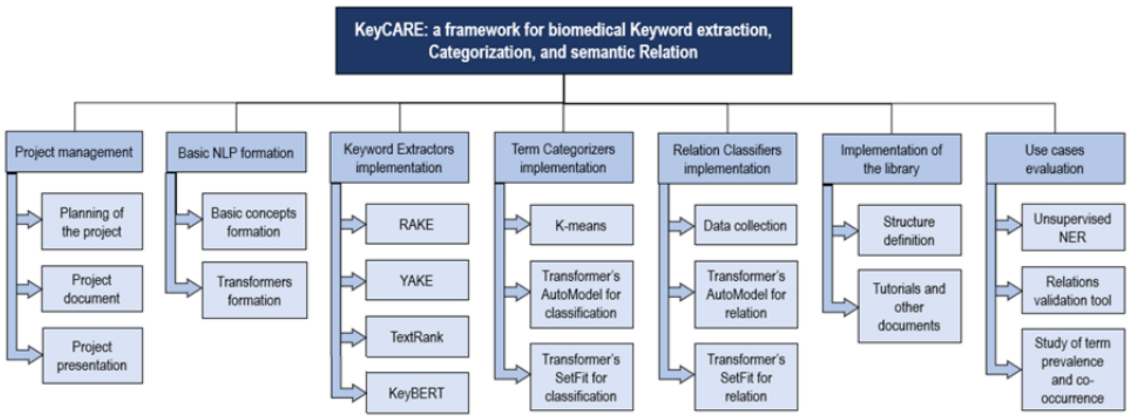

KeyCARE: a framework for biomedical Keyword extraction, Categorization, and semantic Relation
Project management
Planning of the project
Project document
Project presentation
Basic NLP formation
Basic concepts formation
Transformers formation
Keyword Extractors implementation
RAKE
YAKE
TextRank
KeyBERT
Term Categorizers implementation
K-means
Transformer's AutoModel for classification
Transformer's SetFit for classification
Relation Classifiers implementation
Data collection
Transformer's AutoModel for relation
Transformer's SetFit for relation
Implementation of the library
Structure definition
Tutorials and other documents
Use cases evaluation
Unsupervised NER
Relations validation tool
Study of term prevalence and co-occurrence

Figure 21. WBS diagram of the project showing the main tasks into which it has been divided. Own creation.

6.3. PERT-CPM diagram

After defining all the tasks that take place within the project, a Process Evaluation Review Technique - Critical Path Method (PERT-CPM) diagram can be generated. This diagram uses the dependencies among tasks to estimate the timeline for the project completion and find a critical path for on-time completion. To do this, first the precedencies among the different tasks have been defined as shown in Table 4. With the precedencies already defined, the PERT diagram is shown in Figure 24. This diagram is formed by arrows, representing the activities that have been developed, and nodes, each showing the time at which the node can be reached at earliest and at latest. It also shows the Critical Path in red, which is formed by those activities that would delay the whole project if delayed, known as critical activities. The activities in black, flexible activities, are those that can be delayed without affecting the duration of the project. Thus, diagram enables the identification of those activities that should be prioritized in order to keep the project's duration at a minimum. In this case, those activities are A, B, C, G, J, L, N, O, R, S, T, and the total duration of the project would be of 390 working hours, which is equivalent to 48.75 full-time working days if none of these activities were delayed.

Table 4. Precedence analysis of the tasks defined in the WBS diagram.

Letter	Task	Precedents	Duration (hours)
A	Planning of the project	-	10
B	Basic concepts formation	A	25
C	Transformers formation	B	25
D	RAKE	B	25
E	YAKE	B	25
F	TextRank	B	25
G	KeyBERT	C	50
H	K-means	D, E, F, G	20
I	Transformer's AutoModel for classification	C, D, E, F, G	40
J	Transformer's SetFit for classification	C, D, E, F, G	50
K	Data collection	B	30
L	Transformer's AutoModel for relation	C, H, I, J, K	50
M	Transformer's SetFit for relation	C, H, I, J, K	40
N	Structure definition	L, M	50
O	Tutorials and other documents	N	25
P	Unsupervised NER	O	20
Q	Relations validation tool	O	10
R	Study of term prevalence and co-occurrence	O	40
S	Project document	O, P, Q, R	50
T	Project presentation	S	15

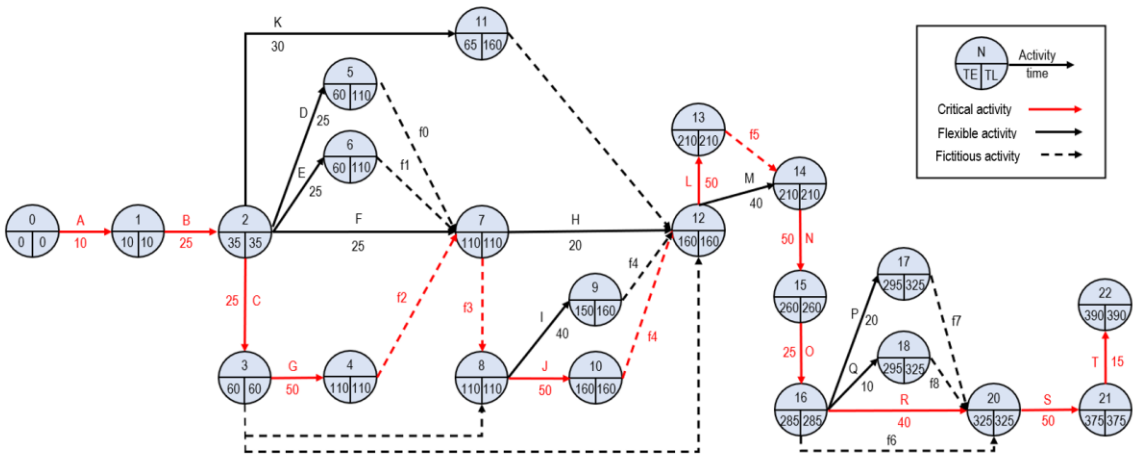


Figure 22. PERT-CPM diagram of the tasks defined in the WBS diagram. It shows in black the flexible activities, in red the critical activities belonging to the Critical Path, and in dotted lines the fictitious activities. Each of the nodes also includes its time early and last. Own creation.

6.4. GANTT diagram

The last step of this section is the elaboration of the GANTT diagram, which illustrates the project schedule by showing when each activity should be performed in a defined timeline. Using the calculations of the PERT diagram, Table 13 of section 12.2. Additional material has been developed. This table shows the earliest and latest times at which each activity can be developed to finish the project in the scheduled time, as well as the slack of each activity, which is the time that activity can be delayed without affecting the schedule. Using this table, the GANTT diagram has been developed as shown in Figure 25. As seen, the critical activities have been marked in red and have no slack, since their delay would affect the project's schedule. On the other hand, the flexible activities in blue, with a light blue shadow showing their corresponding time slack.

Note that both the PERT and the GANTT diagrams account for a total of 390 working hours or 48.75 full-time working days, from following the Critical Path Method. However, this project has not been developed at such a steady rate but rather as part of a half-time internship that lasted from July 2023 to January 2024. Thus, the NLP formation tasks took place in July 2023, the development of the keyword extractors until September 2023, the term categorizers until October 2023, the relation classifiers until December 2023, and the implementation of the library until January 2023. On the other hand, the use cases and the project documents have been elaborated between February 2024 and June 2024, meeting the milestones defined in Table 11.

In addition to the shown activities, some future steps have already started to be developed. These are explained in further detail in section 10. Conclusions and future steps, but here a little detail on their execution schedule is also given. Regarding the implementation of AttentionRank as a keyword extraction method, it started on May 2024 and is expected to finish by August 2024, since it is parallel to other activities. On the other hand, further training the term and relation classifiers with new data is expected to happen between June and July 2024. Finally, the complete implementation of use cases such as the relations validation tool and the study of term prevalence are expected to last until October 2024.

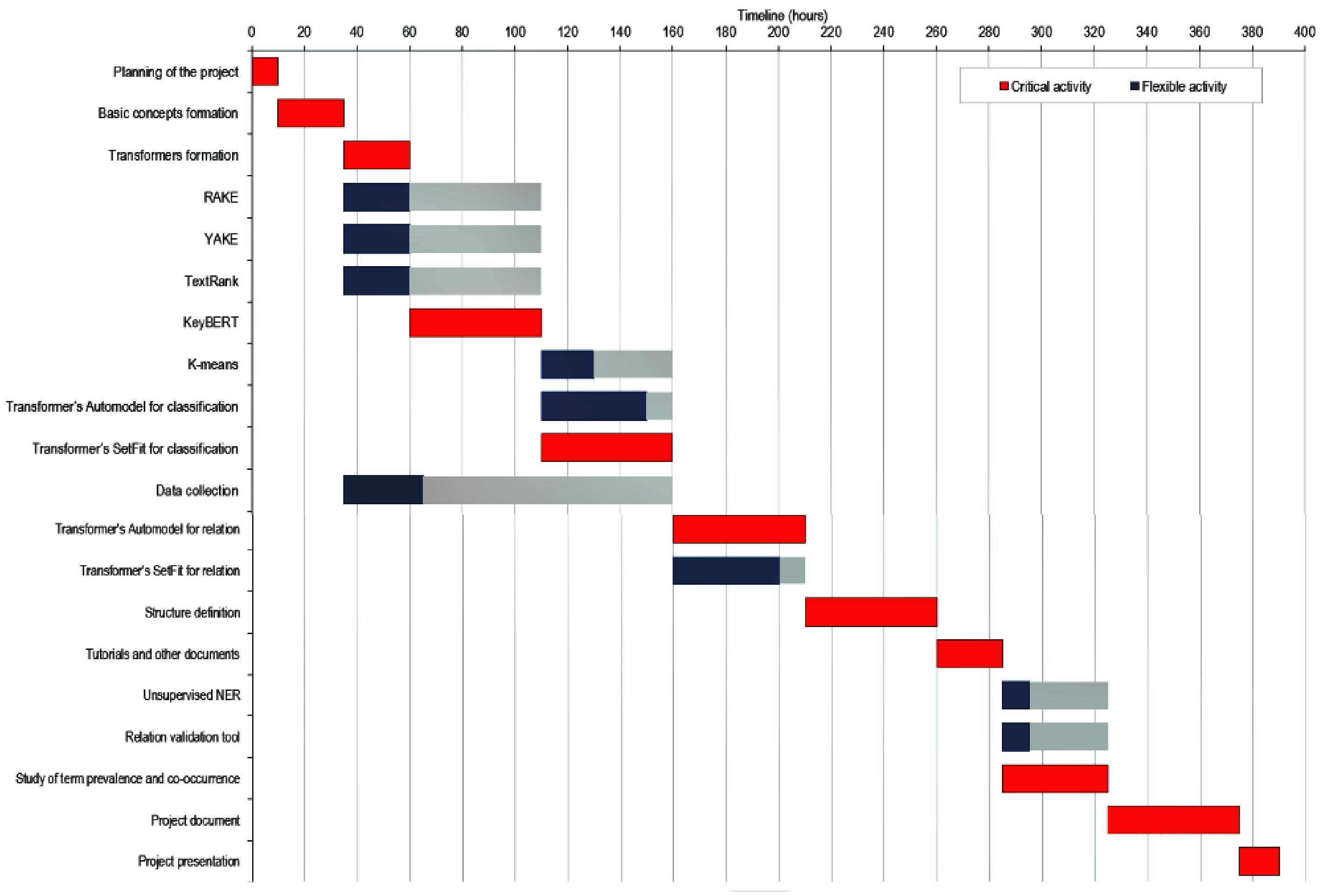


Figure 23. GANTT diagram of the project showing how the tasks are developed along the span of 390 hours. The figure shows the critical activities in red and the flexible ones in blue with their corresponding margin. Own creation.

7. Technical viability: SWOT analysis

This section evaluates the technical viability of the project as has been observed during its development. This enables the identification of scenarios that have affected both positively and negatively the development of the project, as well as those scenarios that might affect its further development and application in the future. This has been achieved with the SWOT method (Strengths-Weaknesses-Opportunities-Threats), as shown in Table 5. This table shows the inherent string and weak points of the project, as well as possible future situations that could constitute either opportunities or threats to its correct development. As seen, some of the library's most significant strengths rely on the resources provided by the BSC, as well as on the use of unsupervised and few-shot models. On the other hand, most of its weaknesses refer to the lack of manually annotated data for specific tasks, since Spanish is not as high-resource a language as English. Regarding the project's opportunities, both the library's wide range of use cases and the publishing of papers on them, as well as its implementation as part of other projects are of note. Other opportunities stem from the library's flexibility of use in different scenarios and with different models, while its main threats are based on the negligence to update the library accordingly to new technologies.

Table 5. SWOT analysis of the project, showing its Strengths, Weaknesses, Opportunities and Threats.

	Positive	Negative
Internal	**Strengths** • Support of the NLP4BIA group at the BSC, guidance from my tutor and supervisor, and access to the MareNostrum 4 and 5 supercomputers. • Previous formation in NLP and Sentence Transformers • Access to pretrained models with biomedical data in Spanish from the private repositories of the BSC (SapBERT in Spanish) [55] • Access to manually annotated NER Gold Standard Corpora and other data (MedProcNER [56], DisTEMIST [57], CanTEMIST [60], etc.) • Use of unsupervised or few-shot methods requiring little training data • Implementation of different methods and models in the library • Possibility for training and evaluating models directly from the library • Adaptability of the library to new and improved models and methods	**Weaknesses** • Lack of Gold Standard data for the evaluation of keyword extraction methods • Lower scores for unsupervised keyword extraction algorithms compared to the NER Gold Standard Corpora • Very computationally expensive training (relation classifiers) • Artificial data for the training of the relation classifiers (obtained from the SNOMED-CT structure [3], not manually validated relations) • Limited time for the development of the library due to the due-date in early June • Initial lack of expertise in NLP, requiring previous formation
External	**Opportunities** • Not language-specific, so it can be applied in other languages with the adequate base models • Not domain-specific, so it can be applied in other domains (such as the juridic domain) with adequate training data • Future implementation of improved models and methods, such as AttentionRank [50] for keyword extraction (soon to be published). This way, threats like the release of generative models would be converted into an opportunity, by being adopted in the library. • Wide range of use cases and applications, including: • Generation of NER candidates in an unsupervised manner • Semiautomatic validation tool for terminology enrichment • Term prevalence and co-occurrence study in specific domains • Automatic key information extraction and relation • Knowledge graph, entity linking and cross-ontology mapping • Paper on the development of the library and its use as a semiautomatic terminology enrichment tool (currently in development) • Paper on its use for the study of keyword prevalence in medical documents against different parameters (currently in development) • Presentation of KeyCARE at events like a BSC's Department Seminar • Use of KeyCARE as part of the BSC's collaboration in the project TeresIA, thus providing continuity and funding to the project [10].	**Threats** • Generation of AI models that might outperform the ones in the library (might be incorporated in the library, if fitting). Generative models suppose a special threat due to their promising results. • Generation of training data that would enable the training of supervised models with equivalent objectives both in Spanish and low-resource languages • Too computationally expensive models training for users working in a different specific domain with their own data • Difficult interpretability of the code, even with the comments and tutorials provided in GitHub • Possible insufficient maintenance or updating of the library

8. Economic viability

This section is comprised of the development of a budget including all the costs associated to the project and a temporal evaluation of the according payments for the length of the project.

8.1. Table of costs

In Table 6, each of the project's costs is accounted for, considering both material resources and human resources, as well as other services. Of the costs listed in the table, the two main costs come from the undergraduate's student salary and the computational resources used to develop the project. This is mainly because the accessed data, models and modules were already of public access and therefore have no associated cost. The computational costs have been approximated with information provided by the BSC, because the supercomputer where the project has been developed is public and therefore does not have a set cost. The provided approximation comes from dividing the submitted jobs between small jobs – under two hours -, big jobs – over 24 hours -, and typical jobs – in between -, and assessing how many of them had been run. This project has entailed the execution of 31 small jobs of an hour of duration, 97 jobs of 48 hours of duration, and 270 jobs of roughly seven hours of duration, all using just one node. The associated CU expense (Computing Resources) and the cost has been computed using the calculator from Archer CU2 [72] which provides an approximation of 6,577 CUs and 2.565,03 GBP. Converting this to euros and applying a correction factor of 2:1 due to the difference in computational capacity between MareNostrum4 (the used supercomputer) and Archer2 (the supercomputer from the calculator), a total of 5,985,32 € are obtained. This correction must be applied because the MareNostrum4 has roughly double the computational capacity of Archer2.

Table 6. Table of costs[8] of the project.

Material resources						
Concept: resources	Function	Payer	Qty	Unit price	Total price	Associated tasks
Dell Personal Computer	Project development	BSC	1	509.00 €	509.00 €	All
Services						
Concept: resources	Function	Payer	Qty	Unit price	Total price	Associated tasks
Software licenses for modules	Access to NLP tools	BSC	1	0.00 €	0.00 €	All
Computational resources	Training and evaluation of DL models	BSC	6,577 CU	0.91 €/CU	5,985,32 €	All
Access to data from SNOMED-CT, UMLS and NER Corpora	Training and evaluation of DL models	BSC	1	0.00 €	0.00 €	All
Formation in NLP	Acquisition of the necessary abilities	BSC	1	0.00 €	0.00 €	Basic concepts formation and Huggingface's Sentence Transformers formation
Electricity, air conditioning and other associated costs	Development of the project	BSC	7 months	20.00 €/month	140.00 €	All
Manpower resources						
Concept: resources	Function	Payer	Qty	Unit price	Total price	Associated tasks
Salary of the undergraduate student	Development of the project	BSC	7 months	719.10 €/month*	5,101.64 €	All
Salary of the supervisor	Supervision of the project	BSC	7 months	350 €/month**	2,450.00 €	All

[8] *Computed as the mean salary over the 7-months internship, in which there were two different salaries.
**Computed as an approximated 10% of the salary of the supervisor.

8.2. Table of payments

The costs described in Table 6 have been fully paid by the BSC over the span of seven months, from July 2023 to January 2024. Table 7 shows how each of the costs has been paid over the span of the project. As seen, the computational resources, electricity and the salary of the supervisor are assumed to have been equally distributed over the project, while the salary of the student is computed according to the working hours per week, which decreased from 25 hours per week to 19 hours per week in October.

Table 7. Table of payments of the project.

Costs	July	August	September	October	November	December	January	Total
Dell Personal Computer	509.00 €	-	-	-	-	-	-	509.00 €
Software licenses for modules	-	-	-	-	-	-	-	0.00 €
Computational resources	855.05 €	855.05 €	855.05 €	855.05 €	855.05 €	855.05 €	855.05 €	5,985.32 €
Access to data	-	-	-	-	-	-	-	0.00 €
Formation in NLP	-	-	-	-	-	-	-	0.00 €
Electricity, air conditioning and other associated costs	20.00 €	20.00 €	20.00 €	20.00 €	20.00 €	20.00 €	20.00 €	53.00 €
Salary of the undergraduate student	788.56 €	788.56 €	788.56 €	683.99 €	683.99 €	683.99 €	683.99 €	5,101.64 €
Salary of the supervisor	350.00 €	350.00 €	350.00 €	350.00 €	350.00 €	350.00 €	350.00 €	2,450.00 €
Total	**2,422.61 €**	**2,013.61 €**	**2,013.61 €**	**1,909.04 €**	**1,909.04 €**	**1,909.04 €**	**1,909.04 €**	**14,185.99 €**

9. Legislation and regulations

This section describes the main legislations, regulations, and ethical aspects concerning the development of KeyCARE and its publication.

The main regulation under which the publication of KeyCARE should be considered is the European AI Act [73]. The AI Act, a first-of-its-kind legal framework, aims to regulate AI development and usage in Europe by setting clear rules for developers and companies. The implementation of this regulation, along with other AI initiatives, aims to ensure safe and ethical AI across the EU, fostering innovation and leadership in the field. The AI Act follows a risk-based approach, which identifies AI uses as one of four categories: unacceptable risk, high risk, limited risk, or minimal risk. According to the guidelines provided by the European Commission, KeyCARE falls squarely within the category of minimal risk, since it does not entail a lack of transparency, and thus can be used within the EU. Note that the AI Act is not yet applicable in the EU, but it will be enforced in under two years.

In addition to that, it is important to note that KeyCARE is not directly intended as a software for providing medical practitioners with relevant information from clinical records. The main applications of KeyCARE are information extraction for terminology enrichment and the generation of data for the training of more complex models. However, should KeyCARE be used for the purpose of extracting information from patient records, its use should always be in an assisting role to the medical practitioner, who should verify all the extracted information. In that case the current European Medical Devices Regulation (MDR) (EU 2017/745) states that "software for general purposes, even when used in a healthcare setting, or software intended for lifestyle and well-being purposes is not a medical Device [74]. Thus, KeyCARE does not necessarily need to comply with the regulations defined in the European MDR.

Another aspect to be considered is the carbon footprint of the used models as part of the KeyCARE library, both in their training and application. According to the information provided in Huggingface, the training of Transformer architectures from scratch is immensely computationally expensive and therefore generates a significative carbon footprint [13]. However, using pre-trained models for fine-tuning, as is the case for this project, helps reduce the carbon emissions of such processes by reducing the training times. In addition to that, the general cluster of the MareNostrum4 – the infrastructure used for the training of the provided models in this project – has been recognised as the "greenest" supercomputer of Europe by the Green 500 List, which evaluates the carbon footprint of the top 500 most powerful supercomputers in the world [75]. Still, this does not mean that it is carbon neutral or sustainable, so the training of large models as some from this project should always be done after careful consideration.

Finally, another issue to consider is whether the trained AI models have learned from data that does not represent well the entire population and can present bias towards some population groups. Since the training data used from the models mainly comes from structured terminologies for medical concepts and Gold Standard Corpora annotated entities, it seems unlikely that it constitutes a biased dataset. However, the possibility that diseases or symptoms that mostly affect minoritarian groups are not correctly represented in the data cannot be entirely discarded.

10. Conclusions and future steps

This last section of the project comments on the extracted conclusions regarding the final implementation of the library and the obtained results from the evaluation of its pipeline and use cases. In addition to that, it introduces some further discussion on how KeyCARE will be implemented in real-life projects and what its further applications might be. Finally, it also comments on the future work that will be done to update the library and give it a more solid applicability.

KeyCARE has been developed as a framework that enables two main pipelines: one for the automatic extraction of biomedical keywords that are classified into different semantic groups and another one for the extraction of semantic relations among medical terms. These two pipelines constitute the two main processes for information extraction, by being able to identify and extract entities in the form of keywords and being able to extract relations among those keywords and with other terms. Each step of these pipelines has successfully been implemented in Spanish with a variety of parameters that allow the tunning of their processes, as well as with alternative methods and models that provide adaptability to the library. In addition to that, each of the implemented models in these pipelines has been evaluated, as well as some of the main applications for which the library is intended.

Regarding the implementation of the **keyword extractors**, four different alternatives have already been successfully implemented. Each one of them presents relevant characteristics for their application, such as their scalability, their domain-independence, their multilingual support, and their ability to function well on single documents. When evaluating them on NER Gold Standard Corpora, some of the extractors have shown great sensibility for relevant entities, resulting in high recall scores. In addition to that, each of the implemented extractors functions internally in a different manner and has better results for specific configurations, what makes this pipeline more adaptable to different scenarios. As for future work, State-of-the-Art keyword extractors are already being implemented to the library. The next version of the library will also include AttentionRank as an extractor that supposedly outperforms the other ones thanks to the use of Language Models as well as a focus on Attention.

With respect to the implementation of the **term categorizers**, an unsupervised clustering algorithm and two supervised classifiers have successfully been implemented. The classifiers have both shown impressive results in the classification of medical entities, especially considering that one of them functions in a few-shot manner and can be trained with small sets of data. The fact that both a few-shot classifier and a completely unsupervised clustering algorithm have been implemented also provides adaptability to multilingual scenarios where there is sparse available data. Additionally, this classification step functions as a filter of the previous one, discarding those keywords that are considered non valid. In reference to the future steps on the term classifiers, the next version of the library will include term classifiers trained on more classes that better describe medical keywords, such as body structures or risk factors.

Concerning the implementation of the **semantic relations extractors**, two supervised classifiers for hierarchical relations have been implemented, one of which is few-shot. Both relation classifiers have also shown impressive results even when dealing with corner cases as those from different evaluation datasets. The fact that this step of the pipeline also provides a few-shot alternative solidifies the adaptability of KeyCARE to low-resource languages or specific domains. Regarding the future work of the relations extractor, it will soon be trained with bigger sets of data from SNOMED-CT which should also have a better representation of all the types of NO_RELS.

The joint use of the keyword extractors and the term categorizers provides the library with an unsupervised alternative to NER. As evaluated in section 5.4.1. Unsupervised generation of NER candidates, this application of the library has yielded impressive results for an unsupervised implementation. While it obviously does not attain results comparable to those of a correctly trained NER system, it does provide a solid alternative that can be used in situations where there is no training data available. This use of the library could be useful to automatically produce Silver Standard data that could be manually revised to generate annotated corpora for the training of NER systems. Furthermore, this application of the library can directly be performed with data from any specific domain and in any given language with only a few minor requirements. For the keyword extractors RAKE needs a stopword list in the given language to function correctly and KeyBERT needs a base model that represents terms adequately. For the term classification, only a base model and a small training set is required if using SetFit. Even if there is no available training data, as little as a hundred mentions per class can be generated manually or using machine translation and SetFit should be able to identify the classes accordingly. It is also a future step of this project to implement this pipeline in other languages, both high-resource such as English and low-resource like Catalan. The first ones will enable a more profound evaluation and advertisement of the library, while the second ones will have a direct application on the production of terminology for low-resource languages.

As seen, the implemented methodologies in KeyCARE also allow for the implementation of other use cases. Regarding the analysis of keyword prevalence in the literature with respect to parameters such as the publication year or the paper's DeCS codes, it is already being conducted by the author at the BSC with the intention of producing valuable research that will be published in a research paper. Moreover, an API (Application Programming Interface) is already being developed for the implementation of the semiautomatic tool for relations validation so that this tool can also soon be published alongside another research paper. In addition to that, KeyCARE has many potential applications for data generation and enrichment that have not been explored as part of this project, as could be cross-ontology mapping and entity linking.

Finally, it is imperative to remark that KeyCARE has been developed within the mark of the TeresIA project and covering some of the main work packages of the project. This is of great importance because it means that KeyCARE already has a real-life application for which it will be used: a nation-wide project for the coordination of AI tools and the creation of a terminology that sets a common framework for the Spanish language. It also means that KeyCARE will continue to be developed and updated with the advancement of NLP methods for as long as TeresIA remains active, and that it will continue to be used as part of the project.

11. References

[1] Jee, K., & Kim, G. H. (2013). Potentiality of big data in the medical sector: focus on how to reshape the healthcare system. *Healthcare informatics research*, *19*(2), 79-85.

[2] Nasar, Z., Jaffry, S. W., & Malik, M. K. (2021). Named entity recognition and relation extraction: State-of-the-art. *ACM Computing Surveys (CSUR)*, *54*(1), 1-39.

[3] Home. (s/f). SNOMED International. Retrieved January 10th, 2024, from https://www.snomed.org/

[4] Unified medical language system (UMLS). (2009). Retrieved January 10th, 2024, from https://www.nlm.nih.gov/research/umls/index.html

[5] DeCS - Descriptores en Ciencias de la Salud. (s/f). Bvsalud.org. Retrieved March 4th, 2024, from https://decses.bvsalud.org/E/homepagee.html

[6] Extensión Catalana SNOMED CT. (2022, August 23). TIC Salut Social. Retrieved May 28th, 2024, from https://ticsalutsocial.cat/es/formulari-dobtencio-de-snomed-ct-2/

[7] Zaratiana, U., Tomeh, N., Holat, P., & Charnois, T. (2023). Gliner: Generalist model for named entity recognition using bidirectional transformer. *arXiv preprint arXiv:2311.08526*.

[8] Bogdanov, S., Constantin, A., Bernard, T., Crabbé, B., & Bernard, E. (2024). NuNER: Entity Recognition Encoder Pre-training via LLM-Annotated Data. *arXiv preprint arXiv:2402.15343*.

[9] Miranda-Escalada, A., Mehryary, F., Luoma, J., Estrada-Zavala, D., Gasco, L., Pyysalo, S., ... & Krallinger, M. (2023). Overview of DrugProt task at BioCreative VII: data and methods for large-scale text mining and knowledge graph generation of heterogenous chemical–protein relations. *Database*, *2023*, baad080.

[10] TERESIA. Portal de acceso a terminologías en España y servicios de inteligencia artificial. (n.d.). Csic.es. Retrieved June 1st, 2024, from https://pti-esciencia.csic.es/project/teresia-portal-de-acceso-a-terminologias-en-espana-y-servicios-de-inteligencia-artificial/

[11] TeresIA, el proyecto de terminología en español de la PTI ES CIENCIA, es finalista de los Premios de Internet en la categoría “Emprendimiento e investigación.” (n.d.). Csic.es. Retrieved June 1st, 2024, from https://www.cchs.csic.es/es/article/teresia-proyecto-terminologia-espanol-pti-es-ciencia-es-finalista-premios-internet

[12] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

[13] Learn. (n.d.). Huggingface.Co. Retrieved May 15th, 2024, from https://huggingface.co/learn/

[14] Luis Gascó Sánchez (2024). "Figura 22. Esquema sobre los tipos de modelos de lenguaje". Apunte del Módulo Text Mining del "Master Data Science, Big Data & Business Analytics". Universidad Complutense de Madrid.

[15] Gu, Y., Tinn, R., Cheng, H., Lucas, M., Usuyama, N., Liu, X., ... & Poon, H. (2021). Domain-specific language model pretraining for biomedical natural language

processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, *3*(1), 1-23.

[16] Liu, F., Shareghi, E., Meng, Z., Basaldella, M., & Collier, N. (2020). Self-alignment pretraining for biomedical entity representations. *arXiv preprint arXiv:2010.11784*.

[17] Johri, P., Khatri, S. K., Al-Taani, A. T., Sabharwal, M., Suvanov, S., & Kumar, A. (2021). Natural language processing: History, evolution, application, and future work. In *Proceedings of 3rd International Conference on Computing Informatics and Networks: ICCIN 2020* (pp. 365-375). Springer Singapore.

[18] Hancox, P. (1996). A brief history of Natural Language Processing. *University of Birmingham School of Computer Science, www. cs. bham. ac. uk/~ pjh/sem1a5/pt1/pt1_history. html*.

[19] Sekine, S. (2004). Named entity: History and future. *Project notes, New York University*, *4*.

[20] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[21] Center, P. C. (2011). 2011 IEEE International Conference on Acoustics, Speech, and Signal Processing.

[22] GoLLIE. (n.d.). GoLLIE. Retrieved June 2nd, 2024, from https://hitz-zentroa.github.io/GoLLIE/

[23] Zhou, W., Zhang, S., Gu, Y., Chen, M., & Poon, H. (2023). Universalner: Targeted distillation from large language models for open named entity recognition. *arXiv preprint arXiv:2308.03279*.

[24] Wang, L., Cao, Z., De Melo, G., & Liu, Z. (2016, August). Relation classification via multi-level attention cnns. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 1298-1307).

[25] Wan, Q., Wei, L., Zhao, S., & Liu, J. (2023). A Span-based Multi-Modal Attention Network for joint entity-relation extraction. *Knowledge-Based Systems*, *262*, 110228.

[26] D. Sui, X. Zeng, Y. Chen, K. Liu, and J. Zhao, "Joint Entity and Relation Extraction With Set Prediction Networks," *IEEE Trans Neural Netw Learn Syst*, 2023, doi: 10.1109/TNNLS.2023.3264735.

[27] Landolsi, M. Y., Hlaoua, L., & Ben Romdhane, L. (2023). Information extraction from electronic medical documents: state of the art and future research directions. *Knowledge and Information Systems*, *65*(2), 463-516.

[28] Collis, B. (1999). Applications of computer communications in education: an overview. *IEEE Communications Magazine*, *37*(3), 82-86.

[29] Panja, S. (2024). Information Retrieval Systems in Healthcare: Understanding Medical Data Through Text Analysis. In M. Garcia & R. de Almeida (Eds.), *Transformative Approaches to Patient Literacy and Healthcare Innovation* (pp. 180-200). IGI Global. https://doi-org.sire.ub.edu/10.4018/979-8-3693-3661-8.ch009

[30] SentenceTransformers Documentation — Sentence Transformers documentation. (n.d.). Sbert.net. Retrieved February 19th, 2024, from https://sbert.net/

[31] El Gobierno aprueba la Estrategia de Inteligencia Artificial 2024. (n.d.). Gob.es. Retrieved May 5th, 2024, from https://portal.mineco.gob.es/es-es/comunicacion/Paginas/20240514-Gobierno-aprueba-Estrategia-IA-2024.aspx

[32] Cervantes, I. (n.d.). *«TeresIA», el proyecto para fomentar la terminología en español con inteligencia artificial, se presenta en Bruselas a los traductores de la Unión Europea*. Cervantes.org. Retrieved June 2nd, 2024, from https://cervantes.org/es/sobre-nosotros/sala-prensa/notas-prensa/teresia-proyecto-fomentar-terminologia-espanol-inteligencia

[33] Día mundial de internet. (n.d.). Diadeinternet.org. Retrieved June 1st, 2024, from https://www.diadeinternet.org/2024/?page=new_index&act=381&cortesia=

[34] Artificial intelligence market size, share, growth report 2030. (2024). Retrieved March 22nd, 2024, from https://www.grandviewresearch.com/industry-analysis/artificial-intelligence-ai-market

[35] *Natural language processing market size, share & trends analysis report by component, by deployment model, by enterprise size, by type, by application, by end-use, by region, and segment forecasts, 2023 - 2030.* (s/f). Grandviewresearch.com. Retrieved March 22nd, 2024, from https://www.grandviewresearch.com/industry-analysis/natural-language-processing-market-report

[36] *Investigación*. (s/f). Sepln.org. Retrieved May 24th, 2024, from http://www.sepln.org/investigacion

[37] Somos, Q. (s/f). *Ontology Engineering Group*. Upm.es. Retrieved May 24th, 2024, from https://oeg.fi.upm.es/

[38] IE & IR - information extraction and information retrieval. (s/f). Upc.edu. Retrieved May 25th, 2024, from https://www.talp.upc.edu/EXPERTISE-AREA-DETAIL/430

[39] Projects. (s/f). Google.com. Retrieved January 27th, 2024, from https://sites.google.com/view/nlp-uned/projects?authuser=0

[40] Plan de Tecnologías del Lenguaje - Plan de Impulso de las Tecnologías del Lenguaje. (s/f). Gob.es. Retrieved May 27th, 2024, from https://plantl.mineco.gob.es/tecnologias-lenguaje/PTL/Paginas/plan-impulso-tecnologias-lenguaje.aspx

[41] Projecte Aina. (s/f). Polítiques Digitals. Retrieved June 1st, 2024, from https://politiquesdigitals.gencat.cat/ca/economia/catalonia-ai/aina/

[42] Wiggins, D. (2023, enero 12). *Hype cycle for AI technologies in business*. Omniscien Technologies. Retrieved April 17th, 2024, from https://omniscien.com/blog/hype-cycle-for-ai-technologies-in-business/

[43] Sammut, C., & Webb, G. I. (2010). Tf–idf. *Encyclopedia of machine learning*, 986-987.

[44] Campos, R., Mangaravite, V., Pasquali, A., Jorge, A., Nunes, C., & Jatowt, A. (2020). YAKE! Keyword extraction from single documents using multiple local features. *Information Sciences*, *509*, 257-289.

[45] Rose, S., Engel, D., Cramer, N., & Cowley, W. (2010). Automatic keyword extraction from individual documents. *Text mining: applications and theory*, 1-20.

[46] Mihalcea, R., & Tarau, P. (2004, July). Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing* (pp. 404-411).

[47] Bennani-Smires, K., Musat, C., Hossmann, A., Baeriswyl, M., & Jaggi, M. (2018). Simple unsupervised keyphrase extraction using sentence embeddings. *arXiv preprint arXiv:1801.04470.*

[48] Bennani-Smires, K., Musat, C., Hossmann, A., Baeriswyl, M., & Jaggi, M. (2018). Simple unsupervised keyphrase extraction using sentence embeddings. *arXiv preprint arXiv:1801.04470.*

[49] Nikzad-Khasmakhi, N., Feizi-Derakhshi, M. R., Asgari-Chenaghlu, M., Balafar, M. A., Feizi-Derakhshi, A. R., Rahkar-Farshi, T., ... & Ranjbar-Khadivi, M. (2021). Phraseformer: Multimodal key-phrase extraction using transformer and graph embedding. *arXiv preprint arXiv:2106.04939*.

[50] Ding, H., & Luo, X. (2021, November). Attentionrank: Unsupervised keyphrase extraction using self and cross attentions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing* (pp. 1919-1928).

[51] Auto Classes. (s/f). Huggingface.co. Retrieved January 12th, 2024, from https://huggingface.co/docs/transformers/model_doc/auto

[52] Tunstall, L., Reimers, N., Jo, U. E. S., Bates, L., Korat, D., Wasserblat, M., & Pereg, O. (2022). Efficient few-shot learning without prompts. *arXiv preprint arXiv:2209.11055*.

[53] KeyCARE. Acces GitHub from https://github.com/nlp4bia-bsc/KeyCARE

[54] KeyCARE. Access PyPI from https://pypi.org/project/keycare/

[55] Gallego, F., López-García, G., Gasco-Sánchez, L., Krallinger, M., & Veredas, F. J. (2024). ClinLinker: Medical Entity Linking of Clinical Concept Mentions in Spanish. *arXiv preprint arXiv:2404.06367*.

[56] Lima-López, S., Farré-Maduell, E., Gascó, L., Nentidis, A., Krithara, A., Katsimpras, G., ... & Krallinger, M. (2023). Overview of MedProcNER task on medical procedure detection and entity linking at BioASQ 2023. *Working Notes of CLEF*.

[57] Miranda-Escalada, A., Gascó, L., Lima-López, S., Farré-Maduell, E., Estrada, D., Nentidis, A., ... & Krallinger, M. (2022, September). Overview of DisTEMIST at BioASQ: Automatic detection and normalization of diseases from clinical texts: results, methods, evaluation and multilingual resources. In *CLEF (Working Notes)* (pp. 179-203).

[58] Lima-López, S., Farré-Maduell, E., Gasco-Sánchez, L., Rodríguez-Miret, J., & Krallinger, M. (2023). Overview of SympTEMIST at BioCreative VIII: corpus, guidelines and evaluation of systems for the detection and normalization of symptoms, signs and findings from text. In *Proceedings of the BioCreative VIII Challenge and Workshop: Curation and Evaluation in the era of Generative Models*.

[59] Lima-López, S., Farré-Maduell, E., Krallinger, M. (2024). DrugTEMIST Guidelines: Annotation of Medication in Medical Documents. Zenodo. doi: 10.5281/zenodo.11065433.

[60] Miranda-Escalada, A., Farré, E., & Krallinger, M. (2020). Named Entity Recognition, Concept Normalization and Clinical Coding: Overview of the Cantemist Track for Cancer Text Mining in Spanish, Corpus, Guidelines, Methods and Results. *IberLEF@ SEPLN*, 303-323.

[61] Miranda-Escalada, A., Farré-Maduell, E., Lima-López, S., Estrada, D., Gascó, L., & Krallinger, M. (2022). Mention detection, normalization & classification of

species, pathogens, humans and food in clinical documents: Overview of the LivingNER shared task and resources. *Procesamiento del Lenguaje Natural*, *69*, 241-253.

[62] Lima-López, S., Farré-Maduell, E., Miranda-Escalada, A., Brivá-Iglesias, V., & Krallinger, M. (2021). Nlp applied to occupational health: Meddoprof shared task at iberlef 2021 on automatic recognition, classification and normalization of professions and occupations from medical texts. *Procesamiento del Lenguaje Natural*, *67*, 243-256.

[63] Lima-López, S., Farré-Maduell, E., Briva-Iglesias, V., Gascó, L., & Krallinger, M. (2023). MEDDOPLACE Shared Task overview: recognition, normalization and classification of locations and patient movement in clinical texts.

[64] *meddoplace_scoring_script: Scoring script of the MEDDOPLACE Shared Task*. (s/f). Retrieved November 10th, 2023, from https://github.com/TeMU-BSC/meddoplace_scoring_script

[65] Rodriguez-Penagos, C., Nentidis, A., Gonzalez-Agirre, A., Asensio, A., Armengol-Estapé, J., Krithara, A., ... & Krallinger, M. (2020). Overview of mesinesp8, a Spanish medical semantic indexing task within bioasq 2020. *Working Notes of CLEF*, 1-12.

[66] (N.d.). Butlleti.Cat. Retrieved May 22nd, 2024, from http://www.butlleti.cat/

[67] *BOE-A-2010-3514 Ley Orgánica 2/2010. de 3 de marzo, de salud sexual y reproductiva y de la interrupción voluntaria del embarazo*. (s/f). Boe.es. Retrieved June 4th, 2024, from https://www.boe.es/buscar/act.php?id=BOE-A-2010-3514

[68] *BSC-NLP4BIA/biomedical-term-classifier-setfit · Hugging Face*. (n.d.). Huggingface.Co. Retrieved June 4, 2024, from https://huggingface.co/BSC-NLP4BIA/biomedical-term-classifier-setfit

[69] *BSC-NLP4BIA/biomedical-term-classifier · Hugging Face*. (n.d.). Huggingface.Co. Retrieved June 4, 2024, from https://huggingface.co/BSC-NLP4BIA/biomedical-term-classifier

[70] *BSC-NLP4BIA/biomedical-semantic-relation-classifier-setfit · Hugging Face*. (n.d.). Huggingface.Co. Retrieved June 4, 2024, from https://huggingface.co/BSC-NLP4BIA/biomedical-semantic-relation-classifier-setfit

[71] *BSC-NLP4BIA/biomedical-semantic-relation-classifier · Hugging Face*. (n.d.). Huggingface.Co. Retrieved June 4, 2024, from https://huggingface.co/BSC-NLP4BIA/biomedical-semantic-relation-classifier

[72] ARCHER2 CU calculator. (n.d.). Archer2.ac.uk. Retrieved May 22nd, 2024, from https://www.archer2.ac.uk/support-access/cu-calc.html

[73] AI Act. (n.d.). Shaping Europe's Digital Future. Retrieved April 20th, 2024, from https://digital-strategy.ec.europa.eu/en/policies/regulatory-framework-ai

[74] Regulation - 2017/745 - EN - medical device regulation - EUR-Lex. (n.d.). Europa.Eu. RetrievedApril 20th, 2024, from https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32017R0745

[75] La nueva máquina del BSC es el supercomputador más "verde" de Europa. (n.d.). BSC-CNS. Retrieved April 20th, 2024, from https://www.bsc.es/es/noticias/noticias-del-bsc/la-nueva-m%C3%A1quina-del-bsc-es-el-supercomputador-m%C3%A1s-%E2%80%9Cverde%E2%80%9D-de-europa

12. Annexes

This section provides extra material for a better comprehension of the project, including a glossary of the used terminology and tables of methodology and results that are not included in the body of the project.

12.1. Glossary

This section includes short descriptions for all the terms that are used in the document but are not indispensable for the understanding of the project and thus are not explained in-depth. These terms have been marked in blue the first time they appear in the document:

Attention mechanisms: Neural Networks mechanisms that leverage learnable weights to selectively focus on relevant elements within a sequence, enhancing model performance in tasks like translation and question answering.

Augmented Transition Networks (ATN): computer science grammar model for NLP.

Compound Annual Growth Rate (CAGR): average measure of variable's growth rate over a period.

Conditional Random Fields (CRF): statistical modeling method for sequence labeling tasks.

Contrastive Learning: Machine Learning technique using similarities and differences between data samples.

Convolutional Neural Networks (CNNs): Artificial neural network architecture effective for image recognition.

Cross-ontology Mapping: aligning concepts and relationships between different knowledge structures.

DeCS: a controlled vocabulary for indexing and searching biomedical literature in Spanish.

Dyspnea: medical term for shortness of breath.

Electronic Health Records (EHRs): digital versions of patient's medical history.

Entity Linking: connecting textual mentions of entities to entries in a knowledge base.

Graph Convolutional Networks (GCNs): Neural network architecture designed for graph-structured data.

Hypercapnia: medical term for abnormally high blood carbon dioxide level.

Jaccard Index: metric used to compare similarity and diversity of sets.

Joint Modeling Approaches: Machine Learning techniques modeling multiple tasks or data aspects simultaneously.

kwargs: Python term referring to keyword arguments passed to a function that can be used as additional parameters.

Large Language Models (LLMs): powerful AI models trained on massive amounts of text data, capable of various tasks like text generation and translation.

Long Short-Term Memory (LSTM): a type of recurrent neural network architecture adept at handling long-term dependencies in sequential data.

Maximal Marginal Relevance (MMR): a ranking technique used to retrieve documents most relevant and diverse from a set of search results.

Multilabel Confusion Matrix: a visualization tool used to evaluate the performance of multilabel classification models.

Multilabel Sentence Classification: Machine Learning task where a sentence can be assigned multiple labels simultaneously.

Multi-Modal Attention Network: Neural network architecture that processes and combines information from different modalities (e.g., text, images) for tasks like sentiment analysis.

n-grams: Sequences of n consecutive words or characters used for various language processing tasks like machine translation.

Non-regressive parallel decoding: a decoding technique in machine translation that predicts the entire target sequence at once, rather than word-by-word.

Open Information Extraction: technique for automatically extracting relationships between entities mentioned in text without relying on predefined taxonomies.

Ontology: a formal representation of knowledge as a set of concepts, their relationships, and properties used for knowledge sharing.

Question Answering: subfield of NLP where machines are trained to answer questions posed in natural language.

Recurrent Neural Networks (RNNs): a class of neural networks designed to handle sequential data like text by processing information from previous steps.

Semantic Knowledge Databases: large repositories of structured information about concepts and their relationships.

SNOMED-CT: a standardized medical terminology for consistently documenting clinical information.

Softmax: an activation function used in neural networks to convert output layer values into probabilities for multi-class classification.

Stopword: a commonly used word (e.g., "the", "a") that is often filtered out before text processing due to its low semantic value.

Structural Support Vector Machine (SSVM): a type of ML model for structured prediction tasks that leverages structural information in data.

Targeted Distillation: a technique for compressing knowledge from a large pre-trained model into a smaller model by focusing on specific tasks.

Terminology: a collection of terms specific to a particular domain or field.

Text Prediction: the NLP task of predicting the next word or sequence of words in a text, often used for tasks like autocompletion or text generation.

Text Understanding: the ability of a machine learning model to process and comprehend the meaning of text data.

Thesaurus: a reference tool that groups words with similar meanings or synonyms, aiding in finding alternative words or expressions.

Transfer Learning: a ML technique where knowledge gained while training a model on one task is leveraged to improve performance on a related but different task.

UMLS: a large knowledge base that links biomedical vocabularies and provides tools for terminology services.

Virtual Environment: an isolated Python environment that allows you to manage project-specific dependencies without affecting your system-wide Python installation.

word2vec: a popular algorithm for learning word embeddings, which represent words as numerical vectors capturing their semantic relationships.

Zero-padding: a technique used to ensure sequences have the same length. Padding characters (often zeros) are added to shorter sequences in a batch.

Zero-shot Learning: a ML approach where a model can perform a task without any training examples for that specific task, relying on previously learned knowledge.

12.2. Additional material

This annex includes all the tables and figures not shown in the document but that have been referenced.

Table 8. Precision, recall and f-score with exact and relaxed overlaps of all extractors over MedProcNER. In this table, A, B, C, D, E and F represent different PoS pattern for extraction.

MedProcNER						
Extractor	precision	recall	f-score	overlapping_precision	overlapping_recall	overlapping_f_score
medprocner_RAKE1	0.0289	0.1268	0.0471	0.1112	0.4880	0.1811
medprocner_TextRank1	0.0370	0.1342	0.0580	0.1244	0.4515	0.1950
medprocner_YAKE1	0.0131	0.0583	0.0213	0.0796	0.3551	0.1300
medprocner_RAKE3	0.0361	0.2618	0.0634	0.1155	0.8380	0.2030
medprocner_YAKE3	0.0171	0.0747	0.0278	0.1020	0.4472	0.1661
medprocner_TextRank3	0.0283	0.1906	0.0493	0.1292	0.8704	0.2251
medprocner_RAKE5	0.0352	0.2622	0.0620	0.1193	0.8900	0.2105
medprocner_YAKE5	0.0171	0.0747	0.0278	0.1020	0.4472	0.1661
medprocner_TextRank5	0.0264	0.1836	0.0462	0.1300	0.9026	0.2273
medprocner_combined3	0.0258	0.4052	0.0485	0.0627	0.9854	0.1179
medprocner_combined1	0.0210	0.2080	0.0382	0.0735	0.7263	0.1335
medprocner_combined5	0.0250	0.4052	0.0470	0.0610	0.9910	0.1150
medprocner_KeyBert_noparents3	0.0165	0.0134	0.0148	0.1696	0.1376	0.1519
medprocner_KeyBert_noparents1	0.0199	0.0121	0.0151	0.1521	0.0925	0.1150
medprocner_KeyBert_parents5	0.0376	0.0316	0.0344	0.2914	0.2452	0.2663
medprocner_KeyBert_parents1	0.0735	0.0341	0.0466	0.2706	0.1254	0.1713
medprocner_KeyBert_parents3	0.0468	0.0347	0.0398	0.2847	0.2112	0.2425
medprocner_KeyBert_noparents5	0.0144	0.0129	0.0136	0.1813	0.1629	0.1716
medprocner_KeyBert_BSC1	0.0338	0.0131	0.0189	0.2165	0.0840	0.1211
medprocner_KeyBert_BSC3	0.0235	0.0113	0.0153	0.2182	0.1050	0.1418
medprocner_KeyBertPOS_parentsD	0.0798	0.0309	0.0446	0.2373	0.0919	0.1325
medprocner_KeyBert_BSC5	0.0187	0.0106	0.0135	0.2266	0.1282	0.1638
medprocner_KeyBertPOS_parentsE	0.0951	0.0373	0.0536	0.2608	0.1024	0.1471
medprocner_KeyBertPOS_parentsF	0.0962	0.0381	0.0546	0.2554	0.1012	0.1450
medprocner_KeyBertPOS_noparentsE	0.0439	0.0166	0.0241	0.2030	0.0769	0.1116
medprocner_KeyBertPOS_noparentsD	0.0389	0.0146	0.0213	0.1953	0.0735	0.1068
medprocner_KeyBertPOS_noparentsF	0.0451	0.0173	0.0250	0.1992	0.0763	0.1103
medprocner_KeyBertPOS_noparentsC	0.0313	0.0109	0.0162	0.1888	0.0661	0.0979
medprocner_KeyBertPOS_parentsA	0.0471	0.0171	0.0251	0.2452	0.0888	0.1304
medprocner_KeyBertPOS_noparentsA	0.0283	0.0099	0.0147	0.1878	0.0661	0.0977
medprocner_KeyBertPOS_noparentsB	0.0392	0.0148	0.0215	0.1881	0.0712	0.1033
medprocner_KeyBertPOS_parentsB	0.0795	0.0310	0.0446	0.2303	0.0898	0.1292
medprocner_KeyBertPOS_parentsC	0.0503	0.0182	0.0267	0.2524	0.0911	0.1339
medprocner_KeyBert_noparents1_top10	0.0204	0.0221	0.0213	0.1488	0.1611	0.1547

medprocner_KeyBert_noparents3_top10	0.0157	0.0224	0.0185	0.1623	0.2315	0.1908
medprocner_KeyBert_noparents5_top10	0.0138	0.0213	0.0168	0.1743	0.2689	0.2115
medprocner_KeyBert_parents5_top10	0.0288	0.0440	0.0348	0.2536	0.3871	0.3065
medprocner_KeyBert_parents3_top10	0.0374	0.0521	0.0435	0.2463	0.3432	0.2868
medprocner_KeyBert_parents1_top10	0.0586	0.0545	0.0565	0.2365	0.2200	0.2279
medprocner_KeyBertPOS_noparentsA_top10	0.0303	0.0217	0.0253	0.1792	0.1282	0.1494
medprocner_KeyBertPOS_noparentsB_top10	0.0406	0.0313	0.0353	0.1748	0.1345	0.1520
medprocner_KeyBertPOS_noparentsF_top10	0.0466	0.0362	0.0407	0.1882	0.1461	0.1645
medprocner_KeyBertPOS_noparentsD_top10	0.0405	0.0309	0.0350	0.1783	0.1362	0.1544
medprocner_KeyBertPOS_noparentsC_top10	0.0336	0.0238	0.0278	0.1818	0.1288	0.1508
medprocner_KeyBertPOS_noparentsE_top10	0.0461	0.0355	0.0401	0.1921	0.1479	0.1672
medprocner_KeyBertPOS_parentsA_top10	0.0457	0.0336	0.0388	0.2121	0.1559	0.1797
medprocner_KeyBertPOS_parentsC_top10	0.0541	0.0394	0.0456	0.2253	0.1641	0.1899
medprocner_KeyBertPOS_parentsE_top10	0.0860	0.0690	0.0765	0.2339	0.1875	0.2082
medprocner_KeyBertPOS_parentsB_top10	0.0665	0.0524	0.0586	0.1955	0.1542	0.1724
medprocner_KeyBertPOS_parentsD_top10	0.0661	0.0519	0.0581	0.2010	0.1578	0.1768
medprocner_KeyBertPOS_parentsF_top10	0.0874	0.0703	0.0779	0.2291	0.1843	0.2042
medprocner_KeyBert_noparents1_top15	0.0191	0.0298	0.0233	0.1423	0.2228	0.1737
medprocner_KeyBert_parents3_top15	0.0320	0.0631	0.0425	0.2221	0.4371	0.2945
medprocner_KeyBert_parents1_top15	0.0505	0.0701	0.0587	0.2118	0.2943	0.2463
medprocner_KeyBert_parents5_top15	0.0248	0.0520	0.0336	0.2292	0.4796	0.3102
medprocner_KeyBert_noparents3_top15	0.0142	0.0285	0.0189	0.1541	0.3094	0.2057
medprocner_KeyBert_noparents5_top15	0.0124	0.0264	0.0169	0.1660	0.3532	0.2259
medprocner_KeyBertPOS_parentsA_top15	0.0455	0.0506	0.0479	0.1924	0.2141	0.2027
medprocner_KeyBertPOS_parentsC_top15	0.0574	0.0632	0.0602	0.2109	0.2320	0.2209
medprocner_KeyBertPOS_parentsB_top15	0.0628	0.0746	0.0682	0.1779	0.2113	0.1931
medprocner_KeyBertPOS_parentsF_top15	0.0824	0.1007	0.0906	0.2142	0.2618	0.2356
medprocner_KeyBertPOS_parentsE_top15	0.0825	0.0999	0.0903	0.2187	0.2649	0.2396
medprocner_KeyBertPOS_parentsD_top15	0.0627	0.0742	0.0680	0.1822	0.2156	0.1975
medprocner_KeyBertPOS_noparentsA_top15	0.0322	0.0349	0.0335	0.1712	0.1855	0.1781
medprocner_KeyBertPOS_noparentsB_top15	0.0416	0.0484	0.0448	0.1621	0.1886	0.1744
medprocner_KeyBertPOS_noparentsE_top15	0.0488	0.0569	0.0526	0.1863	0.2174	0.2006
medprocner_KeyBertPOS_noparentsF_top15	0.0492	0.0579	0.0532	0.1821	0.2142	0.1968
medprocner_KeyBertPOS_noparentsC_top15	0.0362	0.0387	0.0374	0.1745	0.1864	0.1802
medprocner_KeyBertPOS_noparentsD_top15	0.0399	0.0462	0.0428	0.1642	0.1899	0.1761
medprocner_KeyBert_noparents5_top20	0.0110	0.0292	0.0160	0.1566	0.4154	0.2275
medprocner_KeyBert_parents1_top20	0.0448	0.0835	0.0583	0.1953	0.3643	0.2543
medprocner_KeyBert_parents3_top20	0.0283	0.0710	0.0405	0.2047	0.5137	0.2928
medprocner_KeyBert_parents5_top20	0.0224	0.0581	0.0323	0.2123	0.5518	0.3066
medprocner_KeyBertPOS_noparentsA_top20	0.0315	0.0458	0.0373	0.1609	0.2342	0.1907
medprocner_KeyBertPOS_parentsA_top20	0.0453	0.0672	0.0541	0.1807	0.2685	0.2160
medprocner_KeyBertPOS_noparentsB_top20	0.0396	0.0617	0.0482	0.1513	0.2361	0.1844

medprocner_KeyBertPOS_noparentsC_top20	0.0359	0.0513	0.0422	0.1651	0.2364	0.1944
medprocner_KeyBertPOS_noparentsD_top20	0.0389	0.0603	0.0473	0.1546	0.2399	0.1880
medprocner_KeyBertPOS_noparentsE_top20	0.0490	0.0772	0.0599	0.1756	0.2766	0.2148
medprocner_KeyBertPOS_noparentsF_top20	0.0496	0.0788	0.0608	0.1723	0.2740	0.2116
medprocner_KeyBertPOS_parentsB_top20	0.0600	0.0960	0.0738	0.1662	0.2658	0.2045
medprocner_KeyBertPOS_parentsC_top20	0.0593	0.0870	0.0705	0.2007	0.2948	0.2388
medprocner_KeyBertPOS_parentsD_top20	0.0595	0.0946	0.0731	0.1698	0.2699	0.2085
medprocner_KeyBertPOS_parentsE_top20	0.0782	0.1273	0.0969	0.2022	0.3295	0.2506
medprocner_KeyBertPOS_parentsF_top20	0.0786	0.1289	0.0977	0.2002	0.3282	0.2487
medprocner_KeyBert_noparents1_top20	0.0175	0.0354	0.0234	0.1326	0.2688	0.1776
medprocner_KeyBert_noparents3_top20	0.0127	0.0323	0.0182	0.1452	0.3695	0.2085

Table 9. Precision, recall and f-score with exact and relaxed overlaps of all extractors over DisTEMIST. In this table, A, B, C, D, E and F represent different PoS pattern for extraction.

DisTEMIST						
Extractor	precision	recall	f-score	overlapping_precision	overlapping_recall	overlapping_f_score
distemist_YAKE1	0.0152	0.0931	0.0261	0.0742	0.4543	0.1275
distemist_RAKE1	0.0220	0.1324	0.0377	0.0687	0.4138	0.1178
distemist_TextRank1	0.0307	0.1530	0.0511	0.0942	0.4696	0.1569
distemist_RAKE3	0.0343	0.3415	0.0623	0.0816	0.8130	0.1483
distemist_TextRank3	0.0267	0.2472	0.0483	0.0912	0.8430	0.1646
distemist_YAKE3	0.0238	0.1433	0.0408	0.0960	0.5781	0.1647
distemist_RAKE5	0.0341	0.3485	0.0620	0.0844	0.8641	0.1538
distemist_YAKE5	0.0238	0.1433	0.0408	0.0960	0.5781	0.1647
distemist_TextRank5	0.0251	0.2392	0.0454	0.0925	0.8817	0.1675
distemist_KeyBert_noparents1	0.0471	0.0393	0.0428	0.1971	0.1647	0.1794
distemist_KeyBert_parents5	0.0295	0.0341	0.0316	0.1885	0.2179	0.2021
distemist_KeyBert_parents1	0.0487	0.0310	0.0379	0.1661	0.1058	0.1292
distemist_KeyBert_noparents5	0.0338	0.0417	0.0373	0.2171	0.2678	0.2398
distemist_KeyBert_noparents3	0.0408	0.0455	0.0430	0.2109	0.2350	0.2223
distemist_KeyBert_BSC1	0.0458	0.0244	0.0319	0.2213	0.1180	0.1540
distemist_KeyBert_parents3	0.0387	0.0394	0.0391	0.1794	0.1829	0.1811
distemist_KeyBert_BSC3	0.0362	0.0239	0.0288	0.2276	0.1505	0.1812
distemist_KeyBert_BSC5	0.0250	0.0195	0.0219	0.2297	0.1787	0.2010
distemist_KeyBertPOS_parentsD	0.0783	0.0417	0.0544	0.1733	0.0923	0.1204
distemist_KeyBertPOS_parentsF	0.0895	0.0487	0.0631	0.1949	0.1061	0.1374
distemist_KeyBertPOS_parentsE	0.0901	0.0486	0.0631	0.1966	0.1060	0.1377
distemist_KeyBertPOS_noparentsE	0.1627	0.0847	0.1114	0.3487	0.1815	0.2388
distemist_KeyBertPOS_noparentsD	0.1548	0.0800	0.1055	0.3328	0.1720	0.2268
distemist_KeyBertPOS_noparentsA	0.0582	0.0281	0.0380	0.3576	0.1728	0.2331
distemist_KeyBertPOS_noparentsF	0.1629	0.0857	0.1123	0.3454	0.1816	0.2381
distemist_KeyBertPOS_noparentsC	0.0578	0.0278	0.0375	0.3566	0.1714	0.2315

distemist_KeyBertPOS_parentsA	0.0324	0.0161	0.0215	0.2227	0.1108	0.1480
distemist_KeyBertPOS_noparentsB	0.1538	0.0800	0.1052	0.3278	0.1705	0.2243
distemist_KeyBertPOS_parentsC	0.0353	0.0175	0.0234	0.2341	0.1161	0.1552
distemist_KeyBertPOS_parentsB	0.0770	0.0413	0.0538	0.1696	0.0909	0.1183
distemist_KeyBert_noparents1_top10	0.0388	0.0577	0.0464	0.1808	0.2689	0.2162
distemist_KeyBert_noparents3_top10	0.0333	0.0652	0.0441	0.1838	0.3601	0.2434
distemist_KeyBert_noparents5_top10	0.0268	0.0568	0.0364	0.1899	0.4025	0.2581
distemist_KeyBert_parents5_top10	0.0230	0.0481	0.0311	0.1634	0.3425	0.2212
distemist_KeyBert_parents3_top10	0.0298	0.0570	0.0391	0.1556	0.2978	0.2044
distemist_KeyBert_parents1_top10	0.0357	0.0456	0.0401	0.1443	0.1844	0.1619
distemist_KeyBertPOS_noparentsA_top10	0.0530	0.0521	0.0525	0.2885	0.2834	0.2859
distemist_KeyBertPOS_noparentsB_top10	0.1175	0.1242	0.1208	0.2649	0.2801	0.2723
distemist_KeyBertPOS_noparentsD_top10	0.1184	0.1242	0.1212	0.2677	0.2810	0.2742
distemist_KeyBertPOS_noparentsC_top10	0.0553	0.0538	0.0546	0.2917	0.2838	0.2877
distemist_KeyBertPOS_noparentsE_top10	0.1305	0.1381	0.1342	0.2907	0.3076	0.2989
distemist_KeyBertPOS_noparentsF_top10	0.1304	0.1391	0.1346	0.2906	0.3100	0.3000
distemist_KeyBertPOS_parentsA_top10	0.0320	0.0324	0.0322	0.1877	0.1896	0.1887
distemist_KeyBertPOS_parentsC_top10	0.0351	0.0351	0.0351	0.2052	0.2053	0.2052
distemist_KeyBertPOS_parentsB_top10	0.0625	0.0677	0.0650	0.1490	0.1614	0.1550
distemist_KeyBertPOS_parentsD_top10	0.0619	0.0668	0.0643	0.1481	0.1598	0.1538
distemist_KeyBertPOS_parentsF_top10	0.0768	0.0849	0.0807	0.1783	0.1971	0.1873
distemist_KeyBertPOS_parentsE_top10	0.0761	0.0838	0.0798	0.1774	0.1954	0.1860
distemist_KeyBert_parents3_top15	0.0256	0.0692	0.0374	0.1440	0.3895	0.2103
distemist_KeyBert_parents5_top15	0.0195	0.0560	0.0289	0.1508	0.4335	0.2238
distemist_KeyBert_parents1_top15	0.0315	0.0601	0.0413	0.1348	0.2573	0.1769
distemist_KeyBert_noparents3_top15	0.0280	0.0772	0.0411	0.1646	0.4539	0.2416
distemist_KeyBert_noparents1_top15	0.0330	0.0709	0.0450	0.1632	0.3510	0.2228
distemist_KeyBert_noparents5_top15	0.0233	0.0682	0.0348	0.1702	0.4975	0.2537
distemist_KeyBertPOS_parentsA_top15	0.0294	0.0450	0.0356	0.1693	0.2588	0.2047
distemist_KeyBertPOS_parentsC_top15	0.0341	0.0516	0.0411	0.1858	0.2808	0.2236
distemist_KeyBertPOS_parentsB_top15	0.0575	0.0939	0.0713	0.1357	0.2215	0.1683
distemist_KeyBertPOS_parentsD_top15	0.0576	0.0936	0.0713	0.1362	0.2215	0.1687
distemist_KeyBertPOS_parentsF_top15	0.0713	0.1198	0.0894	0.1665	0.2796	0.2087
distemist_KeyBertPOS_parentsE_top15	0.0720	0.1198	0.0899	0.1669	0.2777	0.2085
distemist_KeyBertPOS_noparentsB_top15	0.0995	0.1590	0.1224	0.2278	0.3640	0.2802
distemist_KeyBertPOS_noparentsA_top15	0.0472	0.0703	0.0565	0.2408	0.3586	0.2881
distemist_KeyBertPOS_noparentsD_top15	0.0999	0.1587	0.1226	0.2300	0.3654	0.2823
distemist_KeyBertPOS_noparentsC_top15	0.0492	0.0722	0.0585	0.2452	0.3598	0.2917
distemist_KeyBertPOS_noparentsE_top15	0.1101	0.1764	0.1356	0.2532	0.4060	0.3119
distemist_KeyBertPOS_noparentsF_top15	0.1099	0.1777	0.1358	0.2517	0.4068	0.3110
distemist_KeyBert_noparents3_top20	0.0251	0.0877	0.0390	0.1473	0.5147	0.2290
distemist_KeyBert_noparents5_top20	0.0205	0.0748	0.0322	0.1531	0.5575	0.2402

distemist_KeyBert_noparents1_top20	0.0288	0.0801	0.0423	0.1461	0.4068	0.2150
distemist_KeyBert_parents1_top20	0.0281	0.0720	0.0404	0.1242	0.3183	0.1787
distemist_KeyBert_parents5_top20	0.0176	0.0629	0.0275	0.1404	0.5013	0.2194
distemist_KeyBert_parents3_top20	0.0226	0.0780	0.0351	0.1325	0.4565	0.2054
distemist_KeyBertPOS_noparentsA_top20	0.0432	0.0863	0.0575	0.2136	0.4272	0.2848
distemist_KeyBertPOS_noparentsB_top20	0.0858	0.1840	0.1171	0.1997	0.4281	0.2724
distemist_KeyBertPOS_noparentsC_top20	0.0455	0.0895	0.0603	0.2181	0.4291	0.2892
distemist_KeyBertPOS_noparentsD_top20	0.0866	0.1845	0.1178	0.2010	0.4285	0.2737
distemist_KeyBertPOS_noparentsE_top20	0.0968	0.2095	0.1324	0.2194	0.4751	0.3002
distemist_KeyBertPOS_noparentsF_top20	0.0954	0.2084	0.1309	0.2191	0.4787	0.3006
distemist_KeyBertPOS_parentsA_top20	0.0292	0.0595	0.0391	0.1550	0.3163	0.2080
distemist_KeyBertPOS_parentsB_top20	0.0536	0.1178	0.0737	0.1263	0.2776	0.1737
distemist_KeyBertPOS_parentsC_top20	0.0350	0.0707	0.0468	0.1734	0.3500	0.2320
distemist_KeyBertPOS_parentsD_top20	0.0538	0.1175	0.0738	0.1261	0.2754	0.1730
distemist_KeyBertPOS_parentsE_top20	0.0692	0.1550	0.0957	0.1581	0.3540	0.2186
distemist_KeyBertPOS_parentsF_top20	0.0685	0.1542	0.0949	0.1575	0.3547	0.2182

.

Table 10. Classification report of the term classifier containing all classes.

Category	Precision	Recall	F1-score	Support
ACTIVITY	0.63	0.61	0.62	28
COMMUNITY	0.88	0.74	0.80	91
DEPARTMENT	0.97	0.97	0.97	803
DISEASE	0.83	0.82	0.83	2598
FAC_GEN	0.97	0.96	0.97	760
FAC_NOM	0.90	0.83	0.86	94
DRUG	0.94	0.70	0.81	416
GEO_GEN	0.80	0.94	0.86	85
GEO_NOM	0.50	0.40	0.44	10
GPE_GEN	0.87	0.87	0.87	112
GPE_NOM	0.92	0.94	0.93	500
HUMAN	0.99	0.94	0.96	3150
LANGUAGE	0.00	0.00	0.00	23
NEOPLASM MORPHOLOGY	0.92	0.96	0.94	3633
NO CATEGORY	0.96	0.99	0.97	88
PROCEDURE	0.96	0.97	0.97	3619
PROFESSION	0.79	0.93	0.85	699
SYMPTOM	0.91	0.88	0.90	3103
EMPLOYMENT SITUATION	0.89	0.84	0.86	358

SPECIES	0.98	0.99	0.98	4251
TRANSPORTATION	0.96	0.98	0.97	200
Macro avg	0.84	0.82	0.83	24621
Weighted avg	0.93	0.93	0.93	24621

Table 11. Phases and milestones table

ID	Description	Responsible	Due date	Supervisor/evaluator	Deliverables
1	**Planning of the project:** task planning over a timeline. Includes Phases and Milestones Table, WBS (with its WBS dictionary), PERT-CPM diagram, and GANTT diagram.	Sergi Marsol Torrent	15/07/2023	Juan Ignacio Barrios Arce & Luis Gascó Sánchez	The "Execution Schedule" section of this Project. Has been modified during the length of the Project.
2	**NLP Training with SentenceTransformers:** training on the use of SentenceTransformers for different NLP purposes using Huggingface and other resources.	Sergi Marsol Torrent	15/07/2023	Luis Gascó Sánchez	Certificate of completion of the Huggingface courses.
3	**Implementation of the keyword extractors:** research and implementation of unsupervised methods for keyword extraction within the KeyCARE library. Includes the creation of specific modules and classes with integrated functions.	Sergi Marsol Torrent	15/09/2023	Luis Gascó Sánchez	Python module with classes for each of the extractors, the keywords, and the main class (partially implemented). Also results of the evaluation of the extractors against a NER Gold Standard.
4	**Implementation of the term categorizers:** research and implementation of supervised and unsupervised methods for term classification and clustering, respectively, within the KeyCARE library. Includes the creation of specific modules and classes with integrated functions for training and evaluation.	Sergi Marsol Torrent	15/11/2023	Luis Gascó Sánchez	Python module with classes for each of the classifiers, the clustering, and the main class (fully implemented). Also results of the evaluation of the categorizers against a NER Gold Standard.
5	**Implementation of the term relation classifiers:** research and implementation of supervised methods for term pairs relation classification within the KeyCARE library. Includes the creation of specific modules and classes with integrated functions for training and evaluation.	Sergi Marsol Torrent	01/02/2024	Luis Gascó Sánchez	Python module with classes for each of the relation classifiers, and a second main class. Also results of the evaluation of the relation classifiers against different datasets.
6	**Implementation of KeyCARE:** publication of the library on GitHub and Pypy. Includes the cleaning of the code, the development of tutorials, a readme and other accessory files, and the creation of the library's structure.	Sergi Marsol Torrent	01/05/2024	Luis Gascó Sánchez & Martin Krallinger	Published library with all corresponding documents on GitHub and Pypi.
7	**Evaluation of the use cases:** evaluation of KeyCARE's function for specific tasks with practical uses.	Sergi Marsol Torrent	01/05/2024	Luis Gascó Sánchez & Martin Krallinger	Python script for the evaluation and corresponding scores.
8	**Project document:** complete document detailing the elaboration of the library and all complementary sections.	Sergi Marsol Torrent	05/06/2024	Juan Ignacio Barrios Arce & Project Tribunal	The document itself.
9	**Project presentation:** summarized presentation on the developed project.	Sergi Marsol Torrent	10/06/2024	Juan Ignacio Barrios Arce & Project Tribunal	The presentation itself.

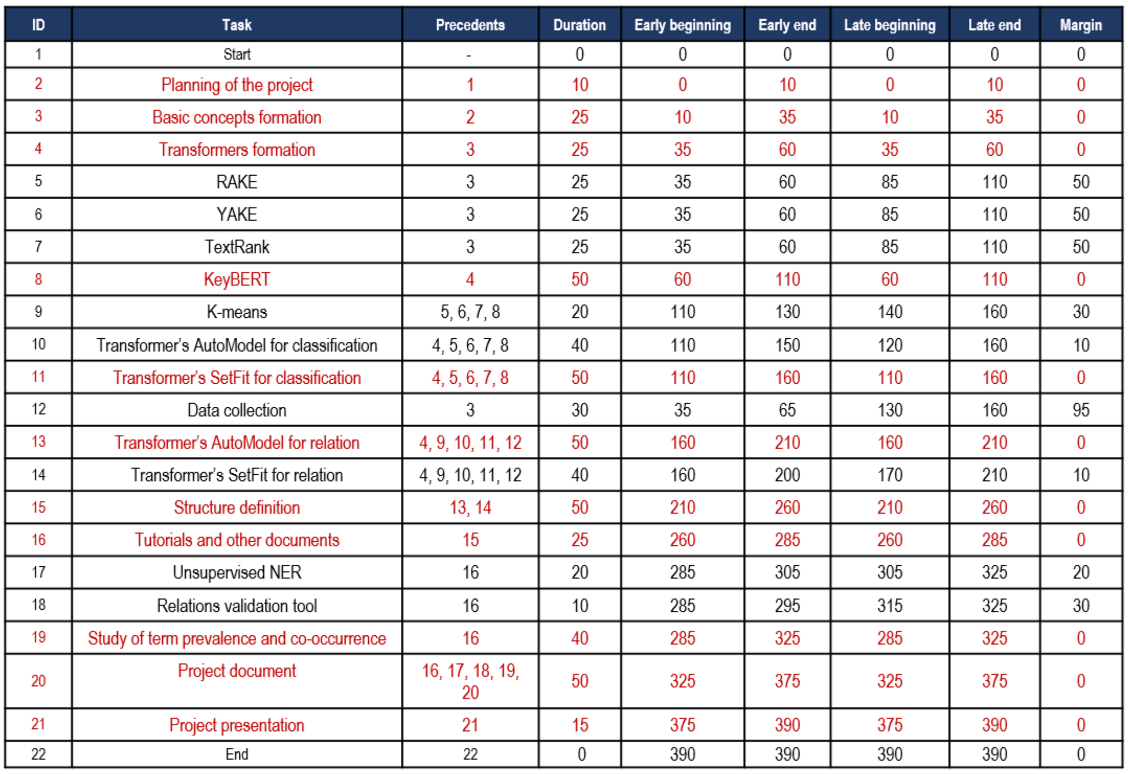
Table 12. WBS dictionary of all the defined tasks in the WBS diagram.

Work package 1: Project management	
Task 1: Planning of the project	
Description	Includes the generation of the diagrams for the execution schedule, as the PERT-CPM diagram or the GANTT chronogram.
Deliverables	The “Execution Schedule” section of this document.
Duration	10 hours
Predecessors	None
Successors	Basic concepts formation
Task 2: Project document	
Description	Includes the creation of the project memory with all the details of the project.
Deliverables	This whole document.
Duration	50 hours
Predecessors	Tutorials and other documents, unsupervised NER, relations validation tool, study of term orevalence and co-occurrence.
Successors	Project presentation
Task 3: Project presentation	
Description	Includes the creation and exposition of the presentation of the project.
Deliverables	The presentation itself.
Duration	15 hours
Predecessors	Project document
Successors	None
Work package 2: Basic NLP formation	
Task 1: Basic concepts formation	
Description	Formation on NLP basic concepts that are used for the duration of the internship and the project.
Deliverables	None
Duration	25 hours
Predecessors	Planning of the project
Successors	Transformers formation, RAKE, YAKE, TextRank, Data collection
Task 2: Transformers formation	
Description	Formation on the Transformer architecture and their use, as well as the use of Huggingface.
Deliverables	None
Duration	25 hours
Predecessors	Basic concepts formation
Successors	KeyBERT, Transformer’s AutoModel for classification, Transformer’s SetFit for classification, Transformer’s AutoModel for relation, Transformer’s SetFit for relation
Work package 3: Keyword Extractors implementation	
Task 1: RAKE	
Description	Development of the necessary classes and functions for the implementation of the RAKE extractor in the library
Deliverables	The corresponding functional part of the library and its evaluation
Duration	25 hours
Predecessors	Basic concepts formation
Successors	K-means, Transformer’s AutoModel for classification, Transformer’s SetFit for classification
Task 2: YAKE	
Description	Development of the necessary classes and functions for the implementation of the YAKE extractor in the library
Deliverables	The corresponding functional part of the library and its evaluation
Duration	25 hours
Predecessors	Basic concepts formation
Successors	K-means, Transformer’s AutoModel for classification, Transformer’s SetFit for classification
Task 3: TextRank	
Description	Development of the necessary classes and functions for the implementation of the TextRank extractor in the library
Deliverables	The corresponding functional part of the library and its evaluation
Duration	25 hours
Predecessors	Basic concepts formation
Successors	K-means, Transformer’s AutoModel for classification, Transformer’s SetFit for classification
Task 4: KeyBERT	
Description	Development of the necessary classes and functions for the implementation of the KeyBERT extractor in the library

Deliverables	The corresponding functional part of the library and its evaluation
Duration	50 hours
Predecessors	Transformers formation
Successors	K-means, Transformer's AutoModel for classification, Transformer's SetFit for classification
Work package 4: Term Categorizers implementation	
Task 1: K-means	
Description	Development of the necessary classes and functions for the implementation of the K-means clustering algorithm in the library
Deliverables	The corresponding functional part of the library
Duration	20 hours
Predecessors	RAKE, YAKE, TextRank, KeyBERT
Successors	Transformer's AutoModel for relation, Transformer's SetFit for relation
Task 2: Transformer's AutoModel for classification	
Description	Development of the necessary classes and functions for the implementation of the Transformers classifier in the library
Deliverables	The corresponding functional part of the library and its evaluation
Duration	40 hours
Predecessors	Transformers formation, RAKE, YAKE, TextRank, KeyBERT
Successors	Transformer's AutoModel for relation, Transformer's SetFit for relation
Task 3: Transformer's SetFit for classification	
Description	Development of the necessary classes and functions for the implementation of the SetFit classifier in the library
Deliverables	The corresponding functional part of the library and its evaluation
Duration	50 hours
Predecessors	Transformers formation, RAKE, YAKE, TextRank, KeyBERT
Successors	Transformer's AutoModel for relation, Transformer's SetFit for relation
Work package 5: Relation Classifiers implementation	
Task 1: Data collection	
Description	Retrieval of data from SNOMED-CT and UMLS for the training of the relation classifiers
Deliverables	A tsv format document with the corresponding data
Duration	30 hours
Predecessors	Basic concepts formation
Successors	Transformer's AutoModel for relation, Transformer's SetFit for relation
Task 2: Transformer's AutoModel for classification	
Description	Development of the necessary classes and functions for the implementation of the Transformers relation classifier in the library
Deliverables	The corresponding functional part of the library and its evaluation
Duration	50 hours
Predecessors	Transformers formation, K-means, Transformer's AutoModel for classification, Transformer's SetFit for classification, Data collection
Successors	Structure definition
Task 3: Transformer's SetFit for classification	
Description	Development of the necessary classes and functions for the implementation of the SetFit relation classifier in the library
Deliverables	The corresponding functional part of the library and its evaluation
Duration	40 hours
Predecessors	Transformers formation, K-means, Transformer's AutoModel for classification, Transformer's SetFit for classification, Data collection
Successors	Structure definition
Work package 7: Implementation of the library	
Task 1: Structure definition	
Description	Definition of the class structure of KeyCARE for an optimal performance.
Deliverables	The src/keycare/ folder of the library
Duration	50 hours
Predecessors	Transformer's AutoModel for relation, Transformer's SetFit for relation
Successors	Tutorials and other documents
Task 2: Tutorials and other documents	
Description	Jupyter notebook tutorials of the functioning of the library and other accessory documents for the correct functioning of the library and its publication.
Deliverables	The files and folders of KeyCARE that are outside the src/ folder.
Duration	25 hours
Predecessors	Structure definition

Successors	Unsupervised NER, Relations validation tool, Study of term prevalence and co-occurrence, Project document
Work package 7: Use cases evaluation	
Task 1: Unsupervised NER	
Description	Generation of NER candidates in an unsupervised manner using KeyCARE.
Deliverables	A short evaluation on medical documents corpora and an example in a low-resource language as Catalan.
Duration	20 hours
Predecessors	Tutorials and other documents
Successors	Project document
Task 2: Relations validation tool	
Description	Generation of relations among terms in an unsupervised manner using KeyCARE.
Deliverables	An example of functioning of the tool over medical documents
Duration	10 hours
Predecessors	Tutorials and other documents
Successors	Project document
Task 3: Study of term prevalence and co-occurrence	
Description	Study of keywords prevalence and co-occurrence over medical literature using KeyCARE.
Deliverables	An example of functioning of the analysis over medical documents
Duration	40 hours
Predecessors	Tutorials and other documents
Successors	Project document

Table 13. GANTT chronogram table including the earliest and latest times that activities can start and end, as well as the time margin of each activity. The critical activities are marked in red.



ID	Task	Precedents	Duration	Early beginning	Early end	Late beginning	Late end	Margin
1	Start	-	0	0	0	0	0	0
2	Planning of the project	1	10	0	10	0	10	0
3	Basic concepts formation	2	25	10	35	10	35	0
4	Transformers formation	3	25	35	60	35	60	0
5	RAKE	3	25	35	60	85	110	50
6	YAKE	3	25	35	60	85	110	50
7	TextRank	3	25	35	60	85	110	50
8	KeyBERT	4	50	60	110	60	110	0
9	K-means	5, 6, 7, 8	20	110	130	140	160	30
10	Transformer's AutoModel for classification	4, 5, 6, 7, 8	40	110	150	120	160	10
11	Transformer's SetFit for classification	4, 5, 6, 7, 8	50	110	160	110	160	0
12	Data collection	3	30	35	65	130	160	95
13	Transformer's AutoModel for relation	4, 9, 10, 11, 12	50	160	210	160	210	0
14	Transformer's SetFit for relation	4, 9, 10, 11, 12	40	160	200	170	210	10
15	Structure definition	13, 14	50	210	260	210	260	0
16	Tutorials and other documents	15	25	260	285	260	285	0
17	Unsupervised NER	16	20	285	305	305	325	20
18	Relations validation tool	16	10	285	295	315	325	30
19	Study of term prevalence and co-occurrence	16	40	285	325	285	325	0
20	Project document	16, 17, 18, 19, 20	50	325	375	325	375	0
21	Project presentation	21	15	375	390	375	390	0
22	End	22	0	390	390	390	390	0