Universidad del País Vasco
Euskal Herriko Unibertsitatea
BILBOKO INGENIARITZA ESKOLA
ESCUELA DE INGENIERÍA DE BILBAO

GRADO EN INGENIERÍA INFORMÁTICA DE GESTIÓN Y SISTEMAS DE INFORMACIÓN

TRABAJO FIN DE GRADO

TEXT SIMILARITY. ESTUDIO DE LA SIMILARIDAD ENTRE CONCEPTOS MÉDICOS

DOCUMENTO 1- *MEMORIA*

Alumno/Alumna: Mateos Corral, Unai

Director/Directora (1): Atutxa Salazar, Aitziber

Director/Directora (2): Gojenola Gallebeitia, Koldobika

Curso: 2017-2018

Fecha: Bilbao, 28 de jun. de 2018

Resumen

El presente Trabajo Final de Grado expone los procesos realizados para, primero obtener en un formato *csv* la Clasificación Internacional de Enfermedades (CIE-10) -emitida por la OMS- contenida en un fichero *pdf* plano, generando un "diccionario" (en castellano) de las enfermedades y sus códigos CIE asociados contenidos en dicho fichero *pdf*. En este sentido, es importante puntualizar que esta clasificación es una unificación de los estándares internacionales más usados para obtener estadísticas de morbilidad y mortalidad en el mundo.

Esta estandarización sirve como referencia para intercambiar información clínica entre hospitales e, incluso, países. No obstante, resulta una tarea costosa realizar el etiquetado de los diagnósticos clínicos, puesto que los datos que quieren ser etiquetados se encuentran escritos en lenguaje poco estandarizado.Dicho de otro modo, para poder estandarizar los diagnósticos y asignarles su código correspondiente, es necesario ser una persona experta, aunque para ellas, de hecho, puede resultar farragoso.

La generación de este "diccionario" debe maximizar la recuperación de la información jerárquica implícita y la información auxiliar contenida. Como ejemplo explícito, en la figura 1.1 se muestran los datos tal cual se presentan en el documento *pdf*, y posteriormente una explicación de los aspectos más relevantes en su obtención:

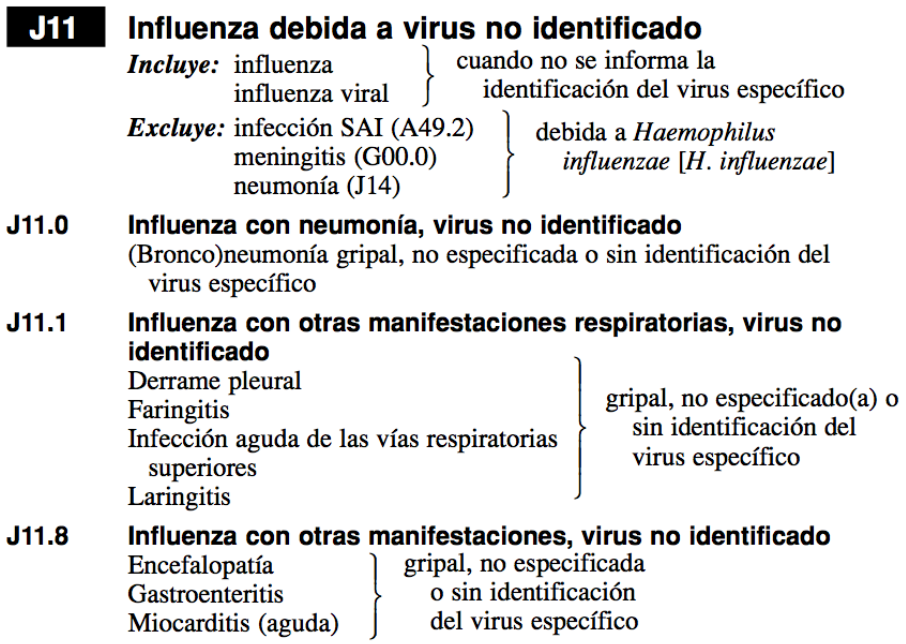


Figura 1: Muestra código J11 en documento CIE

Los códigos tienen un formato común, el cual comienza por una letra y, posteriormente, por dos dígitos. A estos códigos les puede corresponder un dígito o dos más, generando, así, los códigos de segundo y tercer nivel. Como podemos observar, en la figura 1.1 el código principal seria "J11" y los de segundo nivel "J11.0, J11.1 y J11.8". A cada uno de estos códigos les corresponde un término estándar,y , en ocasiones, disponen de información auxiliar. En cambio, los de tercer nivel no se encuentran en el documento de manera explícita como los demás, por ello se requiere una identificación de todos ellos, así como el término que les corresponde.

La información auxiliar puede aparecer de distintas maneras en el documento: por un lado, indicada por los términos *incluye* o *excluye*, como se observa en la imagen en los datos del código "J11", o por el otro, seguido del término estándar, como se puede apreciar en los códigos de segundo nivel. También, esta información puede contener especificaciones que se indican haciendo uso de llaves. Esta información se intercala de manera aleatoria entre los datos siendo necesaria una identificación manual.

Al extraer la información del documento, por motivos relacionados con la codificación del mismo y las herramientas utilizadas para su extracción, se requiere identificar los elementos que no se han extraído correctamente y proceder a su corrección. En ocasiones, estas correcciones se pueden realizar de forma automática, por ejemplo, los caracteres que no se hayan extraído correctamente -ya sean letras acentuadas o símbolos especiales- se identifican y se automatiza su corrección, como puede ser el carácter "á" que se obtiene como el símbolo "/".

Finalmente, tras el proceso de obtención y creación de los datos que forman el "diccionario", se aplica un preproceso para, así, obtener el documento en formato *csv* que contenga los datos con la mayor exactitud posible.

Tras este primer paso importante (la obtención de los datos del "diccionario"), el presente trabajo también se plantea como objetivo fundamental implementar una aplicación que sea capaz de asignar a un término de uso regular (normalmente un término no estándar) un código dentro de la clasificación CIE-10. Para llevar a cabo la asignación, primero, se identifica el termino estándar que muestre mayor similitud respecto del no-estándar de entrada, y luego se devolverá el código CIE del término estándar identificado como mejor candidato o candidatos con mayor similitud. La similitud-semejanza se calcula mediante la aplicación de distintas técnicas de similitud de textos, entre el string no-estándar y todos los estándar pertenecientes al "diccionario".

Asimismo, y finalmente, este proyecto también pretende medir la bonanza de las distintas alternativas propuestas para seleccionar la más adecuada dependiendo de la explotación posterior. Esta herramienta podrá ser utilizada para:

- automatizar completamente el proceso de identificación de enfermedades y asignación del código CIE correspondiente. De esta forma, la persona experta que a día de hoy realiza este proceso manualmente podría ser sustituido enteramente por la máquina.
- guiar y apoyar en el proceso manual identificando las enfermedades y proponiendo a la persona experta los códigos que puedan ser los mejores candidatos.

Con estas finalidades, se utiliza la herramienta "DKPro Similarity" y las distintas medidas de similitud entre cadenas de texto que proporcionan sus módulos *Lexical* y *LSA*.

Además, para obtener y realizar la evaluación y abarcar los intereses propuestos, es preciso puntualizar que se parte de dos corpus; es decir, dos conjuntos de textos médicos donde las enfermedades y sus códigos están correctamente identificados y asignados. Uno de los corpus ha sido etiquetado en castellano por el propio alumno (tomando como fuente documentos de la "Agencia Europea de Medicina" y artículos médicos), y el otro en francés (un corpus facilitado por los tutores del presente trabajo, *CepiDC Causes of Death Corpus* [1]). Así, estos corpus, y con el uso de "DKPro Similarity", sirven como patrón de oro para la evaluación de los resultados obtenidos.

Para evaluar los resultados obtenidos se realizan diferentes procesos. En un primer paso se evalúan todos los algoritmos con los datos de los corpus del castellano y francés, utilizando la figura de mérito *precission at k*.Ello permite calcular la precisión con la que se acierta según el número de los k resultados, donde k tomara los valores {1,5,7}. En un segundo apartado, se escogen los dos mejores algoritmos con el valor de k=1 y se procede a obtener un *threshold* con el cual se obtienen los mejores resultados según el valor de *f-score*, en este apartado únicamente se utilizaran los datos en francés al ser estos los mas completos.

Finalmente, el trabajo termina con unas conclusiones generales, que ayudan a plantear futuras líneas de trabajo y procesos diferentes, que ayudan a plantearse proyectos e investigaciones distintas.

Palabras clave: CIE, corpus, DKPro Similarity, similitud, término estándar, término no-estándar.

Laburpena

Gradu Amaierako Lan honek, lehen bidez, OMS igorritako “Clasificación Internacional de Enfermedades (CIE-10)”‘*csv* formatuan izan ahal izateko prozedurak aurkezten ditu. Modu horretan, *pdf* dokumentu bat erabiliko da sailkapen honen informazioa lortzeko. Dokumentuan aurkitzen diren gaixotasun guztien izena, informazioa eta kodigoak berreskuratuko dira, gaztelania “hiztegia” sortuz. Ildo honetatik, garrantzitsua da zehaztea sailkapen hau gehien erabiltzen diren estandar internazionalen batuketa bat dela, eta horrela, munduko erikortasun eta heriotza-tasen estatistikak izatea ahalbidetzen duela.

Estandarizazio hau ospitaleen artean, eta baita herrialde artean ere, informazio klinikoa partekatzeko erreferente bezala lekuratzen da. Hala ere, diagnostiko klinikoak etiketatzea jarduera gaitza da, etiketatu nahi diren datuak hizkuntza ez-estandarizatu batean idatzi egin baitira. Beste era batera esanda, datuak estandarizatu ahal izateko eta kodigo bat egokitzeko, aditu bat behar da, eta hauentzako ere jarduera korapilatsua izan ahal da.

“Hiztegiaren” eraikuntzak *pdf* dokumentuak jarraitzen duen hierarkia mantendu eta dokumentuan agertzen den informazioaren errekuperazioa maximizatu behar du. Horrela, adibide zehatz bezala, 2 irudian *pdf*-an agertzen diren datuak aurkezten dira, eta geroago bere lorketan izandako aspektu esanguratsuenak azaltzen dira.

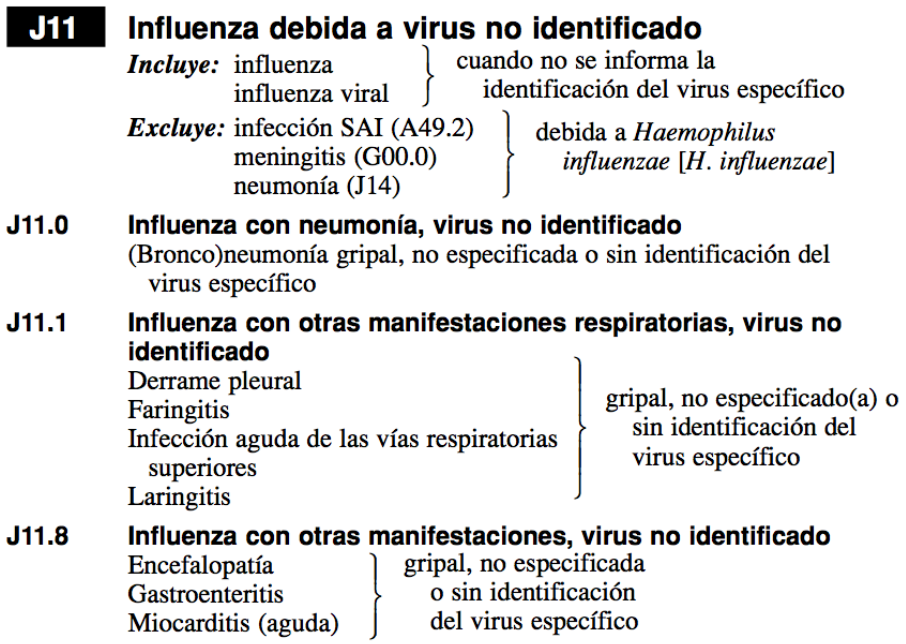


Figura 2: Muestra código J11 en documento CIE

Kodigoek eredu betegarria daukate, hau da, letra batekin hasten dira, eta, geroago, bi digituak dagokie. Kodigo hauei kodigo bat edo bi gehiagorekin gehiago erantsi ahal zaizkie, hau da, bigarren eta hirugarren mailako kodigoa sortuz.2 irudian behatu ahal dugun eran, kodigo printzipala "J11" izango litzake eta bigarren mailako kodigoak "J11.0, J11.1 y J11." izango lirateke. Kodigo bakoitzari termino estandar bat lotzen zaio, eta, batzuetan, kodigo berak informazio osagarria dauka. Aldiz, hirugarren mailako kodigoak, besteak ez bezala, dokumentuan ez dira aurkitzen modu esplizituan, eta hori dela medio, hirugarren mailako kodigo hauek, eta dagokion terminoa, identifikatu eta sortu egin behar dira.

Informazio osagarria dokumentuaren zehar modu desberdinetan aurkitu ahal da: alde batetik, *incluye* edo *excluye* terminoekin sailkatuta egon ahal da ("J11" kodigoaren datuetan agertzen den eran), eta bestetik, estandar terminoen ondoren agertu daiteke, bigarren mailako kodigoetan hauteman ahal den eran. Era berean, informazio honek giltzen bitartez agertzen diren espezifikazioak eduki ahal ditzake. Informazio hau datuen artean ausaz tartekatu ahal da, eta hortaz, eskuz egindako identifikazioa egin behar da.

Dokumentutik informazioa ateratzerakoan, kodifikazioa arazoak aurkitzen dira, datuak ateratzeko prozeduran erabilitako lanabesak direla eta. Horregatik, modu ez-egokian atera diren elementuak identifikatu eta zuzendu egiten dira. Batzuetan, zuzenketa hauek modu automatikoan egin ahal dira, adibidez, modu ez-egokian atera diren karaktereak –letrak edo sinbolo bereziak badira ere- esaterako, "á" karaktereak, "/" itxura hartzen du.

"Hiztegia" sortu ondoren, termino ez-estandarrei CIE-10-eko kodigo bat esartzeko aplikazioa implementatzen da. Egokipen hau aurrera eraman dadin, lehenbiziz, termino estandar eta ez-estandarraren arteko parekotasuna kalkulatu egiten da, eta parekotasun handiago aurkezten duen estandar terminoaren CIE kodigoa erantsi egiten da.

Xede hauek direla eta, "DKPro Similarity" lanabesa eta testuen arteko kateen parekotasuneko neurri desberdinen *Lexical* eta *LSA* moduluak erabiltzen dira.

Halaber, eta, azkenik, proiektu honek proposatutako alternatiben arteko emaitzak neurtu nahi ditu, ustiapenaren arabera egokienak hautatzeko. Tresna hau erabil daiteke:

- Gaixotasunen identifikazioa eta CIE kodigoen egokitzapenaren prozedura automatizatu. Modu horretan, aditua eskuz egindako prozedura hori, makina baten bitartez egian ahal egin ahal izango litzake, aditua ordezkatuz.
- Eskuz egindako gaixotasunen identifikazio prozesuan gidatu eta lagundu, adituari egokiagoak diren hautagaiak proposatuz.

Gainera, ebaluaketa egin ahal izateko eta behatzeko, bi corpus-etik abiatzen dela zehaztea ezinbestekoa da; hau da, bi testu medikoen batuketa erabiltzen dira, zeinetan gaixotasunak(termino ez-estandarrak) eta bere kodigoak zuzenki identifikatuta eta egokituta dauden. Corpus bat ikasle berak gazteleraz etiketatu behar izan du ("Agencia Europea de Medicina" eta medikuntzazko artikuluak iturri bezala izanez), eta beste Corpus-a frantsesez dago (lanaren tutoreek erraztu baitzuten, , *CepiDC Causes of Death Corpus* [1]).Horrela, corpus hauek, eta "DKPro similarit"' erabilerari esker, izandako emaitzen ebaluaketa egiteko urre patroi moduan balio dute.

Izandako emaitzak ebaluatzeko hainbat prozesu erabiltzen dira. Lehenengo pausu batean, gazteleraz eta frantsesez dauden datuekin algoritmo guztiak ebaluatzen dira, *precission at k* erabiliz. Bigarren pausu batean, K=1 balioarekin lortutako algoritmo hoberenak hautatzen dira eta *f-score* hoberena lortzen duen *threshold* bat bilatzeko prozedura aurrera eramaten da. Azken pausu honetan, frantsesez dauden datuak erabiltzen dira.

Azkenik, lana ondorio nagusiekin amaitzen da, zeintsuk iraganeko lan estrategiak eta prozedura desberdinak planteatzen dituztenak, ikerketa eta proiektu desberdin ildo orokorrak hausnartuz.

Hitz-gakoak: CIE, corpus, termino estadarra, termino ez-estandarra, parekotasuna.

Abstract

The current dissertation exposes the processes that have been carried out in order to, first, obtain The International Classification of Diseases (ICD 10) broadcast by WHO (World Health Organization) in *csv* format which is included in a plain *pdf* text document generating a (Spanish Dictionary) about the illnesses and their CIE codes which are also included in the mentioned *pdf* file.

Therefore, In this sense it is essential to point out that this classification is a unification of the most widely used international standards in order to obtain morbidity and mortality statistics in the world.

This standardization serves as a reference for exchanging clinical information between hospitals and, even, countries. However, it is a costly task to label diagnostics clinical trials, since the data that they want to be labeled are written in little standardized language. In other words, in order to standardize diagnoses and assign their code corresponding, it is vital to be an expert, although for them, in fact, it can result tedious.

The generation of this "Dictionary"should maximize the retrieval of implicit hierarchical information and the auxiliary information contained. As an explicit example, the data is given in the *pdf* document, and an explanation of the most relevant aspects to obtain it:

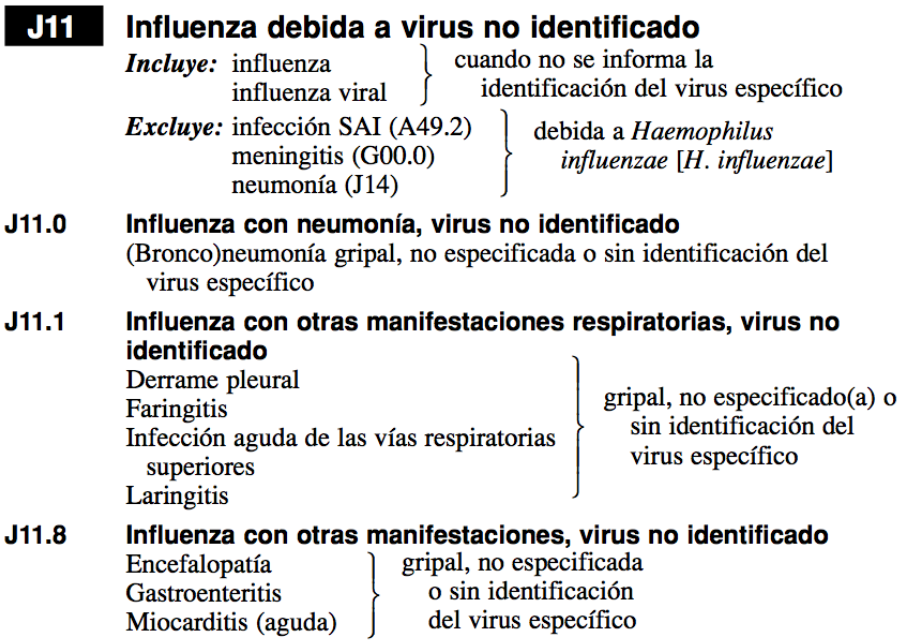


Figura 3: Muestra código J11 en documento CIE

The codes have a common format, which begins with a letter and, subsequently, by two digits, generating, in this way, the codes will be called principals. Sometimes, one or two more digit are assigned in these main codes, generating the codes that will be called as second and third level. As we can see, in the figure 1 the main code would be “J11” and the second level “J11.0, J11.1 and J11.8”. A standard term is corresponded to each of these codes, and sometimes they have auxiliary information. On the other hand, those of the third level are not explicitly found in the document like the others, for that reason an identification of all of them is required, as well as the term that corresponds to them.

The auxiliary information can appear in different ways in the document: on the one hand, indicated by the terms *includes* or *excludes*, as seen in the image in the code data “J11”, or on the other hand, followed by the standard term, as can be seen in the second codes level.In some circumstances keys are used in order to add auxiliary information, when the data is extracted, this information is inserted or positioned incorrectly, making it difficult to obtain data automatically. For this reason, to identify and obtain this information correctly is necessary to make manual revisions.

When the information is extracted from the document, for reasons related to the coding of the same and the tools used for its extraction, it is necessary to identify the elements that have not been extracted correctly and proceed correcting them. On occasion, these corrections can be done automatically, for instance, the characters that have not been extracted correctly - whether accented letters or special symbols - are identified and its correction is automated, for example “á” character is obtained as “/”.

After this vital step (obtaining the "dictionary"data), the current work also considers as a fundamental objective to implement an application capable of assigning to regular use term (usually a non-standard term) a code within the ICD-10 classification. To carry out the assignment, first, the standard term that shows more similarity with respect to the non-standard input, is identified and then the CIE code of the standard term identified as best candidate or candidates with greater similarity will be returned. The similarity-measure is calculated by applying different text similarity techniques, between the non-standard string and all the standards belonging to the “Dictionary”.

Therefore, and finally, this project also aims to measure the bonanza of the different proposed alternatives in order to select the most appropriate depending on the subsequent exploitation. This tool can be used to:

- fully automate the process of disease identification and allocation of corresponding CIE code. In this way, the expert who today makes this process manually could be replaced entirely by a machine.
- Guide and support the manual process by identifying diseases and proposing to the expert the codes that can be the best candidates.

For these purposes, the “DKPro Similarity” tool and the different measures of similarity between text strings that provide their lexical and LSA modules are used.

Furthermore, in order to obtain and carry out the evaluation and cover the proposed interests, it is necessary to point out that it is based on two corpora; that is, two sets of medical texts where diseases and their codes are correctly identified and assigned. One of the corpus as been labeled in Spanish by the student himself (taking as a source documents from the .Europeanan Agency of Medicine.and medical articles), and the other one in French (a corpus provided by the professors of the current work, , *CepiDC Causes of Death Corpus* [1]). Thus, these corpus serve as a gold standard for the evaluation of the results obtained with the tool "DKPro Similarity".

To evaluate the obtained results, different processes are carried out. In a first step, all the algorithms are evaluated with the data of the Spanish and French corpus , using the figure of merit precision at k and thus be able to analyze the precision according to the number of k results, where k will take the values {1,5,7}. In a second section, once these results have been calculated, the best 2 algorithms are chosen with the value of k = 1 and proceed to obtain a threshold that obtains the best f-score, in this section only the data in French will be used as these are the most complete ones.

Eventually, the work ends with some general conclusions, which help to pose future lines of work and different processes that can be carried out in projects and research.

Keywords: CIE, corpus, DKPro Similarity, similarity, standard term, term no-standard

Índice General

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

1.1. Introducción

A día de hoy gran parte de los textos médicos, como por ejemplo los informes médicos, contienen terminología no estándar, como formas más o menos cultas de nombrar las enfermedades, abreviaturas, textos con errores ortográficos etc. De este modo, en las últimas décadas, se ha hecho un gran esfuerzo en vías de la internacionalización y estandarización de dichos textos. La OMS (Organización Mundial de la Salud) ha liderado este proceso proponiendo una clasificación internacional de las enfermedades, donde cada enfermedad tiene asociado un código internacionalmente aceptado y único. La codificación de las enfermedades permite mejorar, por un lado, el intercambio de información entre diferentes servicios de salud, ya sean de una misma región, y país, o entre servicios de diferentes países; y, por el otro, permite llevar a cabo estudios sobre grandes volúmenes de información que antes dada la heterogeneidad de los datos no era posible realizar. Estos estudios tienen como objetivo último mejorar la detección de síntomas, una intervención más precoz de las causas de una determinada enfermedad, etc.

Como se ha dicho previamente, y para recoger las diferentes enfermedades existentes hasta la fecha, la OMS elabora y actualiza regularmente un documento llamado Clasificación Estadística Internacional de Enfermedades y Problemas Relacionados con la Salud "CIE" sus siglas en castellano o International Statistical Classification of Diseases and Related Health Problems "ICD" en inglés. Actualmente, es una clasificación de los estándares internacionales mas usados para obtener estadísticas de morbilidad y mortalidad en el mundo, "con el propósito de permitir el registro sistemático, el análisis, la interpretación y la comparación de los datos de morbilidad y mortalidad en diferentes países o áreas, y diferentes momentos. La clasificación permite la conversión de los términos diagnósticos y de otros problemas de salud, de palabras a códigos alfanuméricos que facilitan su almacenamiento y posterior recuperación para el análisis de la información". Los volúmenes que comprenden cada clasificación, pueden encontrarse fácilmente en la red.

Este ayuda 1) a unificar la terminología escrita sobre las enfermedades; y 2) facilitar la identificación de una enfermedad concreta al personal médico. Sin embargo, para el manejo diario del documento, teniendo en cuenta el colapso, especialmente, de la salud pública, se requieren de instrumentos tecnológicos que ayuden y agilicen este primer paso.

La mayoría de los diagnósticos realizados suelen contener síntomas que el doctor ha detectado al analizar al paciente, así como las principales dolencias que este indica y finalmente la causa de estas. Por ello, la obtención del código correspondiente a la clasificación oficial o el acotamiento de

los posibles códigos que puedan hacer referencia a dicha información, será el problema que aborde este proyecto.

Para ello, como se ha mencionado previamente, será necesario el uso del documento CIE, en concreto, su versión 10, obteniendo la máxima información posible de este documento, y así poder obtener mejores resultados. La generación de este "diccionario" debe maximizar la recuperación de la información jerárquica implícita y la información auxiliar contenida.

Esta información jerárquica, así como, la información auxiliar requiere de un análisis exhaustivo, ya que en el momento en el que se extraen los datos del documento *pdf* plano, la jerarquía se llega a perder en ocasiones y la información auxiliar se intercala entre otros datos. Por tanto, es necesario realizar todo lo posible para identificar estos casos y crear el "diccionario" lo más exacto posible al documento original.

Con el siguiente ejemplo se expone la jerarquía implícita y como se encuentran los datos en su origen.

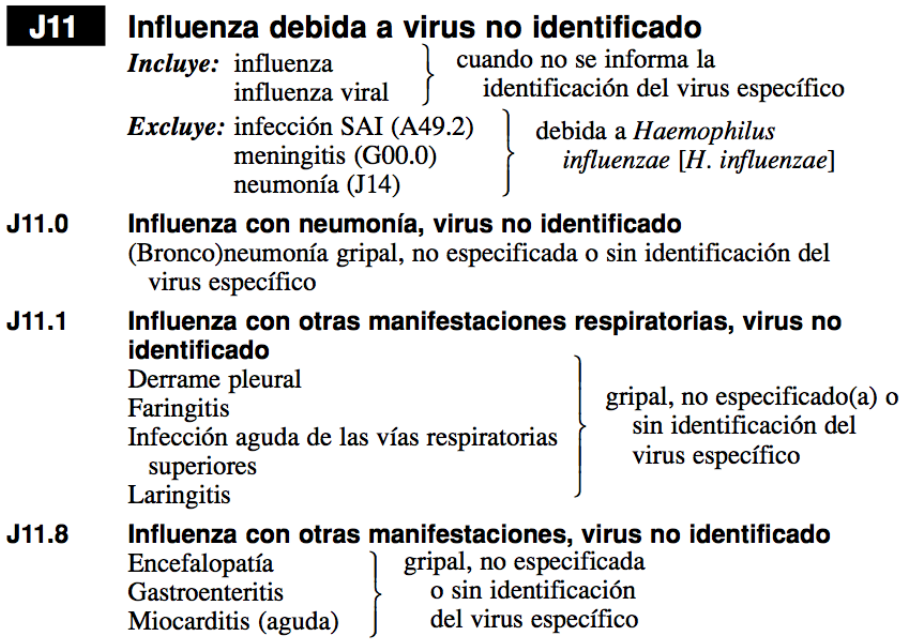


Figura 1.1: Muestra código J11 en documento CIE

En la imagen se puede identificar cómo los datos del documento *pdf* siguen una jerarquía definida por los propios códigos. Así, cuyo formato está comprendido por una letra y, posteriormente, por dos dígitos, en el ejemplo "J11" se denominan como códigos principales. Pero, como se observa, a los códigos principales les puede corresponder un dígito más, generando, así, los códigos a los que denominaremos de segundo nivel (en este ejemplo los códigos "J11.0, J11.1 y J11.8"). A cada uno estos códigos, les corresponde el término estándar correspondiente a la enfermedad a la que hacen referencia e información auxiliar.

En algunos casos, al igual que sucede con los códigos principales, a los de segundo nivel también les puede corresponder otro dígito, generando, así, los códigos que denominaremos de tercer nivel. Estos no se encuentran en el documento de manera explícita, por ello se requiere una identificación de todos ellos, así como el término que les corresponde.

La información auxiliar aparece a lo largo del documento de distintas maneras, y al extraer los datos, en la mayoría de ocasiones, no mantiene el orden original, intercalándose o posicionándose incorrectamente. Ello dificulta la obtención de los datos de forma automática y, por tanto, se requiero una identificación para poder obtenerla correctamente.

En ocasiones, este trabajo no resulta complicado, puesto que la identificación de los distintos códigos y los datos que les pertenecen se mantienen entre un código y el siguiente. Estos, en concreto, son indicados a través de las palabras clave excluye o incluye. En otros casos, en cambio, la información se intercala de forma errónea y requiere de intervención manual para establecerla correctamente. Como se muestra en la figura 1.2:

J11.1Influenza con otras manifestaciones respiratorias, virus no
identificadoDerrame pleural
FaringitisInfecciŠn aguda de las v™as respiratorias
superiores LaringitisJ11.8Influenza con otras manifestaciones, virus no identificado
Encefalopat™a Gastroenteritis Miocarditis (aguda)
CLASIFICACIîN INTERNACIONALDE ENFERMEDADES
484!"""#"""$debido(a) a otro virus de la
influenza identificado
!"#"$debida a otro virus de la influenza
identificado debida a
Haemophilusinfluenzae [H. influenzae
]gripal, no especificado(a) o
sin identificaciŠn del
virus espec™fico
gripal, no especificada
o sin identificaciŠn
del virus espec™fico
cuando no se informa la
identificaciŠn del virus espec™fico
!#$!"#"$!""#""$!"#"$debida a
Haemophilusinfluenzae [H. influenzae
]!"#"$

Figura 1.2: Muestra inicial de los datos extraídos del código J11.1

Como se ha resaltado en la imagen, los datos subrayados son aquellos que corresponden al código “J11.1”. Tanto el término estándar, como la información auxiliar se encuentran entre el propio código y el siguiente “J11.8”. En cambio, la información del documento indicada a través de llaves, se encuentra mas adelante también subrayada.

Además, a lo largo del texto extraído aparecen caracteres o símbolos erróneamente codificados, procediendo a su detección y corrección. En la siguiente tabla se pueden observar algunos de estos:

Carácter Extraído	Carácter Original	Carácter Final
/	á	a
Ç	Á	A
“	é	e
...	É	E
TM	í	i

Tabla 1.1: Ejemplo codificación caracteres.

Finalmente, una vez solventados todos estos escoyos, se obtienen los datos como se puede observar el ejemplo correspondiente al código “J11.1”:

J11.1 ;Influenza con otras manifestaciones respiratorias, virus no identificado
;Derrame pleural gripal, no especificado o sin identificacion del virus especifico
;Faringitis gripal, no especificado o sin identificacion del virus especifico
;Infeccion aguda de las vias respiratorias superiores gripal, no especificado o sin identificacion del virus especifico
;Laringitis gripal, no especificado o sin identificacion del virus especifico

Figura 1.3: Resultado final datos código J11.1

1.2. Motivación

Poder identificar una enfermedad dentro de una clasificación internacional implica que, con un simple código -sin tener la necesidad de compartir el idioma-, sea posible conocer la enfermedad en concreto de la que se trata, ya sea por información, como necesidad (en caso de que una persona tenga que ser atendida en un país extranjero), o realizar estudios a nivel mundial.

Por otro lado, para el ámbito de la salud supone un valor añadido poder agudizar los procesos de detección, ya que conocer cuál es la enfermedad que ha superado o padece un paciente de forma rápida, puede permitir disminuir el coste tanto monetario, como de tiempo que implica tener que analizar los diagnósticos, recetas, partes médicos... Por tanto, esta premura en la codificación del diagnóstico supondría beneficios sustanciales en las dimensiones bio-sociales y económicas que envuelven al sistema de salud.

Siendo conscientes de que el sistema de salud no funciona de la misma forma en todos los lugares, y que en algunos casos el ámbito privado toma un peso importante, como es el caso de las aseguradoras, ya que estas, en ocasiones, deben pagar a un cliente un coste relacionado por razones de enfermedad, de manera que, facilitar el reconocimiento de estas, sería posible disminuir los costes.

1.3. Definiciones, acrónimos y Abreviaturas

Dado que el proyecto queda limitado al ámbito sanitario, y, a su vez, se maneja información técnica relacionada con la informática, con el fin de favorecer la comprensión del proyecto, se crea este apartado que delimita y define acrónimos y términos que son utilizados de aquí en adelante:

- CIE: Clasificación Internacional de Enfermedades
- ICD: International Classification of Diseases
- CIE-10: Se hace referencia a la Clasificación Internacional de Enfermedades en su versión 10
- Diseases and Related Health Problems
- TFG: Trabajo Fin de Grado
- IDE: Entorno de Desarrollo Integrado
- LSA: Latent Semantic Analisis
- Corpus: Banco de datos, en este proyecto aquellos datos formados por término no-estándar que hagan referencia a información clínica correctamente etiquetada con su código CIE.
- csv: Comma-separated values, es un formato de archivo.
- txt: formato de archivo cuyas siglas en ingles hacen referencia a "textfile"(archivo de texto).
- MSSSI: Ministerio de Sanidad, Servicios Sociales e Igualdad (España).
- Subespecificación: palabra que referencia a problemas que pueden surgir a la hora de trabajar con texto. En este caso, surge a la hora de tratar de identificar un término no-estándar con respecto a un término estándar del CIE. Es decir, en el documento CIE los términos estándar hacen referencia a una enfermedad de forma concreta, en cambio, mediante un término no-estándar, el cual esta escrito en lenguaje natural de forma espontanea, puede hacer referencia a una enfermedad que aparezca en el CIE pero mediante su contexto. Por ejemplo, en documento CIE-10 aparece la “cefalea” con su código correspondiente, pero mediante un término no-estándar se puede escribir “dolor de cabeza agudo” generándose así una subespecificación.

Capítulo 2

Planteamiento Inicial

2.1. Descripción del Proyecto

El proyecto que se desarrolla se encarga de obtener del conjunto de los datos que forman el CIE, en este caso la versión 10, el código o "k" códigos correspondientes a la enfermedad que mayor similitud tengan respecto a un término no-estándar que representa una enfermedad de la cual se desconoce su código. Para ello, como objetivo principal se aplicarán diferentes herramientas de similitud de términos, y como objetivo secundario y el tiempo lo permite redes neuronales.

Asimismo, para el desarrollo del propio proyecto se precisa, por un lado, un "diccionario", que en este caso es la lista de los códigos CIE-10 y su información, y por el otro, serian los diagnósticos, términos escritos en lenguaje no estándar, descritos por los médicos y correctamente etiquetados. En lo que hace referencia a estos diagnósticos médicos, son datos personales protegidos por la ley de protección de datos, y, por tanto, se procede a crear un corpus, con un contenido médico que simule estos datos protegidos.

También, se procede a la creación de un nuevo diccionario CIE-10 en formato *csv*, que contenga la mayor cantidad de información posible de la Clasificación Internacional de Enfermedades emitida por la Organización Mundial de la Salud. Además del código y del término estándar con el que se hace referencia a la enfermedad, al analizar los datos del documento se puede observar que una gran parte de los códigos contienen información auxiliar que puede ser de utilidad para mejorar los resultados del proyecto, y por ello, se decide obtenerla. Esta información se encuentra a lo largo del documento a través de dos palabras clave *excluye* e *incluye*, o a tras el código y término estándar al que corresponde.

2.2. Objetivos del Proyecto

2.2.1. Objetivos Principales

En consecución de los intereses detallados, el presente proyecto parte de los siguientes objetivos:

- Obtener la mayor cantidad de información disponible de la clasificación CIE-10, para generar un "diccionario" que contenga la mayor información posible de las enfermedades clasificadas en esta lista.

- Implementar una aplicación que sea capaz de obtener la similitud existente entre dos términos, mediante la aplicación de distintas técnicas de similitud de textos, de manera que se pueda obtener el código que le corresponde del CIE-10 al término no-estándar que contiene la información médica.

- Realizar una evaluación de los resultados que se obtienen al aplicar las técnicas de similitud de texto. Para, así, poder determinar el funcionamiento de estas.

2.2.2. Objetivos personales

El proyecto se propone con distintos y muy diversos retos. Un trabajo que, sin duda, permite afianzar los conocimientos adquiridos durante los años de recorrido académico, como puede ser el lenguaje de programación de "Java". Pero, el proyecto, no únicamente refuerza algunos conocimientos, sino que alberga otras cuestiones interesantes e importantes como el aprendizaje del lenguaje de programación "Python" y el tratamiento de texto, enfrentándose a problemas reales derivados del uso de la lengua, como problemas de normalización, variabilidad, errores en el uso de la propia lengua, así como la ambigüedad que se puede ocasionar en esta o la subespecificación.

De igual modo, la minería de datos es un ámbito que recién comienza a ser integrado en análisis de datos informáticos, lo que supone adentrase en un aprendizaje de gran utilidad para el futuro, tanto para el personal, como profesional.

Finalmente, esta investigación contiene retos importantes para el avance de la asistencia sanitaria, lo que se convierte en un importante propulsor de motivación. Las investigaciones aplicadas al ámbito socio-sanitario permite generar beneficios que ayudan en el avance de una ciudadanía más sana.

Por todo ello, se considera que este proyecto es un recorrido y proceso enriquecedor para poner fin a este recorrido académico.

2.3. Arquitectura

Para implementar este proyecto, se va a hacer uso de una arquitectura local. En esta arquitectura, tanto el software, como los datos de los que se hacen uso, se encuentran en el mismo sistema.

Dependiendo de la privacidad de los datos que se vayan a utilizar, existe la posibilidad de instalar la aplicación en un servidor en el cual se encuentren estos datos, y acceder a él de manera remota, ejecutando la aplicación en el servidor que contiene los datos.

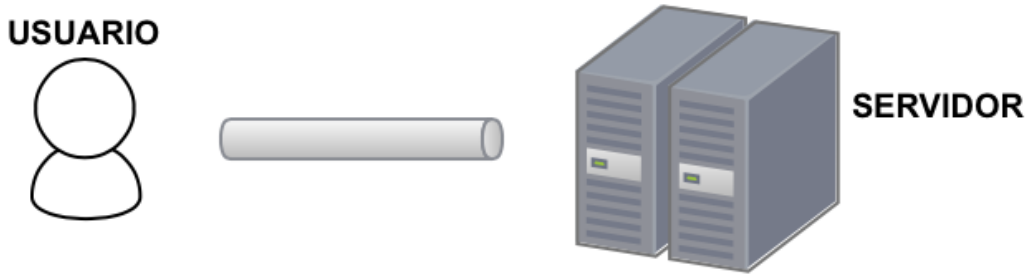


Figura 2.1: Usuario-Servidor

Para facilitar el uso de la aplicación, se facilita una interfaz de usuario. Para ello, se utiliza la arquitectura Modelo Vista Controlador (MVC). Esta gestiona la aplicación en tres estructuras, una se encarga del acceso a los datos, la segunda de interactuar con el usuario, y la tercera de controlar el acceso a los datos según las órdenes del usuario.

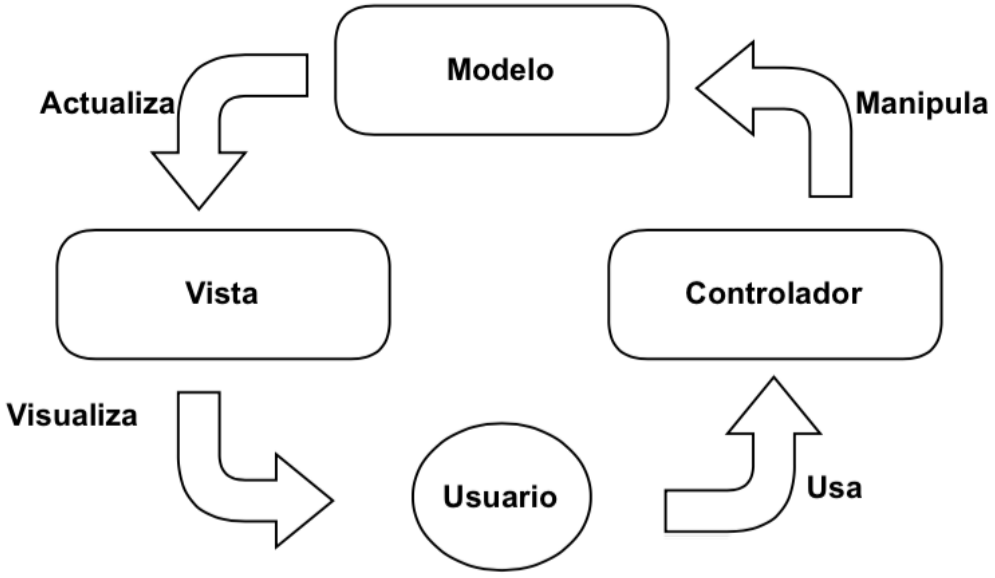


Figura 2.2: Diagrama modelo-vista-controlador

2.4. Alcance del Proyecto

2.4.1. Ciclo de Vida

El proyecto se desarrolla de forma interactiva e incremental. Es decir, se divide el proyecto en prototipos o módulos que se encargan de una tarea concreta, añadiendo al sistema y desarrollando dicho módulo en cada una de las interacciones. De este modo, se obtiene un sistema que añade de manera incremental las funcionalidades necesarias para su correcta ejecución.

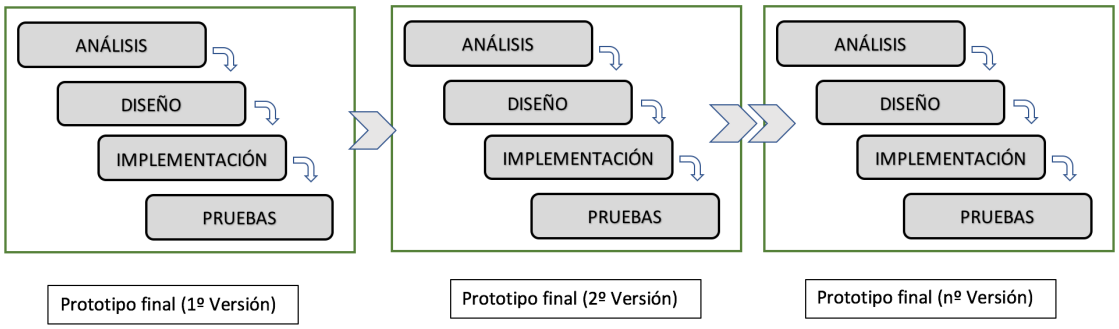


Figura 2.3: Ciclo de vida iterativo

Esta metodología, en concreto, es utilizada debido a las siguientes razones:

- Facilita el control de pequeños módulos, y permite, mediante el desarrollo individual de estos, ir incrementando pausadamente las funciones de la aplicación.
- En el caso de cometer errores, estos únicamente pertenecerán al módulo en cuestión, facilitando la solución de estos e impidiendo su continuación.
- Facilita el seguimiento de los resultados, así como la calidad de cada uno de los prototipos.

2.4.2. Prototipos

El proyecto consta de los siguientes prototipos:

- Prototipo 1: Obtención, Creación Y Prepocesamiento de los Datos.

  Los datos se encuentran en un archivo en formato *pdf*, por lo que se procede a la extracción de dichos datos. Estos se analizan, para identificar los que son de necesarios para el diccionario, almacenándolos finalmente en un archivo en formato *csv* con una determinada estructura. Finalmente, obtenido el diccionario final, se procede a su preprocesamiento.

- Prototipo 2: Obtención de similitudes Mediante “DKPro Similarity” *lexical*.
  Haciendo uso de la herramienta “DKPro Similarity”, este prototipo se centra en comprender y ejecutar las diferentes técnicas que ofrece y son útiles para el objetivo del proyecto, y as obtener sus resultados.

- Prototipo 3: Obtención de similitudes Mediante “DKPro Similarity” LSA.
  Este prototipo, se centra en crear los datos necesarios para ejecutar el módulo *LSA* que incluye “DKPro Similarity” y obtener sus resultados.

- Prototipo 4: Exportar y evaluar resultados.
  Este prototipo se encarga de exportar los resultados obtenidos, y de la evaluación de los resultados.

2.4.3. Etapas

- Planificación Inicial:
  La primera tarea consiste en delimitar claramente el alcance del proyecto.

  Se realiza una planificación del trabajo y de los tiempos que se estiman necesarios para realizar cada una de las tareas. También, se procede a realizar la planificación temporal estimada del proyecto, mediante el uso del diagrama EDT, Gantt....

- Aprendizaje:
  Esta etapa está centrada en la búsqueda de las diferentes técnicas y herramientas que se aplican durante el proyecto, y a la formación necesaria para poder hacer uso de ellas.

- Análisis y Diseño:
  Se procede a identificar los diferentes objetivos que le corresponde a cada uno de los prototipos. Una vez determinados, se inicia un proceso para identificar las diferentes herramientas y técnicas necesarias para la implementación de cada uno de ellos. Para finalmente realizar el diseño del prototipo.

- Implementación:
  En esta etapa se aplican las diferentes técnicas y herramientas previamente estudiadas para realizar la implementación diseñada de cada prototipo.

- Documentación:
  A medida que se va avanzando con el desarrollo del proyecto, así como con sus prototipos y etapas, se realiza la documentación correspondiente. Esta información es la que conforma la memoria del proyecto.

2.5. Herramientas

2.5.1. Hardware

Portátil Mac Book Pro Retina

2.5.2. Software

Herramientas para la documentación:

- macOS Sierra, concretamente la versión 10.12.6, sistema operativo en el que se desarrolla el proyecto.

- Overleaf: Procesador de textos de código abierto que emplea LaTex, y facilita su uso y se muestra de una forma amigable para el usuario. Además, reproduce en tiempo real y de manera automática el documento que se está generando.

- Ganttproyect: herramienta gratuita que facilita la creación de los diagramas Gantt de gestión de tiempos.

- www.cacoo.com : herramienta online gratuita para el diseño de diagramas, usada para generar el diagrama EDT.

- Visual Paradigm Comunity edition: Herramienta para generar los diagramas de casos de uso, diagramas de clases y de secuencia.

- Sublime Text 3: editor de texto libre dirigido al desarrollo de código.

Herramientas para el desarrollo:

- Edición electrónica de la clasificación CIE-10, `http://eciemaps.msssi.gob.es/ecieMaps/browser/index_10_2008.html`

- Eclipse IDE: software para desarrollar código java.

- Librerías externas Python:
  - PyPDF2: para la transformación de textos en formato *pdf* a *txt*.
  - NLTK: facilita el tratamiento de "StopWords" y generación de "n-gramas".

- "DKPro Similarity": es un framework de código abierto para similitud de textos, y cuyo objetivo es proporcionar un repositorio completo de medidas de similitud de textos que se implementan utilizando interfaces. Se implementarán en el proyecto usando las siguientes librerías que proporciona:
  - módulo *lexical*: contiene los distintos algoritmos a usar.
  - módulo *LSA*, para poder aplicar a los datos el algoritmo que gestiona dicho módulo.

- Librerías externas Java: son requeridas por los módulos de DKPro utilizados.

2.6. Planificación Temporal

2.6.1. Diagrama EDT

A continuación se puede ver el diagrama correspondiente a la estructura de desglose del trabajo(EDT).

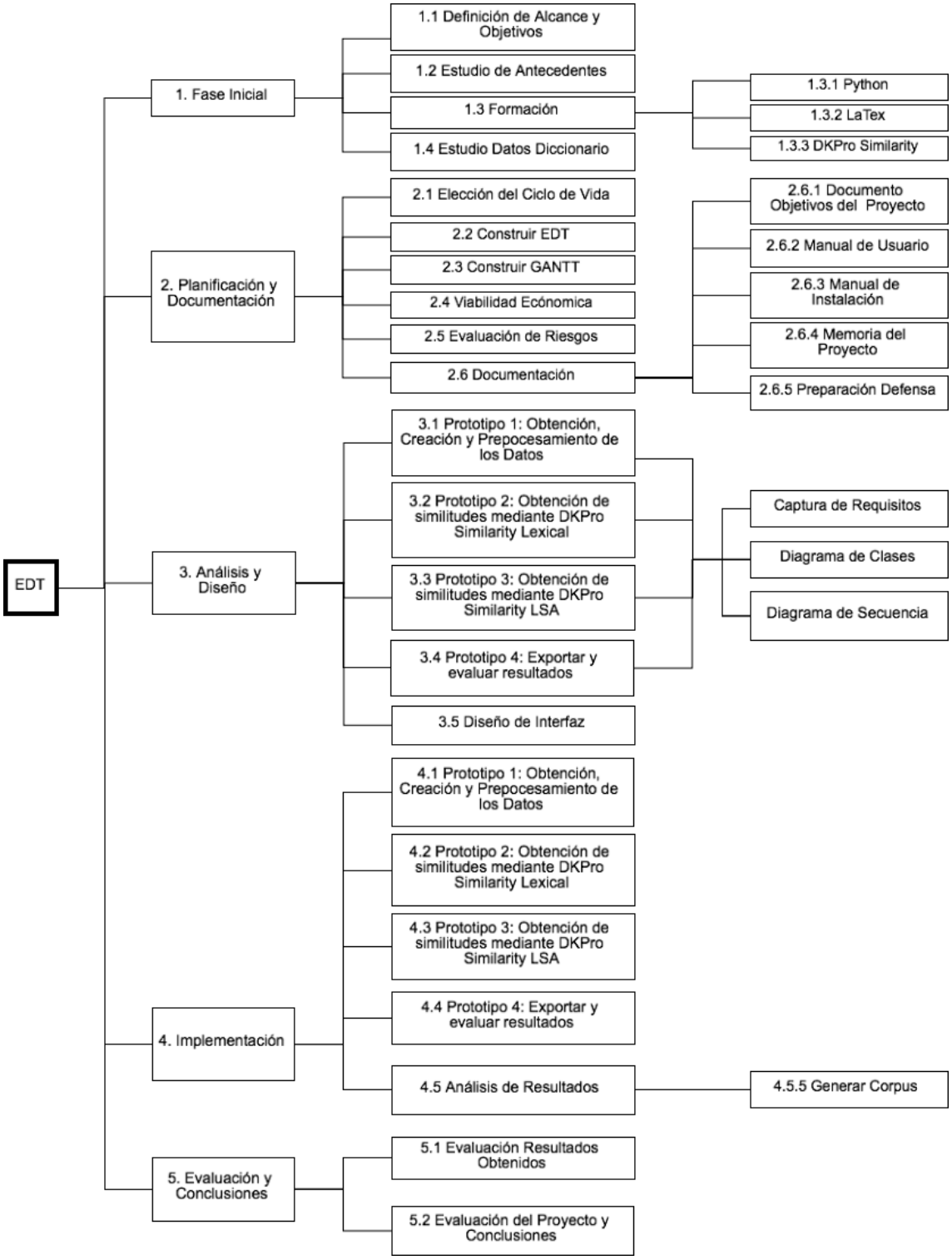


Figura 2.4: Diagrama EDT

2.6.2. Descripción de Tareas

Tarea 1 Fase Inicial

A continuación, se desglosan las tareas empleadas en la fase inicial del proyecto:

- **Tarea 1.1 Definición de alcance y objetivos**
  Se definen los objetivos y el alcance del proyecto.

- **Tarea 1.2 Estudio de Antecedentes**
  Se realiza una análisis de técnicas, de estudios y proyectos previos que sean similares al proyecto, y resulten significativos para los objetivos planteados. Al ser un proyecto que utiliza métodos que comienzan a ser empleados, no se cuenta con un gran número de fuentes, y las pocas que se hayan son de origen anglosajón.

- **Tarea 1.3 Formación**
  Esta etapa especialmente sirve para profundizar y conocer las técnicas y herramientas que se utilizan. Esta etapa resulta sumamente importante, ya que el buen desarrollo del proyecto depende del conocimiento y el manejo de ellas.

  - **Tarea 1.3.1 Python**
    Primero, familiarizarse con el aprendizaje del lenguaje de programación Python.

  - **Tarea 1.3.2 LaTex**
    Segundo, se aprende el uso de LaTex, para poder realizar el desarrollo de la documentación en dicho formato.

  - **Tarea 1.3.3 "DKPro Similarity"**
    Tercero, se analizan las posibilidades que nos ofrece "DKPro Similarity", así como los algoritmos que nos proporciona, para seleccionar de entre ellos los que se van a aplicar.

- **Tarea 1.4 Estudio datos CIE**
  Para la creación del diccionario, es decir, para decidir los datos que se van a extraer del documento original y conocer su estructura se realiza un análisis de los datos que contiene el documento CIE.

Tarea 2 Planificación

Etapa que comprende toda la duración del proyecto, ya que consta de todas las tareas de planificación y documentación, desde la redacción del documento de objetivos del proyecto, hasta la redacción de la memoria, las evaluaciones y conclusiones finales.

- **Tarea 2.1 Elección del ciclo de vida**
  Se define el ciclo de vida en el que se desarrolla el proyecto.

- **Tarea 2.2 Construir EDT**
  Se desglosan las distintas tareas en las que se divide el proyecto, haciendo una breve descripción de ellas.

- **Tarea 2.3 Construir GANTT**
  Con las tareas descritas en el diagrama EDT, se procede a realizar una estimación temporal de la duración de cada una de ellas.

- **Tarea 2.4 Viabilidad económica**
  En esta etapa se desarrolla la estimación de los costes y beneficios que puede suponer el desarrollo del proyecto, teniendo especialmente en consideración los aspectos materiales, como las horas de trabajo.

- **Tarea 2.5 Evaluación de riesgos**
  Etapa para realizar una estimación de los riesgos que pueden surgir a lo largo del desarrollo del proyecto. Asimismo, se genera un plan de prevención y contingencia para cada uno de ellos con el fin de minimizar su impacto.

- **Tarea 2.6 Documentación**
  Este proceso consiste en detallar los procesos realizados a lo largo del proyecto, y documentar tanto los procesos seguidos para alcanzar los objetivos, como las decisiones tomadas a lo largo de su desarrollo, las dificultades que han surgido y la manera de solventarlas, las conclusiones, etc.

  - **Tarea 2.6.1 Documento de objetivos del proyecto**
    Se describen los objetivos del proyecto y la estrategia para llevarlos a cabo.

  - **Tarea 2.6.2 Manual de Usuario**
    Se describe los procedimientos concretos para poder hacer uso de la aplicación y sus funcionalidades.

  - **Tarea 2.6.3 Manual de Instalación**
    Se describen los distintos requerimientos para hacer uso de la aplicación de una manera correcta, y poder ejecutarla.

  - **Tarea 2.6.4 Memoria del proyecto**
    En esta etapa se describe el proceso seguido en el desarrollo del proyecto, y finalizar con unas conclusiones generales.

  - **Tarea 2.6.5 Preparación defensa**
    Se prepara la defensa del proyecto, realizando la correspondiente presentación y sus respectivos ensayos.

Tarea 3 Análisis y Diseño
En esta etapa se realiza el análisis y el diseño de cada uno de los prototipos

- **Tarea 3.1 Prototipo 1: Obtención, Creación y Prepocesamiento de los Datos**
  En este prototipo se extraen los datos del documento CIE-10 requeridos para la creación del diccionario, así como el preprocesamiento de los datos.

- **Tarea 3.2 Prototipo 2: Obtención de similitudes Mediante "DKPro Similarity" *lexical***
  Este prototipo tiene como objetivo obtener la similitud entre el término no-estándar y el estándar, mediante la aplicación de distintos algoritmos que facilita el módulo *lexical*. También, en este prototipo se obtienen los datos necesarios para poder ejecutar los algoritmos.

- **Tarea 3.3 Prototipo 3: Obtención de similitudes Mediante "DKPro Similarity" *LSA***
  Este prototipo, al igual que el anterior, se encarga de obtener la similitud entre los términos, siendo necesario en este caso obtener y crear los datos necesarios para su funcionamiento.

- **Tarea 3.4 Prototipo 4: Exportar y evaluar resultados**
  A partir de los datos obtenidos en los pasos anteriores, se obtiene los resultados.

- **Tarea 3.5 Diseño de Interfaz:**
  Se diseña una interfaz sencilla para que un usuario base sea capaz de ejecutar los distintos algoritmos.

Tarea 4 Implementación
En este apartado se procede a realizar la implementación de los prototipos indicados en el apartado anterior.

- **Tarea 4.1 Prototipo 1: Obtención, Creación y Prepocesamiento de los Datos**
  Se implementan los distintos pasos necesarios para obtener los datos del documento *pdf* que forman parte del "diccionario" y realizar su preprocesado.
  También, se implementara el preproceso a realizar a los términos no-estándar, eliminación de *stop words*, caracteres especiales y símbolos.

- **Tarea 4.2 Prototipo 2: Obtención de similitudes mediante "DKPro Similarity" *lexical***
  Implementación del prototipo que obtiene la similitud entre términos con el uso de los algoritmos que ofrece el módulo *lexical* de "DKPro Similarity".

- **Tarea 4.3 Prototipo 3: Obtención de similitudes mediante "DKPro Similarity" *LSA***
  Implementación del prototipo que obtiene la similitud entre términos a través del uso de los algoritmos que ofrece el módulo *LSA* de "DKPro Similarity".

- **Tarea 4.4 Prototipo 4: Exportar y evaluar resultados**
  Se implementa el apartado que exportara los resultados en archivos de texto y el apartado que evalué los resultados obtenidos si así se requiere.

- **Tarea 4.5: Análisis de Resultados**
  Se realiza un análisis de los resultados.

  - **Tarea 4.5.1: Generar Corpus**
    Para realizar un análisis de los resultados se genera un corpus.

Tarea 5 Evaluación y Conclusión
Se procede a la evaluación del funcionamiento de la aplicación y los resultados obtenidos.

- **Tarea 5.1 Evaluación del proyecto y conclusiones**
  Se realiza una evaluación detallada de todo el proyecto, así como la obtención de las conclusiones.

2.6.3. Diagramas GANTT

Diagrama de Gantt que representa gráficamente los intervalos de las tareas previamente descritas.(2.5).

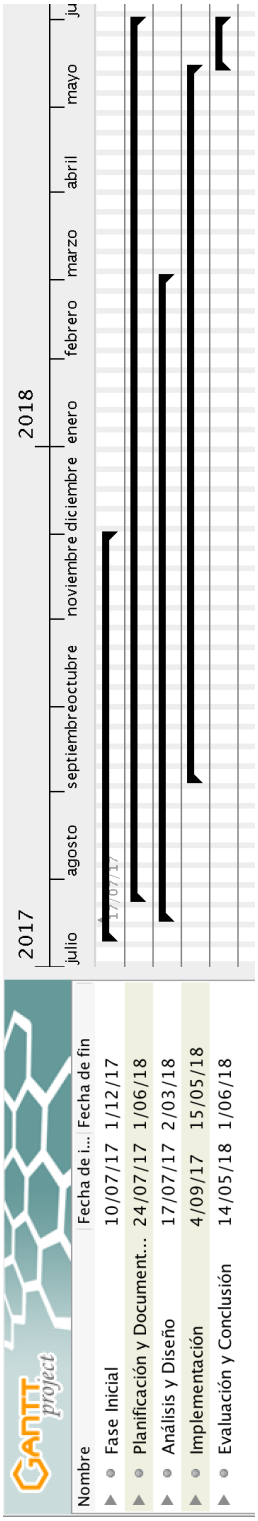


Figura 2.5: Diagrama Gantt General

Diagrama de Gantt que representa gráficamente las subtareas que pertenecen a la tarea Fase Inicial.(2.6).

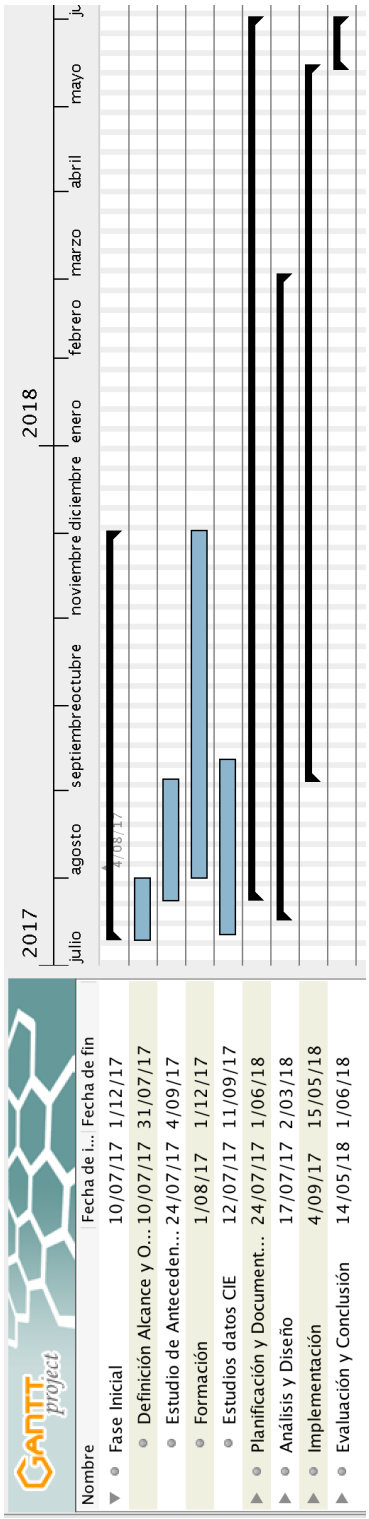


Figura 2.6: Diagrama Gantt Fase Inicial

Diagrama de Gantt que representa gráficamente las subtareas que pertenecen a la tarea Planificación.(2.7).

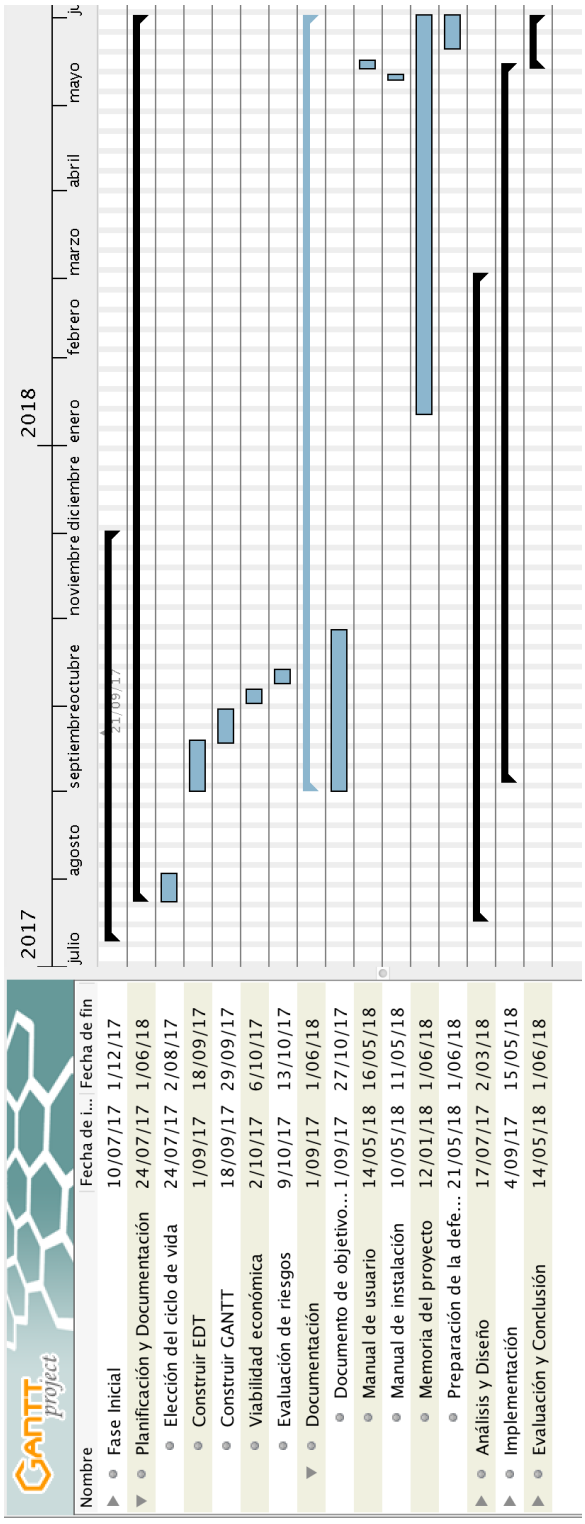


Figura 2.7: Diagrama Gantt Planificación

Diagrama de Gantt que representa gráficamente las subtareas que pertenecen al análisis y diseño.(2.8).

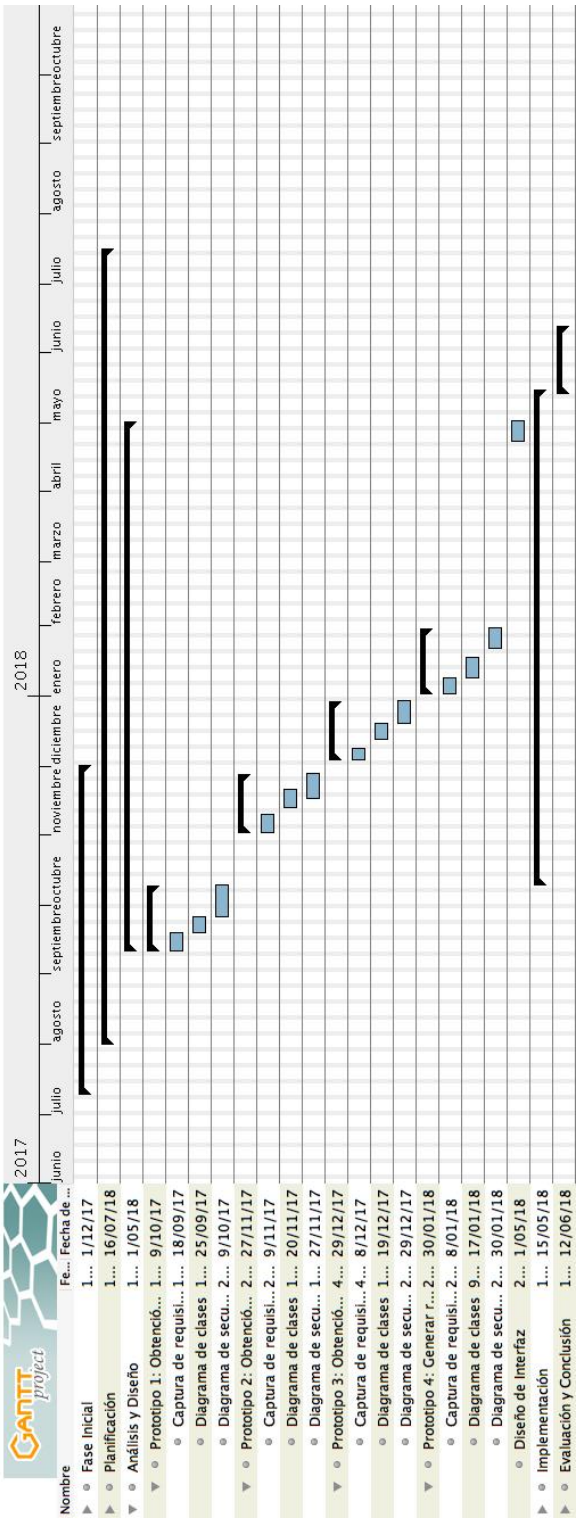


Figura 2.8: Diagrama Gantt Análisis y Diseño

Diagrama de Gantt que representa gráficamente las subtareas que pertenecen a la tarea de implementación.(2.9).

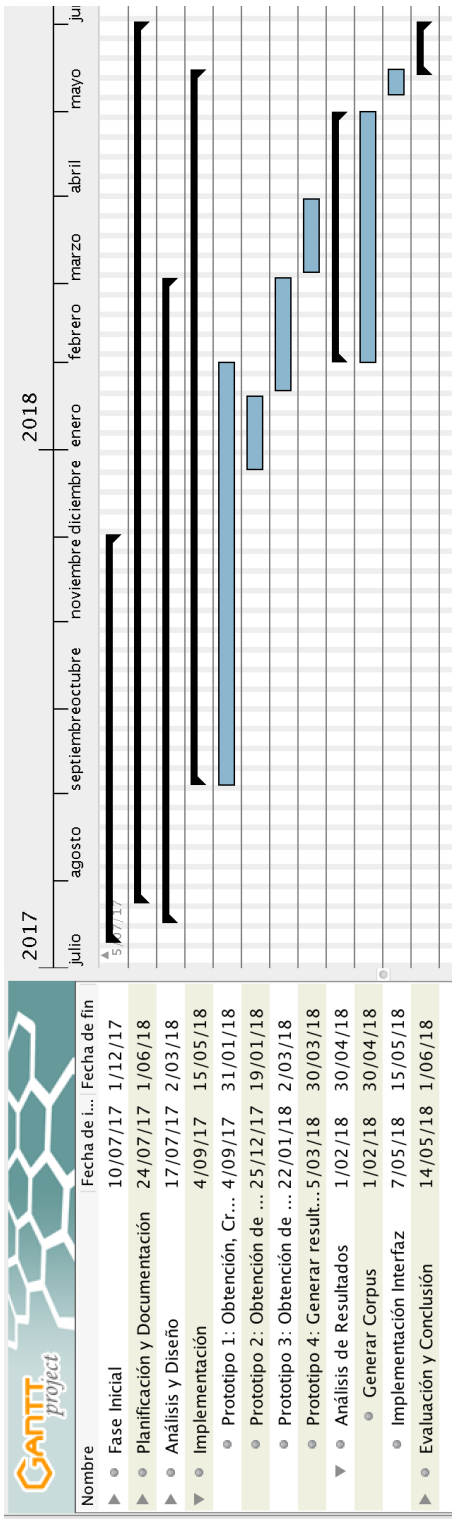


Figura 2.9: Diagrama Gantt Implementación

Diagrama de Gantt que representa gráficamente las subtareas que pertenecen a la tarea Evaluación y conclusión.(2.10).

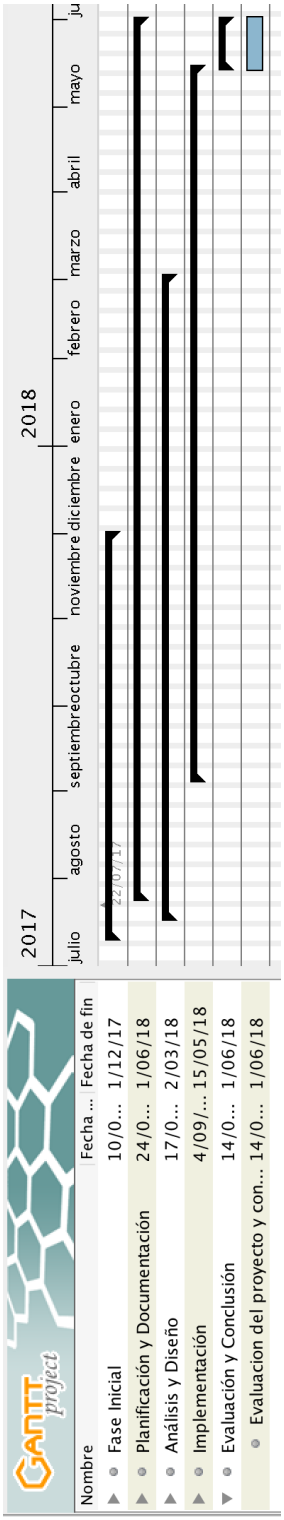


Figura 2.10: Diagrama Gantt Evaluación

2.7. Gestión de Riesgos

A lo largo del desarrollo del proyecto, pueden surgir problemas que afecten y alteren la planificación temporal y pongan en riesgo los plazos establecidos, tanto de las tareas, como de las entregas.Con el objetivo de minimizar estos riesgos se desarrolla un plan de gestión.

2.7.1. Plan General de Gestión de Riesgos

En este punto se determinan los posibles riesgos que puedan surgir a lo largo del desarrollo del proyecto. Además, para poder realizar un análisis de los mismos, se establece para cada uno de estos un plan de prevención y de contingencia.

Para facilitar el calculo de los posibles riesgos, se establecen tres niveles de probabilidad para indicar la posibilidad de aparición de estos riegos, ver tabla 2.1:

Nivel	**Probabilidad**
Baja	0-33 %
Media	33-66 %
Alta	66-100 %

Tabla 2.1: Probabilidades de Riesgos

Resulta complejo concretar el impacto que supondría un riesgo, por lo que se establece un nivel de incidencia a través de computo de días, tal y como se puede comprobar en la siguiente tabla:

Nivel de Incidencia	**Retraso**
Muy Bajo	Inferior a un día
Bajo	1 o 2 días
Medio	Entre 3 y 4 días
Alto	Entre 5 y 7 días
Muy Alto	Superior a 7 días

Tabla 2.2: Nivel de incidencia

A continuación, se numeran los riesgos posibles que puedan surgir:

- Problemas con la información
- Problemas con el diccionario CIE-10 castellano
- Problemas derivados de codificaciones
- Problemas jerarquía CIE-10
- Problemas en la obtención de todos los códigos y su información auxiliar
- Obtención de datos etiquetados con códigos CIE-10
- Problemas con el diccionario CIE-10 francés
- Problemas de versiones
- Problemas informáticos
- Errores de implementación
- Problemas personales o de salud
- Estimaciones mal realizadas
- Problemas por falta de conocimiento

Una vez identificados los posibles riegos, se realiza el plan de prevención correspondiente con el objetivo de minimizar el daño que pueda producir en el proyecto.

2.7.2. Plan de Prevención

En las tablas que se muestran a continuación se analiza cada uno de los riesgos previamente enumerados.

Problemas con la Información	
Descripción	Perdida de información relacionada con el proyecto.
Probabilidad de que suceda	Alta
Impacto	Medio
Plan de Prevención	Guardar periódicamente lo realizado hasta el momento y realizar copias de seguridad de todos los datos.
Contingencia	Cargar los datos recogidos en las copias de seguridad realizadas.

Tabla 2.3: Problemas con la información

Problemas con el diccionario CIE-10 castellano	
Descripción	Los datos que contiene el diccionario inicial son incompletos o contienen información escasa.
Probabilidad de que suceda	Alta
Impacto	Muy Alto
Plan de Prevención	Generar un documento que contenga la mayor cantidad de datos posibles, de la Clasificación de Enfermedades Internacional en castellano.
Contingencia	Generar el diccionario que contenga mas información valiosa para el objetivo del proyecto.

Tabla 2.4: Problemas con el diccionario CIE-10 castellano

Problemas derivados de la codificación	
Descripción	Problemas en la codificación de los datos extraídos del documento que contiene los datos CIE-10, es decir, caracteres mal codificados, símbolos no comprendidos por la codificación, modificación de la estructura original de los datos al extraerla para su tratamiento.
Probabilidad de que suceda	Alta
Impacto	Muy Alto
Plan de Prevención	Tener la capacidad y conocer las herramientas necesarias para realizar los cambios de codificación (utf-8, unicode, ASCII,...) necesarios, así como analizar los datos en busca de aquellos que tienen que ser codificados de una forma especial.
Contingencia	Conocer las herramientas necesarias y los datos con los que se trabaja para poder codificar correctamente los datos.

Tabla 2.5: Problemas derivados de codificaciones

Problemas jerarquía CIE-10	
Descripción	Desconocimiento de la codificación del CIE-10.
Probabilidad de que suceda	Alta
Impacto	Muy Alto
Plan de Prevención	Analizar los distintos códigos que forman parte del CIE-10 e identificar su nomenclatura así como la jerarquía de estos códigos.
Contingencia	Herramientas y documentos que faciliten la comprensión de la jerarquía de los códigos que componen el CIE-10.

Tabla 2.6: Problemas jerarquía CIE-10

Problemas en la obtención de todos los códigos y su información auxiliar	
Descripción	No obtener de manera automática todos los códigos y su información, siendo necesario tener que obtenerlos de forma manual.
Probabilidad de que suceda	Alta
Impacto	Muy Alto
Plan de Prevención	Analizar todas las posibilidades existentes para obtener la mayor cantidad de datos de forma automática.
Contingencia	Analizar e identificar aquellos datos que no es posible su obtención automática y obtenerlos manualmente.

Tabla 2.7: Problemas jerarquía CIE-10

Problemas de Obtención de datos etiquetados con códigos CIE-10	
Descripción	No disponer de datos etiquetados con códigos CIE-10, es decir, cadenas de texto con su correspondiente código o datos insuficientes.
Probabilidad de que suceda	Alta
Impacto	Muy Alto
Plan de Prevención	Generar datos de forma manual o tener que hacer uso de datos de otras lenguas, para poder evaluar de forma fiable.
Contingencia	Generar datos de forma manual y obtener datos de otros idiomas.

Tabla 2.8: Obtención de datos etiquetados con códigos CIE-10

Problemas con el diccionario CIE-10 francés	
Descripción	Necesidad de tener que modificar lo implementado para un idioma dependiendo de como se encuentren los datos en otro.
Probabilidad de que suceda	Alta
Impacto	Alto
Plan de Prevención	Analizar la estructura de los datos según el idioma.
Contingencia	Establecer una estructura común para los datos que se van a utilizar.

Tabla 2.9: Problemas con el diccionario CIE-10 francés

Problemas de versiones	
Descripción	Posibles incompatibilidades que puedan surgir entre las distintas librerías, así como, incompatibilidad de software con el sistema operativo.
Probabilidad de que suceda	Alta
Impacto	Medio
Plan de Prevención	Comprobar las distintas librerías ya utilizadas, las versiones que utilizan para que no surjan problemas con lo ya utilizado.
Contingencia	Uso de la versión mas estable

Tabla 2.10: Problemas de versiones

Problemas informáticos	
Descripción	Daños en los equipos, material que se utiliza (incendios, inundaciones, sobretensiones, uso, . . .)
Probabilidad de que suceda	Bajo
Impacto	Muy Alto
Plan de Prevención	Realizar copias de seguridad de todo lo realizado de manera periódica y almacenarlo en diferentes soportes.
Contingencia	Compra de nuevos equipos.

Tabla 2.11: Problemas informáticos

Errores de Implementación	
Descripción	Errores de funcionamiento.
Probabilidad de que suceda	Alta
Impacto	Medio
Plan de Prevención	Comprobar cual es el funcionamiento correcto de lo métodos de los que se hacen uso, así como los datos o parámetros que son necesarios para el correcto funcionamiento.
Contingencia	Modificar tanto los parámetros como los diferentes métodos aplicados.

Tabla 2.12: Errores de implementación

Problemas personales o de salud	
Descripción	Problemas familiares, enfermedad propia o de un familiar cercano, fallecimiento, etc.
Probabilidad de que suceda	Media
Impacto	Muy Alto
Plan de Prevención	Los problemas externos suceden de forma inesperada. Respecto a lo personal, tener el mayor cuidado posible, y controlando tanto los periodos de actividad, como de ocio.
Contingencia	Aprovechar tiempo libre para avanzar en las tareas retrasadas.

Tabla 2.13: Problemas familiares,enfermedad propia o de familiar, fallecimiento...

Estimaciones mal realizadas	
Descripción	El mal calculo de los tiempos de realización de cada una de las tareas que forman parte del desarrollo del proyecto.
Probabilidad de que suceda	Alta
Impacto	Muy alto
Plan de Prevención	Tratar de cumplir con las estimaciones realizadas y siendo realistas dichas estimaciones a la hora de calcularlas.
Contingencia	Modificar la planificación del proyecto.

Tabla 2.14: Estimaciones mal realizadas

Problemas por falta de conocimiento	
Descripción	La falta de conocimiento de las distintas herramientas como datos que se van a utilizar en el proyecto pueden provocar retrasos.
Probabilidad de que suceda	Alta
Impacto	Muy alto
Plan de Prevención	Realizar un buen estudio de las herramientas y tomar el tiempo necesario para su comprensión.
Contingencia	Aumentar el tiempo de aprendizaje.

Tabla 2.15: Problemas por falta de conocimiento

2.8. Planificación Económica

En este apartado se exponen los costes estimados, tanto directos, como indirectos asociados a la realización de este proyecto. Al ser un proyecto enmarcado dentro de la investigación, no consta de una amortización como tal, por tanto, estos datos serían necesarios a tener en cuenta a la hora de pedir una subvención para su financiación.

2.8.1. Software

En este apartado se especifican las distintas herramientas empleadas y su respectivo coste:

	Herramienta	Precio (€)
Sistema Operativo	Mac OS Sierra V 10.12.6	0
Documentación	GanttProject	0
	Overleaf	0
	Sublime Text 3	0
	www.cacoo.com	0
	Visual Paradigm Community Edition	0
Desarrollo software	Eclipse IDE	0
	Librerías Python	0
	DKPro Similarity	0
	Total	0

Tabla 2.16: Costes Software

2.8.2. Hardware

En este apartado se especifica el material hardware que se ha utilizado y su respectivo coste:

Herramienta	**Precio (€)**
MacBook Pro (Retina, 13 pulgadas, mediados de 2014)	1.600
Total	1.600

Tabla 2.17: Costes Hardware

2.8.3. Mano de Obra

Se ejemplifica el coste correspondiente a la mano de obra relacionada con la realización del proyecto y las horas empleadas en el.

Coste Mano de Obra	
Coste del Programador	18 €/h.
Horas totales	300 h.
Total, mano de obra	**5.400 €**

Tabla 2.18: Costes mano de obra

2.8.4. Gastos Indirectos

En este apartado se introducen los gastos que no influyen de manera directa al desarrollo del proyecto, pero que son necesarios para realizarlo.

Recursos	**Precio**	**Tiempo**
Luz	8 €/Mes	12 meses
Conexión Internet	30,49 €/Mes	12 meses
Gastos Indirectos totales	**461,95 €**	

Tabla 2.19: Gastos indirectos

2.8.5. Costes Totales

Una vez obtenidos los costes previamente desglosados, se procede a realizar el calculo total:

Recurso	**Coste (€)**
Software	0
Hardware	1.600,00
Mano de obra	5.400,00
Gastos Indirectos	461,95
Coste Total	**7.461,95 €**

Tabla 2.20: Costes Totales

Por tanto, si se solicitase un proyecto de investigación en la convocatoria correspondiente, seria necesario solicitar al menos una cantidad de 7.461,95 euros.

Capítulo 3

Análisis de Antecedentes

El presente apartado se consolida como un primer acercamiento a estudios y líneas investigaciones previas que han trabajado previamente las técnicas de similitud de textos. Este bagaje más "teórico" ayuda a comprender con mayor profundidad el uso de las técnicas que se emplean comúnmente en este ámbito.

3.1. Métodos de Clasificación

Existen diversas técnicas de clasificación de textos. Así, a continuación, se concretan algunas de las más utilizadas en los estudios e investigaciones.

3.1.1. Clustering

El clustering se enfoca en separar el conjunto de datos que se desea analizar en subgrupos cuyos datos estén relacionados entre si.

Un ejemplo interesante es expuesto por Marc Damashek en [3]. En este artículo se puede observar el uso de combinaciones de 5-grama, según un alfabeto dado y agrupa las distintas palabras del conjunto de datos en función de su significado, basándose en los n-gramas generados.

3.1.2. Redes Neuronales

Las redes neuronales han ido tomando fuerza en tareas de recuperación de información. Con lo que se consolida como una posible técnica a utilizar en nuestro proyecto. En concreto, una tarea similar A. Moschitti and A. Severyn (2015) plantearon en su trabajo. En el artículo mencionado, nos explican cómo mediante redes neuronales pretenden clasificar pares de textos. Para ello, usan las redes neuronales, mapeando las oraciones en vectores, con los cuales facilitan su comparación y utilizan medidas de similitud.

3.2. Medidas de Similitud (Similarity Measures)

Slimani (2013) [4]
Las medidas de similitud semántica, calculan la similitud entre conceptos y términos, o la distancia semántica dada una ontología. En definitiva, se utiliza para identificar conceptos que tengan características comunes.

3.3. Text Similarity

Para profundizar en Text Similarity se acude a las fuentes **M.K.Vijaymeena and K.Kavitha** y **Wael H. Gomaa and Aly A. Fahmy**[5][6]

3.3.1. String Bases similarity measure

Las medidas de similitud basadas en cadenas de texto, trabajan con secuencias de cadenas o caracteres. Y la métrica de similitud es usada para medir el valor que se indica mediante la distancia existente entre las partes coincidentes de ambos términos.

Character based similarity measure

La similitud entre dos cadenas viene dada por la cadena de caracteres de mayor tamaño que coincida.

- Longest Common Substring Algorithm (Algoritmo del substring en común mas largo): Consiste en encontrar la subsecuencia común mas larga entre dos cadenas.

- Levenshtein: calcula la similitud entre las cadenas de texto indicadas, utilizando la "Distancia Levenshtein", el conjunto mínimo de operaciones de edición necesarias para transformar la cadena A en B. A continuación, se muestra la ecuación para calcular esta similitud.

$$lev_{a,b}(i,j) = \begin{cases} max(i,j) & \text{if } min(i,j) = 0 \\ min \begin{cases} lev_{a,b}(i-1,j)+1 \\ lev_{a,b}(i,j-1)+1 \\ lev_{a,b}(i-1,j-1)+1_{(a_i \neq b_j)} \end{cases} & \text{otherwise} \end{cases} \tag{3.1}$$

La función $1_{(a_i \neq b_j)}$ toma el valor 0 si son iguales los dos caracteres que se están comparando en ese momento y 1 en el caso contrario. $lev_{a,b}(i,j)$ es la distancia entre los primeros i caracteres de a y los j primeros de b.

- Jaro Distance: La distancia Jaro se basa en el número y el orden de los caracteres en común de una cadena de texto. La fórmula correspondiente es la siguiente, donde la distancia Jaro(d_j) de dos cadenas de texto S_1 y S_2 es:

$$d_j = \begin{cases} 0 & \text{if } m = 0 \\ \frac{1}{3}\left(\frac{m}{|s^1|} + \frac{m}{|s^2|} + \frac{m-t}{m}\right) & \text{otherwise} \end{cases} \tag{3.2}$$

  Donde m es el número de caracteres que coinciden y t indica la mitad del número de transposiciones.

- Jaro Winkler: Es una extension de la distancia antes comentada, pero si las cadenas a comparar tienen un prefijo similar les atribuye mas valor.

- N-gram: Son subsecuencias de n elementos dados en una secuencia de texto. Los algoritmos que calculan la similitud de los n-gramas, generan tantas combinaciones de estos como son posibles y realizan comparaciones entre los n-gramas de los términos indicados. Una vez realizado esto, calculan el resultado en base a las coincidencias que hayan habido y el máximo de n-gramas.

Term-based similarity measure

- Cosine Similarity: Es la medida de similitud que existe entre dos vectores, estos vectores representan los términos que se van a comparar, y valor de la similitud se calcula mediante el valor del angulo del coseno que forman los dos vectores.

$$S_{cos}(d_1, d_2) = \frac{d_1.d_2}{\sqrt{d_1.d_1}.\sqrt{d_2.d_2}} \tag{3.3}$$

- Dice Coeficient: Se define como el doble del número de términos comunes en las cadenas a comparar, dividido entre el número total de términos de las dos cadenas.

$$S_{Dic}(d_1, d_2) = \frac{2.(d_1.d_2)}{d_1.d_1 + d_2 d_2} \tag{3.4}$$

- Jaccard Coefficient: Se calcula realizando la división de todos los términos que comparten las dos cadenas, entre los términos únicos presentes en ellas.

$$S_{Jaccard}(x, y) = \frac{tok(x) \cap tok(y)}{|tok(x)| + |tok(y)| - |tok(x) \cap tok(y)|} = \frac{|tok(x) \cap tok(y)|}{|tok(x) \cup tok(y)|} \tag{3.5}$$

- Simple Matching Coefficient: Se calcula mediante el numero de términos que comparten ambas cadenas.

- Overlap Coefficient: Similar al coeficiente Dice, aunque este, si una cadena es subcadena de otra, considera que la una coincide con la otra.

$$overlap(X, Y) = \frac{|X \cap Y|}{min(|X|, |Y|)} \tag{3.6}$$

Corpus-Based Similarity

Es una medida de similitud semántica, que determina la similitud que existe entre dos términos en base a la información obtenida de un corpus.

3.4. DKPro Similarity[2]

Es una herramienta de código abierto que tiene como objetivo la creación de interfaces estandarizadas para el uso de medidas de similitud en la búsqueda de similitud de textos.
También proporciona componentes software para NLP, procesamiento de lenguaje natural. Incluye una gran variedad de de medidas relacionadas con la búsqueda de similitud entre n-gramas o subsecuencias de cadenas de texto.

3.5. Conclusión

Finalmente, tras realizar el análisis de los distintos métodos más utilizados y teniendo en cuenta los datos de los que se disponen para este proyecto, se opta por el uso de la herramienta "DKPro Similarity" que facilita la implementación de distintas medidas de similitud.

En caso de que el tiempo lo permita y si es posible lograr un conjunto de datos mayor con el cual poder implementar otros métodos, se escogería profundizar en el estudio de la redes neuronales.

Capítulo 4

Captura de Requisitos

En este aparatado se exponen las diferentes funcionalidades de las que consta el proyecto; las que permiten abarcar los objetivos indicados previamente, y, por ende, que el proyecto se lleve a cabo.

En este proyecto se tiene en cuenta la aparición de un requisito no funcional. Este requisito viene dado por hacer uso de la herramienta "DKPro Similarity", ya que las librerías necesarias para su implementan se encuentran en desarrolladas en Java, por ello el apartado correspondiente a su implementación se desarrolla en dicho lenguaje.

En este proyecto se descarta realizar diagramas de entidad relación, debido a que no se hace uso de una base de datos que requiera su implementación. Esto se debe a que los datos que se van a utilizar provienen de documentos, así como, los resultados que se generan se almacenan en archivos de texto.

4.1. Jerarquía de Actores

En este proyecto únicamente existe un actor: el usuario que de uso al proyecto. Esta persona es la que ejecuta la la aplicación en el momento de su uso e interactúe con los datos.

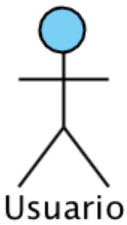


Figura 4.1: Jerarquía de actores.

4.2. Diagramas de Caso de Uso

En el siguiente diagrama de casos de uso se pueden distinguir las funcionalidades.

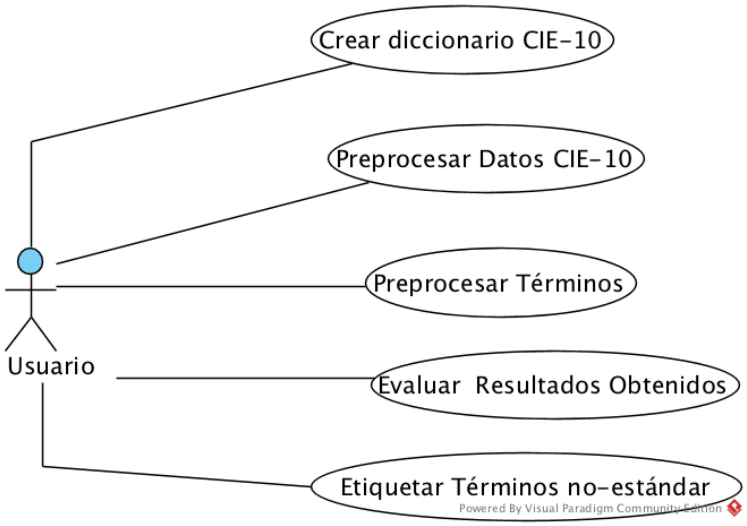


Figura 4.2: Diagrama de casos de uso.

4.2.1. Extraer y exportar los datos del archivo CIE-10

Este primer módulo se encarga de extraer del documento en formato *pdf* –el cual contiene los datos del CIE-10–, la información de la que consta el "diccionario". Una vez extraídos todos estos datos se procede a realizar el preprocesamiento de los mismos, para poder hacer uso de ellos.

Este módulo, también, se encarga de preprocesar los términos no-estándar, según el idioma que se vaya a utilizar, castellano o francés. De esta manera, al establecer el mismo formato entre los datos del "diccionario" y los términos no-estándar (de los que se quiere obtener su correspondiente código), permiten realizar una comparación entre los datos mucho mas homogénea.

4.2.2. Etiquetar Términos no-estándar

A través de los módulos *lexical* y *LSA* que facilita la herramienta "DKPro SimilariTy", este módulo se encarga de aplicar distintos algoritmos de similitud de textos que han sido utilizados para comparar los datos del "diccionario" y los términos no-estándar.

4.2.3. Evaluar los resultados obtenidos

Con esta opción el usuario del sistema podrá etiquetar los términos no-estándar introducidos y además, obtener la evaluación de los mismos. siempre y cuando se dispongan de los datos necesarios para su evaluación. Esto resulta útil a la hora de analizar el funcionamiento del software.

Capítulo 5

Análisis y Diseño

En este capítulo se exponen los pasos previos realizados a la implementan de los prototipos. Con lo cual, se identifican las funcionalidades que debe cumplir cada uno de los prototipos y su diseño.

5.1. Análisis

El proyecto, como ya se ha explicado previamente, se divide en cuatro prototipos, los cuales constan de las siguientes funcionalidades:

- Prototipo1: Obtención, creación y preprocesamiento de los datos:
  - Extracción de los datos del documento *pdf* y obtención de los datos para su almacenamiento en un documento en formato *csv*.
    - Extraer datos del documento original y codificar caracteres mal extraídos o no identificados correctamente por el formato de codificación.
    - Obtención de códigos principales y términos estándar correspondientes a la clasificación CIE.
    - Obtención de los códigos de segundo nivel y generación de los códigos de tercer nivel correspondientes a cada uno de los códigos principales.
    - Obtener los datos auxiliares correspondientes a los códigos.
  - Pre-proceso de los datos.

- Prototipo2: Obtención de similitudes Mediante "DKPro Similarity" *lexical*:
  - Aplicación de los algoritmos que nos propone "DKPro Similarity" en su módulo *lexical*.
  - Obtención de los datos necesarios de aquellos algoritmos que lo requieren para su ejecución.

- Prototipo3: Obtención de similitudes Mediante "DKPro Similarity" *LSA*:
  - Aplicación del algoritmo que nos propone "DKPro Similarity" en su módulo *LSA*.
  - Obtención de los datos necesarios para su ejecución.

- Prototipo4: Exportar y evaluar resultados:
  - Generación de los resultados y su exportación en archivos de texto.
  - Realizar las comprobaciones necesaria de los resultados obtenidos para obtener los datos con los que realizar la evaluación.

5.2. Diseño

Para el diseño de este software se ha utilizado un proceso iterativo, de manera que los diseños del prototipo 2 al 4 se complementan entre si, es decir, los apartados desarrollados en el prototipo previo, son base para continuar con el siguiente, añadiendo funcionalidades nuevas al anterior.

Por otro lado, el prototipo 1 se encarga de extraer los datos del documento *pdf* con la información del CIE-10, y obtener sus datos con la mayor exactitud posible. Este prototipo a lo largo de su diseño y posterior implementación ha tenido diversos cambios. Estos cambios se deben a que los datos que se obtienen, se han de ir analizando paso a paso, identificando cuáles se pueden lograr de manera automática y cuáles deben ser generados o modificados manualmente. Además, de añadir nuevos datos auxiliares, que en un principio no se decidieron obtener y que al final si se introdujeron.

5.2.1. Diagrama de clases

A continuación, se muestran dos diagramas de clase. El primero, se corresponde con la creación del "diccionario". Dicho diagrama hace referencia a las distintas clases generadas para realizar el primer prototipo -el cual se realiza en Python-, ya que la idea inicial era realizar el proyecto en su totalidad en este lenguaje de programación. Sin embargo, los demás prototipos se desarrollaron en el lenguaje de programación Java.

Figura 5.1: Diagrama de clases creación y preproceso del diccionario.

Figura 5.2: Diagrama de clases implementación DKPro Similarity.

Capítulo 6

Desarrollo de Prototipos

6.1. Prototipo 1: Obtención, Creación y prepocesamiento de los Datos

Para la finalidad de esta investigación se precisan dos significativos recursos. Por un lado, el ”diccionario”, que en este caso sería la lista de los códigos CIE-10, con sus términos estándar y su información auxiliar, y por el otro, los diagnósticos descritos por los médicos,es decir, los términos no-estándar escritos en lenguaje natural espontaneo y etiquetados con su respectivo código CIE-10.

Para la obtención de los recursos mencionados, desde el equipo de investigación se facilitó un diccionario con los distintos códigos y términos del CIE-10. Este diccionario contiene los códigos principales con sus términos. Sin embargo, al acceder a la Web del Ministerio de Sanidad, Servicios Sociales e Igualdad se observa que en la mayoría de códigos se muestra una información auxiliar, tal y como se puede ver en la siguiente imagen:

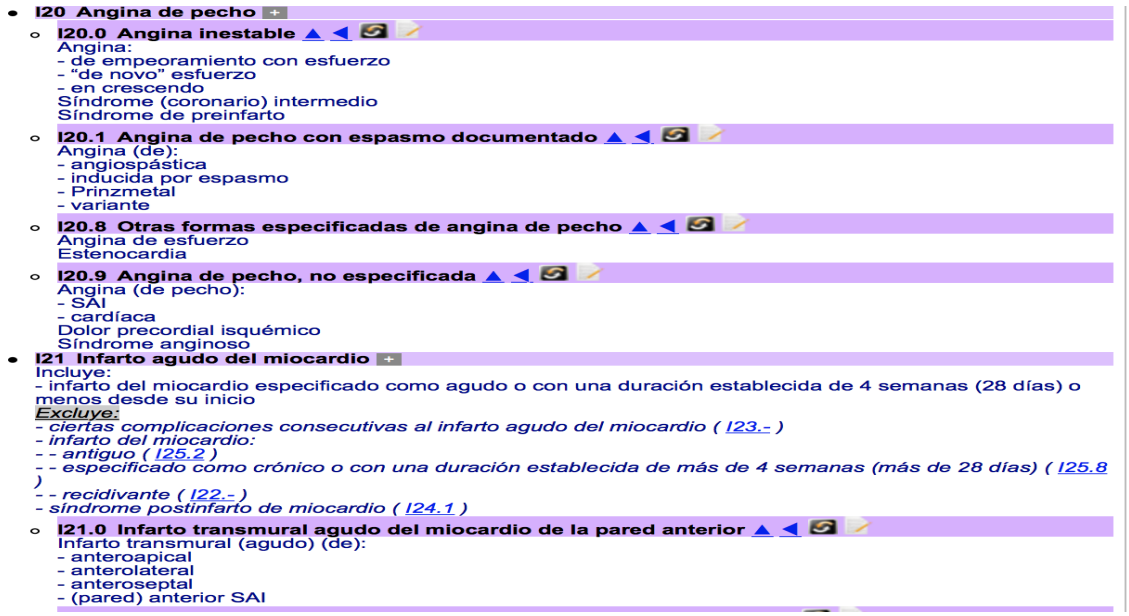


Figura 6.1: Ejemplo datos CIE-10 de la web del MSSSI

Concretamente, respecto a la imagen cabe detallar que se obtiene un código principal “I20 Angina de pecho”, que se subdivide en otros códigos como (I20.0, I20.1, I20.8 y I20.9). Cada uno de estos códigos, además de su término estándar, contienen descripciones propias que aportan información auxiliar, tal y como se puede comprobar a continuación:

- **Código y término estándar**: “I20.0 Angina inestable”
  - **Información auxiliar:**
    - Angina:
      - de empeoramiento con esfuerzo
      - “de novo” esfuerzo
      - Angina en crescendo
    - Síndrome (coronario) intermedio
    - Síndrome de pre-infarto
- **Código y término estándar**: “I20.9 Angina de pecho, no especificada”
  - **Información auxiliar:**
    - Angina (de pecho):
      - SAI
      - Cardíaca
    - Dolor pectoral isquémico
    - Síndrome anginoso

Además de la información auxiliar que se muestra en el ejemplo, o términos correspondientes a los códigos, se puede apreciar que alguna información adicional se indica mediante dos palabras clave: incluye y/o excluye. La información indicada a través de estas dos palabras también se obtiene y mas adelante en este documento se profundiza sobre ellas.

Sin embargo, en el diccionario aportado únicamente se encuentran los datos correspondientes al código, es decir, el código y el término estándar, tal y como se puede ver a continuación:

I200	ANGINA INESTABLE
I240	TROMBOSIS CORONARIA QUE NO RESULTA EN INFARTO DEL MIOCARDIO
I241	SINDROME DE DRESSLER
I248	OTRAS FORMAS DE ENFERMEDAD ISQUEMICA AGUDA DEL CORAZON
I249	ENFERMEDAD ISQUEMICA AGUDA DEL CORAZON, NO ESPECIFICADA
I252	INFARTO ANTIGUO DEL MIOCARDIO
I200	ANGINA INESTABLE
I201	ANGINA DE PECHO CON ESPASMO DOCUMENTADO
I208	OTRAS FORMAS ESPECIFICADAS DE ANGINA DE PECHO
I209	ANGINA DE PECHO, NO ESPECIFICADA
I251	ENFERMEDAD ATEROSCLEROTICA DEL CORAZON
I258	OTRAS FORMAS DE ENFERMEDAD ISQUEMICA CRONICA DEL CORAZON
I253	ANEURISMA CARDIACO

Figura 6.2: Ejemplo contenido datos CIE-10 inicial

Al analizar los códigos arriba mencionados, se distinguen los siguientes datos:

- “I201 Angina inestable”
- “I209 Angina de pecho, no especificada”

A la luz de estos “escasos” resultados se decide explorar y conseguir un diccionario CIE-10 más completo, y que se asemeje a los datos encontrados en la Web del Ministerio en concreto.

Con este fin, y con la intención de lograr los datos de forma procesada y almacenarlos –en formato *csv*. – se realiza una búsqueda exhaustiva a través de Internet. No obstante, no se obtienen resultados concluyentes, ya que los archivos correspondientes de CIE-10 en formato *csv* contienen datos básicos –los cuales ya se disponían–, y los que aportaban una información más amplia, carecían del grosor de los datos necesarios.

Finalmente, y aunque no se logra conseguir un archivo en formato *csv* se obtiene un documento *pdf* que contiene los datos requeridos, aunque en algunos códigos se añadía alguna otra información difusa y poco necesaria para el fin propuesto.

Este *pdf*, concretamente, se encuentra disponible en la siguiente dirección (clic aquí). Así, al ser un archivo con la información más próxima a la necesitada, se opta por extraer los datos y prescindir de aquellos que para el diccionario no eran relevantes. Tras esta decisión, se considera obtener la mayor información posible, y así aumentar los datos, ya que, a mayor número de datos, mayor fiabilidad de resultados se obtiene.

Para detallar los pasos dados en el logro del diccionario CIE-10, y, a su vez, exponer las decisiones tomadas en la extracción de los datos, se realiza el siguiente resumen:

6.1.1. Comprobación, extracción y obtención de los datos

Antes de explicar los pasos dados, es importante aclarar qué información se va a manejar. De esta manera, y para facilitar el seguimiento y comprensión de los códigos, así como el formato que compone el CIE-10, se limita una leyenda explicativa. Esta, en concreto, sirve para 1) diferenciar los tipos de códigos que existen en el archivo, y, 2) asignar una denominación para denominarlos de aquí en adelante.

He aquí la leyenda:

- Código Principal, con el formato “[A-Z][0-9][0-9]” Ej.: M00
- Códigos de segundo nivel, con el formato “[A-Z][0-9][0-9].[0-9]” Ej.: M00.0
- Códigos de tercer nivel, con el formato “[A-Z][0-9][0-9].[0-9][0-9]” Ej.: M00.00

[A-Z] Indican que están compuestos por la letra de la “A” a la “Z”, excepto la "Ñ’, y [0-9] que los dígitos permitidos están entre el “0” y el “9”.

6.1.2. Primer Paso: Comprobación de contenidos

Antes de comenzar con la exportación de los datos del documento *pdf*, es necesario realizar una comprobación previa de los datos. Es decir, demostrar que los datos que aparecen en el archivo concuerdan con los datos que podemos observar en la Página Web del MSSSI.

Para ello, primero, se lleva a cabo una comparación aleatoria de un número “n”, de códigos que aparecen en un lugar y en el otro, para, así, decidir y mostrar que el contenido en uno, corresponde con el contenido del otro. Para ilustrar estas comprobaciones, a continuación, se añade una tabla que contiene los "n"valores, en este caso, 100 códigos seleccionados indicando si concuerdan o no.

Nº Comprobación	Código Principal	Resultados
1	I20	Concuerdan
2	K61	Concuerdan
3	K59	Concuerdan
4	L66	Concuerdan
5	L67	Concuerdan
6	E24	Concuerdan
7	A15	Concuerdan
8	G04	Concuerdan
9	I36	Concuerdan
10	I73	Concuerdan
11	I83	Concuerdan
12	I84	Concuerdan
13	J10	Concuerdan
14	J11	Concuerdan
15	L03	Concuerdan
16	N36	Concuerdan
17	N64	Concuerdan
18	O08	Concuerdan
19	P15	Concuerdan
20	Q16	Concuerdan
21	Q69	Concuerdan
22	Q89	Concuerdan
23	R49	Concuerdan
24	S02	Concuerdan
25	K81	Concuerdan
26	L12	Concuerdan

27	M05	Concuerdan
28	M01	Concuerdan
29	M54	Concuerdan
30	N90	Concuerdan
31	P39	Concuerdan
32	Q06	Concuerdan
33	R04	Concuerdan
34	R68	Concuerdan
35	S13	Concuerdan
36	S42	Concuerdan
37	S52	Concuerdan
38	TO4	Concuerdan
39	T63	Concuerdan
40	V19	Concuerdan
41	Y42	Concuerdan
42	Y91	Concuerdan
43	Z13	Concuerdan
44	Z62	Concuerdan
45	Z94	Concuerdan
46	F80	Concuerdan
47	G21	Concuerdan
48	H35	Concuerdan
49	J84	Concuerdan
50	K80	Concuerdan
51	R96	Concuerdan
52	S03	Concuerdan
53	Q71	Concuerdan
54	H52	Concuerdan
55	H93	Concuerdan
56	I49	Concuerdan
57	J00	Concuerdan
58	J37	Concuerdan
59	Y35	Concuerdan
60	V97	Concuerdan

61	T78	Concuerdan
62	Q55	Concuerdan
63	N11	Concuerdan
64	E87	Concuerdan
65	D64	Concuerdan
66	D04	Concuerdan
67	C62	Concuerdan
68	D10	Concuerdan
69	B65	Concuerdan
70	F20	Concuerdan
71	F95	Concuerdan
72	E72	Concuerdan
73	G31	Concuerdan
74	K71	Concuerdan
75	T91	Concuerdan
76	E10	Concuerdan
77	E00	Concuerdan
78	D59	Concuerdan
79	A00	Concuerdan
80	S26	Concuerdan
81	L70	Concuerdan
82	F45	Concuerdan
83	H16	Concuerdan
84	098	Concuerdan
85	N73	Concuerdan
86	L90	Concuerdan
87	J31	Concuerdan
88	G46	Concuerdan
89	H05	Concuerdan
90	K42	Concuerdan
91	K14	Concuerdan
92	A09	Concuerdan
93	B08	Concuerdan
94	B76	Concuerdan

95	T25	Concuerdan
96	D60	Concuerdan
97	I77	Concuerdan
98	B65	Concuerdan
99	A31	Concuerdan
100	R94	Concuerdan

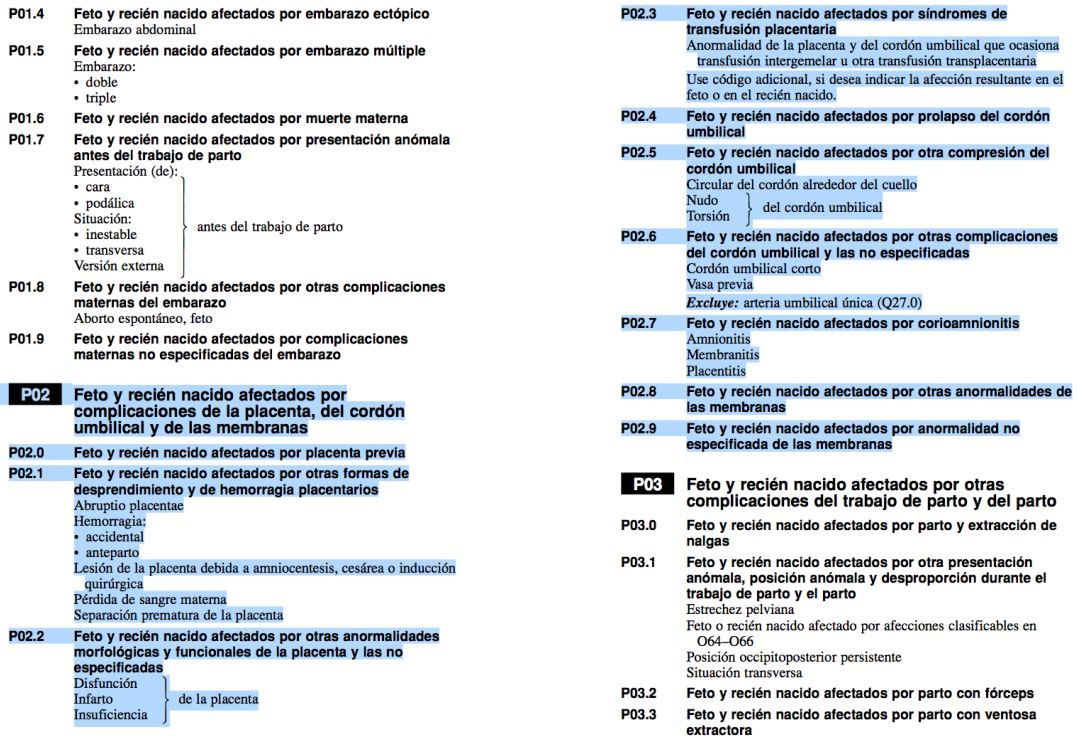
Tabla 6.1: Comprobación de códigos aleatoria

Brevemente evidenciar que, como se puede observar en la tabla, los códigos que aparecen son los principales. Aun así, en determinadas ocasiones, también existen códigos de segundo nivel, como de tercer nivel, y los que también han sido comprobados.

Junto con la tabla adjuntada, Tabla 6.1, también se procede a una explicación gráfica de como se realizan estas comprobaciones, para ello se selecciona un código principal del CIE-10 de manera aleatoria, por ejemplo:

"P02 Feto y recién nacido afectados por complicaciones de la placenta, del cordón umbilical y de las membranas"

En primer lugar, se realiza la búsqueda de dicho código en el documento:



P01.4 **Feto y recién nacido afectados por embarazo ectópico**
Embarazo abdominal

P01.5 **Feto y recién nacido afectados por embarazo múltiple**
Embarazo:
- doble
- triple

P01.6 **Feto y recién nacido afectados por muerte materna**

P01.7 **Feto y recién nacido afectados por presentación anómala antes del trabajo de parto**
Presentación (de):
- cara
- podálica

Situación:
- inestable
- transversa

Versión externa

} antes del trabajo de parto

P01.8 **Feto y recién nacido afectados por otras complicaciones maternas del embarazo**
Aborto espontáneo, feto

P01.9 **Feto y recién nacido afectados por complicaciones maternas no especificadas del embarazo**

P02 **Feto y recién nacido afectados por complicaciones de la placenta, del cordón umbilical y de las membranas**

P02.0 **Feto y recién nacido afectados por placenta previa**

P02.1 **Feto y recién nacido afectados por otras formas de desprendimiento y de hemorragia placentarios**
Abruptio placentae
Hemorragia:
- accidental
- anteparto

Lesión de la placenta debida a amniocentesis, cesárea o inducción quirúrgica
Pérdida de sangre materna
Separación prematura de la placenta

P02.2 **Feto y recién nacido afectados por otras anormalidades morfológicas y funcionales de la placenta y las no especificadas**
Disfunción
Infarto
Insuficiencia

} de la placenta

P02.3 **Feto y recién nacido afectados por síndromes de transfusión placentaria**
Anormalidad de la placenta y del cordón umbilical que ocasiona transfusión intergemelar u otra transfusión transplacentaria
Use código adicional, si desea indicar la afección resultante en el feto o en el recién nacido.

P02.4 **Feto y recién nacido afectados por prolapso del cordón umbilical**

P02.5 **Feto y recién nacido afectados por otra compresión del cordón umbilical**
Circular del cordón alrededor del cuello
Nudo
Torsión

} del cordón umbilical

P02.6 **Feto y recién nacido afectados por otras complicaciones del cordón umbilical y las no especificadas**
Cordón umbilical corto
Vasa previa
Excluye: arteria umbilical única (Q27.0)

P02.7 **Feto y recién nacido afectados por corioamnionitis**
Amnionitis
Membranitis
Placentitis

P02.8 **Feto y recién nacido afectados por otras anormalidades de las membranas**

P02.9 **Feto y recién nacido afectados por anormalidad no especificada de las membranas**

P03 **Feto y recién nacido afectados por otras complicaciones del trabajo de parto y del parto**

P03.0 **Feto y recién nacido afectados por parto y extracción de nalgas**

P03.1 **Feto y recién nacido afectados por otra presentación anómala, posición anómala y desproporción durante el trabajo de parto y el parto**
Estrechez pelviana
Feto o recién nacido afectado por afecciones clasificables en O64–O66
Posición occipitoposterior persistente
Situación transversa

P03.2 **Feto y recién nacido afectados por parto con fórceps**

P03.3 **Feto y recién nacido afectados por parto con ventosa extractora**

Figura 6.3: Datos del código P02 resaltados en el pdf

Y posteriormente en la web, para ello, se accede a la siguiente dirección (pulsar aquí) y se muestra la siguiente pantalla:

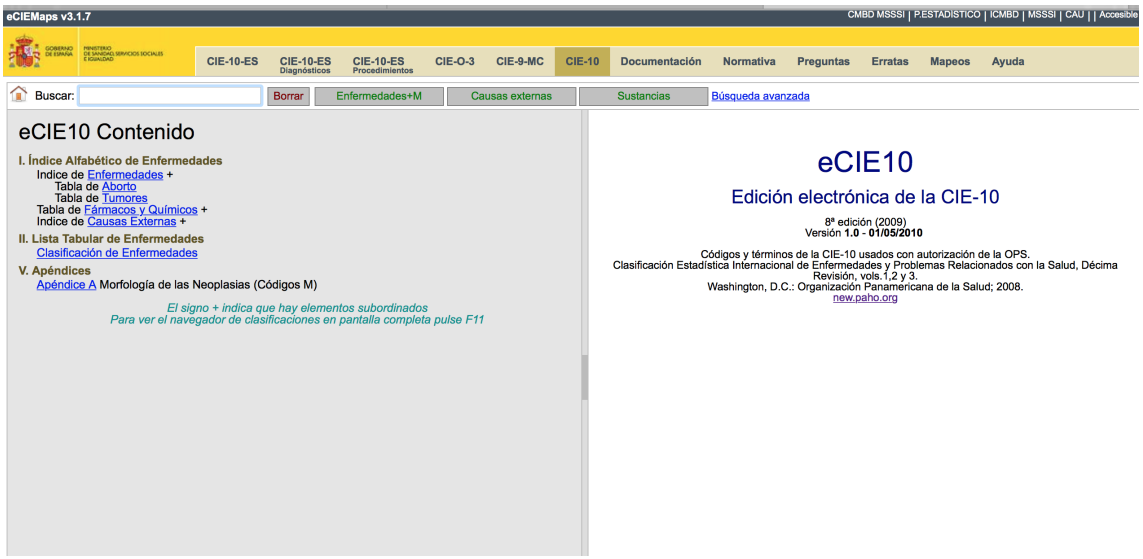


Figura 6.4: Página web del MSSSI

Como se puede observar, en la parte superior de la pantalla aparece una menú con diferentes opciones, de la cual se selecciona la opción CIE-10 y en posteriormente en el recuadro de texto que se sitúa al lado de “Buscar:”, se introduce el código principal y se realiza la búsqueda, obteniendo los siguientes resultados:

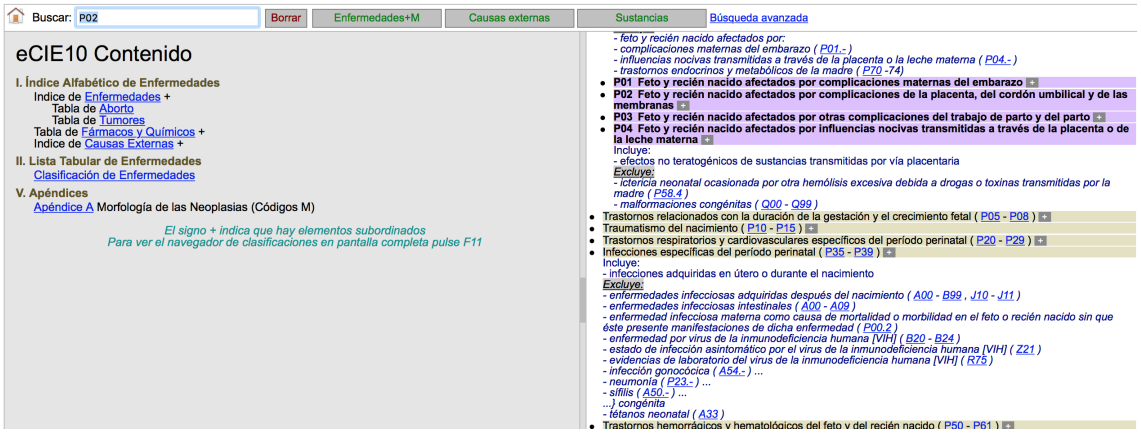


Figura 6.5: Búsqueda P02 web MSSSI

En un análisis exhaustivo se aprecia que al lado de la descripción del código “P02”, se haya un símbolo “+”. Este al pulsarlo se expande, y muestra más información de dicho código, como se puede observar en la posterior imagen:

Figura 6.6: Resultados código P02 web MSSSI

Una vez se obtienen estas dos informaciones, se realiza la comparaciones entre ambas. Remarcar que todas y cada una de las comprobaciones han sido realizadas siguiendo las instrucciones indicadas.

6.1.3. Segundo Paso: Extracción Datos Volumen1.pdf

Para efectuar la extracción de los datos contenidos en el documento en formato *pdf*, a un formato en que los datos puedan ser manejados y tratados, se plantea el uso de dos módulos de Python, el módulo "PyPDF2" y el módulo "pdfmainer". Sin embargo, en la extracción del contenido del *pdf* a texto plano, cada uno de los módulos ocasiona desventajas y ventajas que se pasan a detallar:

- Pdfmainer:
  Recoge los datos (palabras) codificados de manera correcta, correctamente codificados respecto a los datos originales. No obstante, el módulo alteraba el orden de los datos, lo que imposibilita la obtención de cada uno de los códigos y sus correspondientes descripciones, siendo mas sencillo copiar de manera manual uno a uno cada uno de los datos.
- PyPDF2:
  Al obtner los datos en texto plano, no decodifica de manera correcta todos los caracteres o símbolos que aparecen en el documento, y en ciertos puntos inserta datos que no son útiles, aun así, al obtener los datos en el orden correcto, habilita la posibilidad de identificar los distintos datos que componen el CIE-10 facilitando su obtención.

A continuación se expone un ejemplo visual que describe lo arriba expuesto:

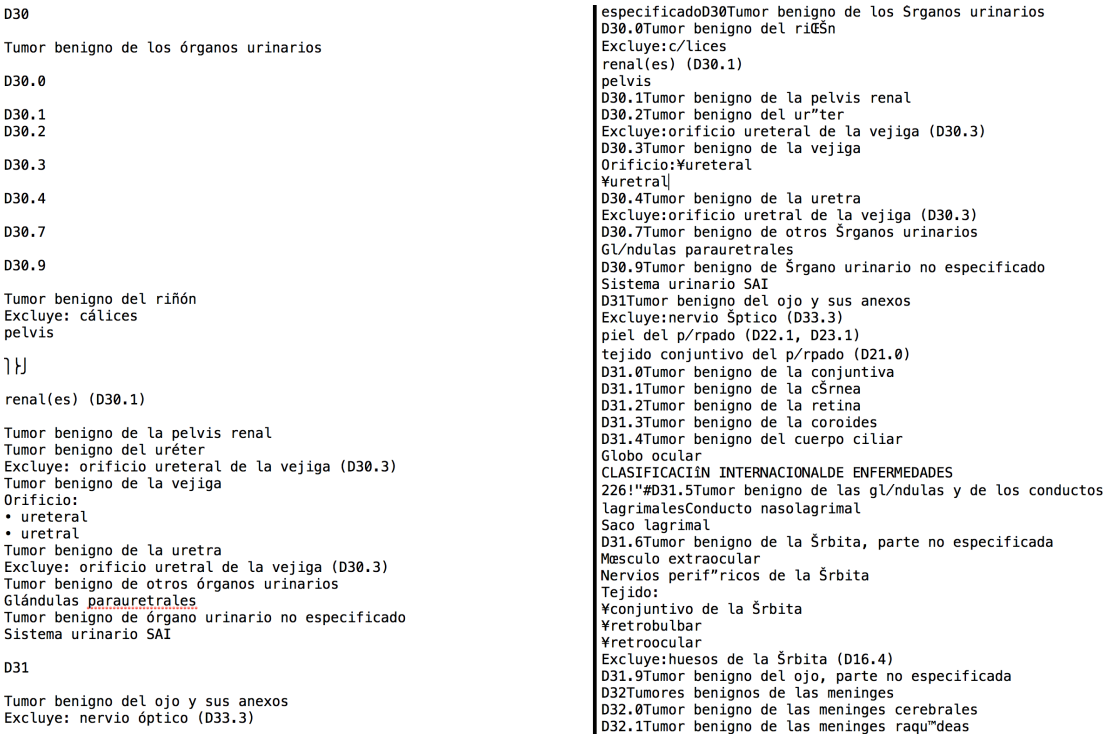


Figura 6.7: Resultados pdfminer(izq.) vs PypDF2 (dch.)

Tras el análisis de la recogida de datos en ambos módulos, se decide utilizar el módulo "PyPDF2", ya que resulta más sencillo identificar a qué código principal corresponde una determinada información.

6.1.4. Tercer Paso, caracteres especiales

Una vez obtenidos los datos del archivo *pdf* en formato de texto, para un manejo más sencillo, se unifican todos los datos en una misma línea, y de esta manera, mediante un filtro se modifican los caracteres especiales o símbolos que no han sido correctamente codificados en la extracción de los datos de un formato a otro. Este paso se realiza utilizando el módulo "re" de Python, ya que facilita el uso de expresiones regulares y búsquedas en cadenas de caracteres, permitiendo cambiar los caracteres especiales por los caracteres correspondientes de manera sencilla.

```
Enfermedades infecciosas intestinales
(A00ÐA09)A00CŠleraA00.0CŠlera debido a
Vibrio cholerae
O1, biotipo cholerae
CŠlera cl/sico
A00.1CŠlera debido a
Vibrio cholerae
O1, biotipo El Tor
CŠlera El Tor
```

Figura 6.8: Ejemplo de datos sin modificar

Como se aprecia en la captura, las palabras "Cólera", "clásico" y "paralítica" contienen un carácter erróneamente codificado. Para solventar este punto, se realiza la sustitución de los caracteres especiales, por los correspondientes, los cambios realizados son los que se muestran en la siguiente tabla:

Carácter Extraído	Carácter Original	Carácter Final
/	á	a
Ç	Á	A
"	é	e
...	É	E
™	í	i
ê	Í	I
Š	ó	o
î	Ó	O
œ	ú	u
ò	Ú	U
Œ	ñ	ni
Ð	-	-

Tabla 6.2: Caracteres especiales codificación.

6.1.5. Cuarto Paso,obtención de códigos principales

En este paso se dispone de la información codificada como se requiere, esto es, con los caracteres especiales modificados. Por ello, se procede a recorrer los datos y a dividirlos según el código principal, es decir, con el objetivo de conseguir diferenciar los códigos principales en los que se divide el CIE10.

Se parte de un archivo en formato de texto, con los datos en una misma línea y siendo estos una mezcla de letras, palabras y códigos, como se puede ver en la imagen.

```
Enfermedades infecciosas intestinales(A00-A09)A00ColeraA00.0Colera debido a
Vibrio cholerae 01, biotipo choleraeColera clasicoA00.1Colera debido a Vibrio
cholerae 01, biotipo El TorColera El TorA00.9Colera, no
especificadoA01Fiebres tifoidea y paratifoideaA01.0Fiebre tifoideaInfeccion
debida a Salmonella typhiA01.1Fiebre paratifoidea AA01.2Fiebre paratifoidea
BA01.3Fiebre paratifoidea CA01.4Fiebre paratifoidea, no especificadaInfeccion
debida a Salmonella paratyphiSAIAA02Otras infecciones debidas a
SalmonellaIncluye:infeccion o intoxicacion alimentaria debida a
cualquierespecie de Salmonella excepto S. typhi yS. paratyphiA02.0Enteritis
debida a SalmonellaSalmonelosisA02.1Septicemia debida a
SalmonellaA02.2Infecciones localizadas debidas a SalmonellaArtritis€
(M01.3*)Enfermedad renal tubulointersticial€(N16.0*)Meningitis€
(G01*)Neumonia€(J17.0*)Osteomielitis€(M90.2*)A02.8Otras infecciones
especificadas como debidas aSalmonellaA02.9Infeccion debida a Salmonella, no
especificadaCLASIFICACION INTERNACIONALDE ENFERMEDADES106debida
aSalmonella!""#""$A03ShigelosisA03.0Shigelosis debida a Shigella
dysenteriaeShigelosis grupo A[disenteria de Shiga-Kruse]A03.1Shigelosis
debida a Shigella flexneriShigelosis grupo BA03.2Shigelosis debida a Shigella
boydiiShigelosis grupo CA03.3Shigelosis debida a Shigella sonneiShigelosis
grupo DA03.8Otras shigelosisA03.9Shigelosis de tipo no especificadoDisenteria
bacilar SAIA04Otras infecciones intestinales bacterianasExcluye:enteritis
```

Figura 6.9: Muestra de datos en formato de texto

A pesar de que los datos se encuentran unidos, siguen un patrón. Aparece el código principal, seguido de su término estándar e información auxiliar en el caso de disponer de ella. Además, si hay códigos de segundo nivel también se encuentran entre los datos, así como, su información. He aquí un ejemplo ilustrativo:

- Leyenda:
  - Códigos Principales
  - Información auxiliar
  - Códigos Segundo Nivel
  - Información innecesaria
  - Códigos Referenciados

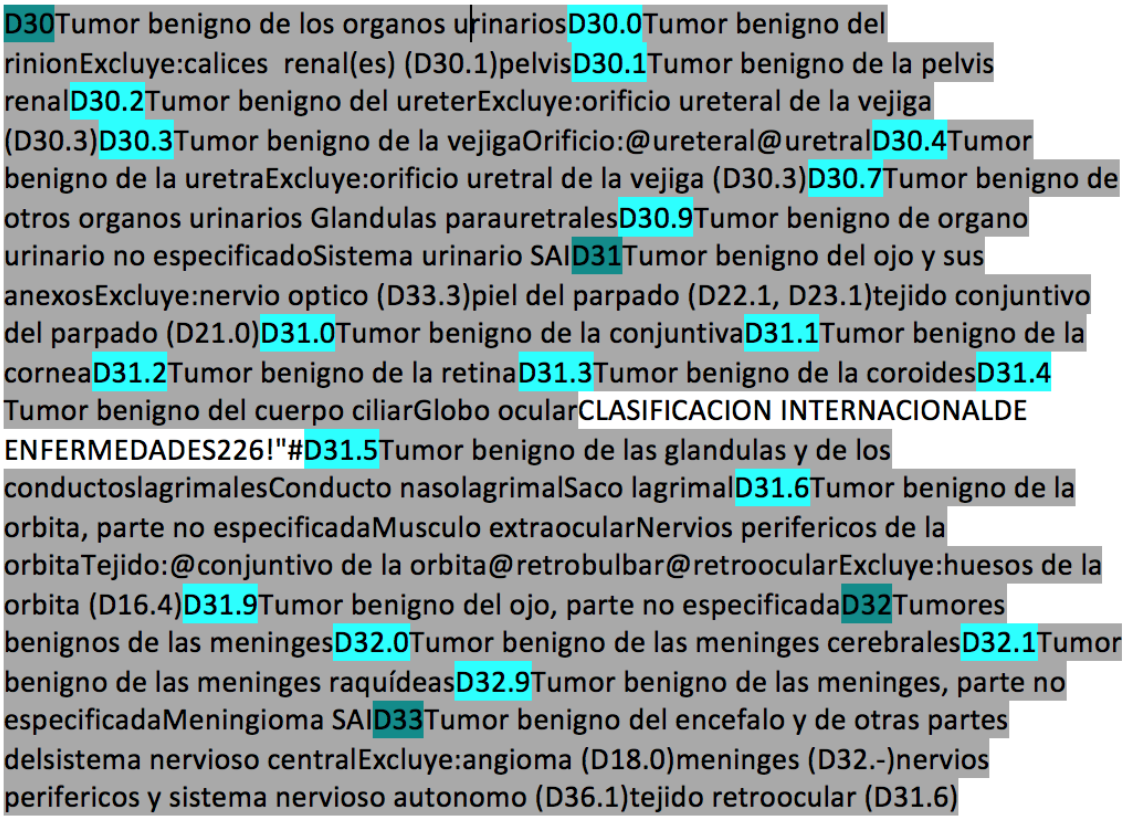


Figura 6.10: Tramo de códigos(D30-D31-D32)

En la muestra destacada se examina un tramo de datos referente a los códigos principales "D30, D31, D32 y D33". Respecto al modelo que se ha adjuntado, es importante señalar que la parte correspondiente a la información innecesaria es eliminada a lo largo del proceso.

No obstante, como se puede ver en el ejemplo, entre un código principal y el siguiente se obtiene información distinta. Esto es, desde el código “D30” hasta el código “D31” del archivo se añade información que corresponde al primer código (D30), incluyendo los que serían los códigos de segundo nivel, el término estándar y la información auxiliar. Esta información se asume como información significativa, y que, por tanto, se necesita obtener.

Tomando como el estado inicial de los datos la figura 6.10 el siguiente fragmento de información muestra el resultado tras detectar los códigos principales:

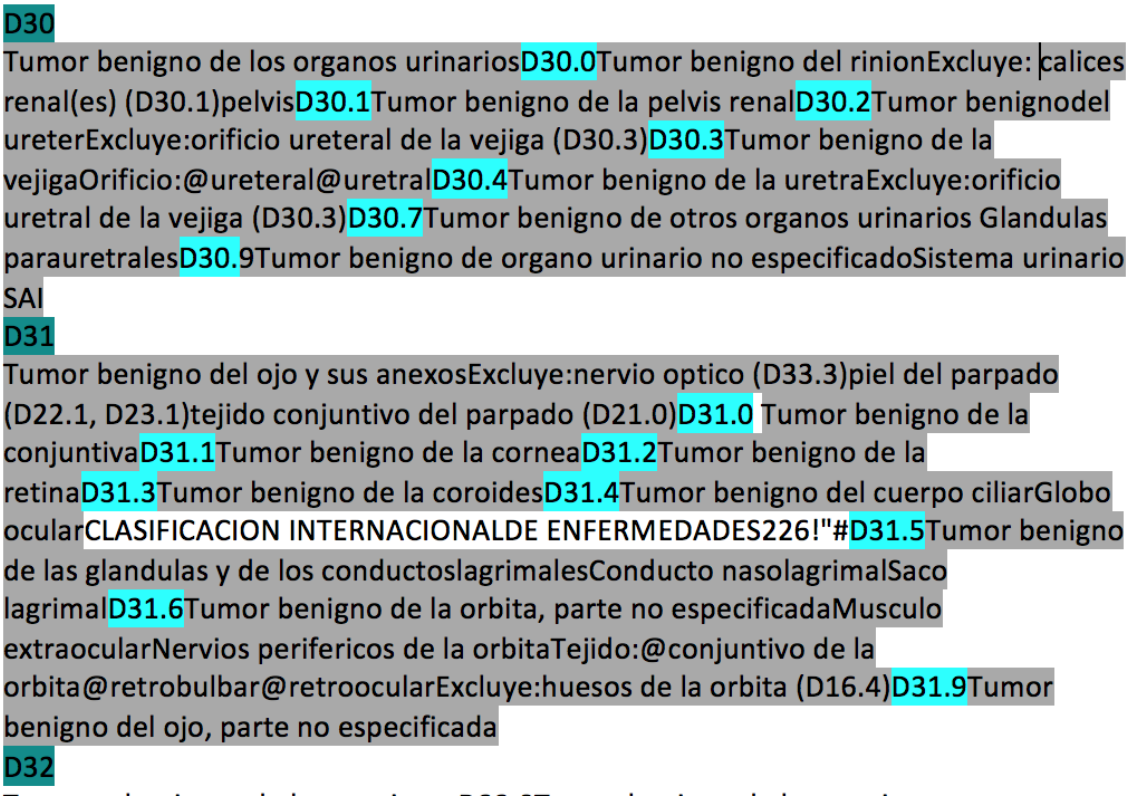


Figura 6.11: Tramo de datos (D30-D31-D32), al final del cuarto paso

6.1.6. Quinto paso, códigos de segundo nivel

Como se señalaba antes, tras la identificación de los códigos principales, es necesario obtener la "información" que se añadía entre códigos, esto es: el término estándar, la información auxiliar -si dispone de ellas- y los códigos de segundo nivel con sus correspondientes datos. El presente proceso es uno de los más costosos, puesto que hay bastantes códigos y estos no deben ser ignorados; y, además, no todos los códigos de segundo nivel que aparecen entre los distintos códigos corresponden al código principal al que pertenecen.

Siendo más preciso, en algunas ocasiones, una información auxiliar puede contener una referencia a un código principal o código de otro nivel. Por ello, el código que se detalla en esta información es sumamente importante que sea tratado como un "código extra", puesto que forma parte de la descripción. Para visualizar de forma gráfica lo comentado, a continuación, se adjunta el ejemplo práctico de la extracción de los datos referentes al código D31:

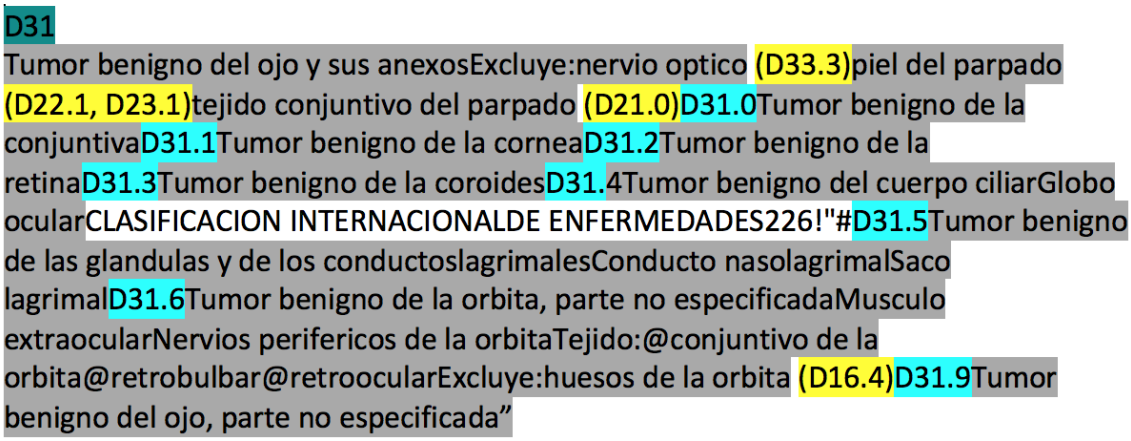

D31
Tumor benigno del ojo y sus anexosExcluye:nervio optico (D33.3)piel del parpado (D22.1, D23.1)tejido conjuntivo del parpado (D21.0)D31.0Tumor benigno de la conjuntivaD31.1Tumor benigno de la corneaD31.2Tumor benigno de la retinaD31.3Tumor benigno de la coroidesD31.4Tumor benigno del cuerpo ciliarGlobo ocularCLASIFICACION INTERNACIONALDE ENFERMEDADES226!"#D31.5Tumor benigno de las glandulas y de los conductoslagrimalesConducto nasolagrimalSaco lagrimalD31.6Tumor benigno de la orbita, parte no especificadaMusculo extraocularNervios perifericos de la orbitaTejido:@conjuntivo de la orbita@retrobulbar@retroocularExcluye:huesos de la orbita (D16.4)D31.9Tumor benigno del ojo, parte no especificada"

Figura 6.12: Datos diferenciados código D31

Como se puede ver en el ejemplo, las partes resaltadas en amarillo son los códigos que acompañan como referencia a los datos. Por lo que para que no sean procesados como un código principal o de segundo nivel, únicamente se obtienen los códigos de segundo nivel que comparten los 3 primeros dígitos con el principal (resaltados en turquesa) y que no se encuentran delimitados por paréntesis:

- Si el código principal es D31 y coinciden los 3 primeros valores de D31.0 y no esta delimitado por paréntesis:
  - Se crea código de segundo nivel
- Sino:
  - Pertenece a la descripción

Una vez realizado este paso el resultado es el siguiente:

```
D31
Tumor benigno del ojo y sus anexosExcluye:nervio optico (D33.3)piel del parpado (D22.1,
D23.1)tejido conjuntivo del parpado (D21.0)
D31.0
Tumor benigno de la conjuntiva
D31.1
Tumor benigno de la cornea
D31.2
Tumor benigno de la retina
D31.3
Tumor benigno de la coroides
D31.4
Tumor benigno del cuerpo ciliarGlobo ocular
D31.5
Tumor benigno de las glandulas y de los conductoslagrimalesConducto nasolagrimalSaco
lagrimal
D31.6
Tumor benigno de la orbita, parte no especificadaMusculo extraocularNervios perifericos d
la orbitaTejido:@conjuntivo de la orbita@retrobulbar@retroocularExcluye:huesos de la
orbita (D16.4)
D31.9
Tumor benigno del ojo, parte no especificada
```

Figura 6.13: Resultado final tras el quinto paso.

6.1.7. Sexto paso, códigos de tercer nivel

Algunos códigos de segundo nivel les corresponde otro nivel, es decir, los códigos de tercer nivel. Sin embargo, es necesario identificar cuáles son, ya que en el documento original no se detallan como códigos pertenecientes al de segundo nivel, sino que, tal y como aparece en la imagen añadida, estos aparecen tras los datos de X enfermedad como información genérica. Por lo que es sumamente importante identificarles, para poder añadirles de manera automática y obtener sus correspondientes datos.

M23 **Trastorno interno de la rodilla**

La siguiente subclasificación complementaria de sitio de compromiso se ofrece para su uso opcional con las subcategorías apropiadas de M23.–; ver también la nota antes de M00–M25.

0 Sitios múltiples
1 Ligamento cruzado anterior o asta anterior del menisco interno
2 Ligamento cruzado posterior o asta posterior del menisco interno
3 Ligamento colateral interno u otro menisco interno y el no especificado
4 Ligamento colateral externo o asta anterior del menisco externo
5 asta posterior del menisco externo
6 otro menisco externo y el no especificado
7 Ligamento capsular
9 Ligamento no especificado o menisco no especificado

Excluye: anquilosis (M24.6)
deformidad de la rodilla (M21.–)
luxación o subluxación recidivante (M24.4)
• rótula (M22.0–M22.1)
osteocondritis disecante (M93.2)
trastornos de la rótula (M22.–)
traumatismo presente —ver traumatismo de rodilla y pierna (S80–S89)

Figura 6.14: Aparición de los códigos de tercer nivel en el Documento original

Es necesario recalcar, que existen diferentes descripciones para los códigos de tercer nivel. Como podemos observar en la figura 6.14 a cada uno de los dígitos le corresponde una descripción, la cual, no es la misma para todos los casos.

Para ello se crea un archivo en el cual se guardan todas las distintas descripciones de tercer nivel en total alrededor de unas 217 descripciones distintas identificadas, para posteriormente identificar a cada uno de los códigos que les corresponde uno de tercer nivel y así crearlos.

A continuación, se pueden ver algunos de los códigos de tercer nivel generados:

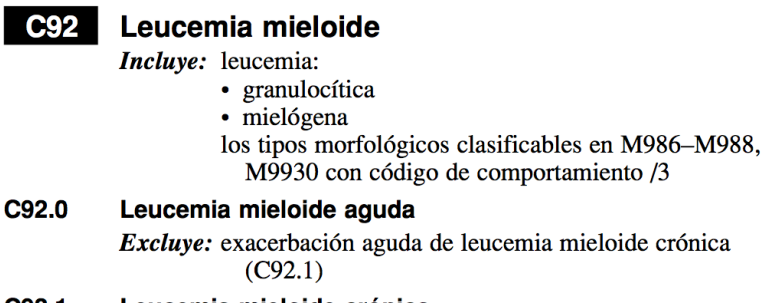
```
M02.80 Sitios multiples
M02.81 Region del hombro
M02.82 Brazo
M02.83 Antebrazo
M02.84 Mano
M02.85 Region pelviana y muslo
M02.86 Pierna
M02.87 Tobillo y pie
M02.88 Otros
M02.89 Sitio no especificado
V29.00 Mientras realiza una actividad deportiva
V29.01 Mientras realiza una actividad de recreacion
V29.02 Mientras trabaja en forma remunerada
V29.03 Mientras esta ocupado en otros tipos de trabajo
V29.04 Mientras descansa, duerme, come o realiza otras actividades vitales
V29.08 Mientras esta ocupado en otras actividades especificadas
V29.09 Durante una actividad no especificada
V29.10 Mientras realiza una actividad deportiva
W10.60 Mientras realiza una actividad deportiva
W10.61 Mientras realiza una actividad de recreacion
W10.62 Mientras trabaja en forma remunerada
W10.63 Mientras esta ocupado en otros tipos de trabajo
W10.64 Mientras descansa, duerme, come o realiza otras actividades vitales
W10.68 Mientras esta ocupado en otras actividades especificadas
W10.69 Durante una actividad no especificada
```

Figura 6.15: Ejemplo de códigos de tercer nivel creados

6.1.8. Séptimo paso,Excluyes e incluyes

Además de estos pasos, como se ha precisado anteriormente, a lo largo del proceso se encuentra información referente a grupos de enfermedades o anotaciones explicativas sobre la familia a las que pertenecen dichos códigos. Esta información, en concreto, no es necesaria para el objetivo de la investigación, por lo que se debe eliminar. Esta eliminación no siempre se ha podido dar de forma automática, por lo que, se ha requerido de un trabajo manual, comparando los resultados obtenidos con los originales.

Junto con esta información, algunas enfermedades contienen una o dos clausulas que añaden información adicional al código. Estos, como se muestra a continuación, son precedidos en el archivo por una palabra clave, "incluye" y "excluye. Estas palabras pueden aparecer de manera conjunta o individual, es decir, un código puede contener un "excluye" o un "incluye", la combinación de ambas o no contener ninguna de las dos:


C92 Leucemia mieloide

Incluye: leucemia:
- granulocítica
- mielógena

los tipos morfológicos clasificables en M986–M988, M9930 con código de comportamiento /3

C92.0 Leucemia mieloide aguda

Excluye: exacerbación aguda de leucemia mieloide crónica (C92.1)

Figura 6.16: Código con clausulas incluye y excluye

Concretamente apuntar que la información aportada por la cláusula incluye, son términos relacionados con la enfermedad. En cambio, los excluyes, como se puede observar en la imagen, hacen referencia a otros términos estándar de una manera más exacta, indicando que hacen referencia a su propio código.

Para el procesamiento de esta información, en primera instancia, se decide eliminar dichos datos, y únicamente quedarse con la información auxiliar que no esta indicada por estas dos palabras clave. Pero tras una observación más exhaustiva, con el fin de realizar una valoración más profunda, se decide obtener y almacenar la información.

La obtención de los excluyes se realiza de manera sencilla al momento de extraer los datos, ya que su cantidad es superior a la de los incluyes y aparece de manera mas intuitiva para obtenerla. En cambio, los incluyes al decidirse obtenerlos en un momento posterior, se obtienen junto al código principal con el que aparece, siendo la manera mas sencilla de obtenerlo. En ciertos casos, el incluye no pertenecía a un código principal, sino a uno de segundo nivel, por tanto se realizo mediante una comparación de los datos originales con los obtenidos para añadirlos de manera correcta.

Finalmente como se ha mencionado, las clausulas excluyes e incluyes aportan información significativa. Sin embargo, para obtener el diccionario final en formato *csv*, se decide excluir los datos que aportan los excluyes, ya que recordemos que el objetivo del presente trabajo es obtener los códigos correspondientes al diagnóstico; y, sin embargo, al incluir los excluyes, se obtiene una información extra que puede llevar a falsos positivos.

No obstante, con el fin de que estos resultados puedan servir para futuras líneas de trabajo, se almacenan en un archivo con los códigos correspondientes.

6.1.9. Problemas y Dificultades surgidas en la obtención del diccionario

En el proceso de obtención de los datos se han dado dificultades y problemáticas que requieren ser mencionadas, ya que han provocado cambios en el prototipo. Por ejemplo, tener que realizar comprobaciones manuales para introducir datos que no se exportaron correctamente o realizar modificaciones de forma manual imposibles de automatizar, separaciones de palabras, etc .

Por tanto al archivo *csv* que obtenemos de forma automática en el primer prototipo, es necesario realizar modificaciones de manera manual para añadir datos y corregir aquellos datos que de manera automática no se han podido generar correctamente. Una vez realizadas las correcciones y arreglos manuales, el archivo *csv* resultante, sera el que se utilice de aquí en adelante, y al que se le aplicara el pre-proceso, convirtiéndose finalmente en el "diccionarioçon los datos la clasificación de enfermedades. A continuación, se exponen algunas de las modificaciones realizadas.

Junto con lo expuesto, es importante precisar que por el formato en el que se encontraba el documento (*pdf*), en el momento de su exportación a texto plano, algunas palabras, especialmente en los saltos de línea, se obtenían en una misma línea sin separación, dificultando su comprensión:

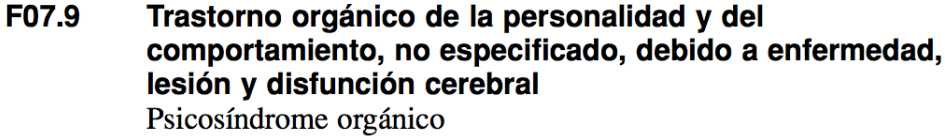


Figura 6.17: Aparición del código F07.9 en el pdf

Se obtenían de la siguiente manera:

“F07.9 Trastorno orgánico de la personalidad y delcomportamiento, no especificado, debido a enfermedad, lesión y disfunción cerebralPsicosindrome orgánico”

Realizar estas divisiones, en ocasiones, resultaba sencillo. Por ejemplo, si se da una combinación de palabras minúscula, seguida de una mayúscula, como la que podemos ver al final del ejemplo “cerebralPsicosindrome”. En cambio, cuando surgía, por ejemplo, “delcomportamiento” se complejiza la posibilidad de automatizar. Por tanto, estas separaciones se realizaron de manera manual, comprobando los datos exportados en texto plano con los originales del *pdf*. No obstante, es preciso apuntar que este hecho no sucede en muchas ocasiones, pero si dificultaba conseguir una descripción correcta y completa de forma simultánea.

Otra dificultad a la hora de identificar la información auxiliar completa era cuando parte de esta información es indicada a través de llaves. Esta información, en la mayoría de los casos, en el texto exportado se situaba al final de la información correspondiente a los códigos y no era posible identificar a que apartado pertenecían. Por tanto, para lograr que esta información se ubique donde le corresponde se realiza de forma manual. Comprobando los datos originales y modificándolos en los datos extraídos.

H33.3 **Desgarro de la retina sin desprendimiento**
Agujero redondeado
Desgarro en herradura
Desgarro SAI
Opérculo
} de la retina, sin mención de desprendimiento

Excluye: cicatrices coriorretinianas consecutivas a cirugía por desprendimiento de la retina (H59.8)
degeneración periférica de la retina sin desgarro (H35.4)

H33.4 **Desprendimiento de la retina por tracción**
Vitreorretinopatía proliferativa con desprendimiento de retina

H33.5 **Otros desprendimientos de la retina**

H34 **Oclusión vascular de la retina**

Excluye: amaurosis fugaz (G45.3)

H34.0 **Oclusión arterial transitoria de la retina**

H34.1 **Oclusión de la arteria central de la retina**

H34.2 **Otras formas de oclusión de la arteria de la retina**
Microembolismo retiniano
Oclusión de arteria retiniana:
- parcial
- de rama

Placa de Hollenhorst

Figura 6.18: Datos en el documento original

Como se puede observar en la figura 6.18, al código de segundo nivel “H33.3” le corresponde una información adicional y a la que también se le añade otra información mediante una llave. Esta información es necesaria identificar y añadirla de manera manual, puesto que, al extraerla del documento, se obtiene del siguiente modo (remarcado en negrita en la figura 6.19).

H33.3Desgarro de la retina sin desprendimiento
Agujero redondeado
Desgarro en herradura
Desgarro SAI
Op”rculoExcluye:cicatrices coriorretinianas consecutivas a cirug™a por desprendimiento de la retina (H59.8)
degeneraciŠn perif”rica de la retina sin desgarro (H35.4)
H33.4Desprendimiento de la retina por traccišn
Vitreorretinopat™a proliferativa con desprendimiento de retina
H33.5Otros desprendimientos de la retina
H34OclusiŠn vascular de la retina
Excluye:amaurosis fugaz (G45.3)
H34.0OclusiŠn arterial transitoria de la retina
H34.1OclusiŠn de la arteria central de la retina
H34.2Otras formas de oclusiŠn de la arteria de la retina
Microembolismo retiniano
OclusiŠn de arteria retiniana:
¥parcial
¥de rama
Placa de Hollenhorst
ENFERMEDADES DELOJO YSUS ANEXOS
415!"#"$**de la retina, sin menciŠn de desprendimiento**

Figura 6.19: Datos obtenidos en texto plano

Además de dichas comprobaciones manuales, también resulta necesario comprobar que todos los códigos de cada una de las enfermedades se exporten correctamente; así como, los códigos de segundo y tercer nivel. De tal modo, se anotan manualmente la cantidad de códigos que aparecen en el documento, y posteriormente los obtenidos en el archivo final. Para visualizar lo comentado, se adjunta la siguiente tabla, en la que se muestran las comprobaciones correspondientes a los códigos principales:

Formato Código	**Cantidad Códigos**		**Formato Código**	**Cantidad Códigos**	
	PDF	**TXT**		**PDF**	**TXT**
AXX	86	86	NXX	82	82
BXX	85	85	OXX	76	76
CXX	88	88	PXX	59	59
DXX	82	82	QXX	87	87
EXX	73	73	RXX	90	90
FXX	78	78	SXX	100	100
GXX	67	67	TXX	95	95
HXX	71	71	UXX	5	5
IXX	77	77	VXX	97	97
JXX	64	64	WXX	88	88
KXX	71	71	XXX	97	97
LXX	72	72	YXX	91	91
MXX	79	79	ZXX	84	84

Tabla 6.3: Comprobación Códigos Principales

Si bien se solventa esta dificultad, aparece otro reto, el de conseguir la coincidencia del numero de códigos de segundo nivel. Para lograr esta coincidencia también se necesitan realizar las comprobaciones de cada uno de los códigos. Como ejemplo visual se exponen las tablas obtenida de las comprobaciones de todos los códigos pertenecientes a la letra A y la letra M.

Código	**Cantidad Códigos 2ºNivel**	**Código**	**Cantidad Códigos 2ºNivel**
A00	3	A51	7
A01	5	A52	7
A02	5	A53	2
A03	6	A54	9
A04	10	A55	0
A05	7	A56	6
A06	10	A57	0
A07	6	A58	0
A08	6	A59	3

A09	0	A60	3
A15	10	A63	2
A16	9	A64	0
A17	4	A65	0
A18	9	A66	10
A19	5	A67	5
A20	7	A68	3
A21	7	A69	5
A22	6	A70	0
A23	6	A71	3
A24	5	A74	3
A25	3	A75	5
A26	4	A77	6
A27	3	A78	0
A28	5	A79	4
A30	8	A80	6
A31	4	A81	5
A32	5	A82	3
A33	0	A83	9
A34	0	A84	4
A35	0	A85	4
A36	6	A86	0
A37	4	A87	5
A38	0	A88	3
A39	8	A89	0
A40	6	A90	0
A41	8	A91	0
A42	6	A92	7
A43	4	A93	4
A44	4	A94	0
A46	0	A95	3
A48	6	A96	5
A49	6	A98	7
A50	9	A99	0

Tabla 6.4: Ejemplo Letra A

En ocasiones, además de los códigos de segundo nivel, también les corresponden los de tercer nivel:

Código	**Cantidad Códigos**		**Código**	**Cantidad Códigos**	
	Códigos 2^{o}Nivel	**Códigos 3^{o}Nivel**		**Códigos 2^{o}Nivel**	**Códigos 3^{o}Nivel**
M00	5	50	M49	7	70
M01	8	80	M50	6	0
M02	6	60	M51	7	0
M03	4	40	M53	6	43
M05	6	60	M54	9	68
M06	7	70	M60	5	50
M07	7	56	M61	7	70
M08	7	70	M62	9	90
M09	4	40	M63	5	50
M10	6	60	M65	7	57
M11	5	50	M66	6	52
M12	7	70	M67	7	63
M13	4	40	M68	2	20
M14	8	80	M70	10	40
M15	7	70	M71	8	75
M16	9	9	M72	7	48
M17	7	7	M73	3	30
M18	7	7	M75	8	19
M19	5	50	M76	10	36
M20	7	7	M77	8	36
M21	10	70	M79	10	100
M22	7	10	M80	8	80
M23	9	90	M81	9	90
M24	10	91	M82	3	30
M25	10	100	M83	8	80
M30	5	50	M84	7	70
M31	10	100	M85	9	82
M32	4	40	M86	9	90

M33	4	40	M87	6	60
M34	5	50	M88	3	22
M35	10	100	M89	9	90
M36	6	60	M90	9	90
M40	6	60	M91	6	60
M41	8	80	M92	10	40
M42	3	30	M93	5	34
M43	9	67	M94	6	52
M45	1	10	M95	8	33
M46	8	73	M96	9	52
M47	5	50	M99	10	100
M48	8	80			

Tabla 6.5: Ejemplo letra M

A continuación, se detallan algunas de las posibles combinaciones con las que pueden aparecer los códigos de tercer nivel:

Ejemplo código Principal: M25

Código 2 Nivel	**Código 3 Nivel**
M25.0	M25.01,M25.02.....M25.09
M25.1	M25.11,M25.12.....M25.19
M25.2	M25.21,M25.22.....M25.29
M25.3	M25.31,M25.32.....M25.39
M25.4	M25.41,M25.42.....M25.49
M25.5	M25.51,M25.52.....M25.59
M25.6	M25.61,M25.62.....M25.69
M25.7	M25.71,M25.72.....M25.79
M25.8	M25.81,M25.82.....M25.89
M25.9	M25.91,M25.92.....M25.99

Tabla 6.6: Ejemplo códigos de segundo y tercer nivel de M25

Ejemplo código Principal: M76

Código 2 Nivel	Código 3 Nivel
M76.0	M76.00 y M76.05
M76.1	M76.15, M76.18 y M76.19
M76.2	M76.25 y M76.29
M76.3	M76.30, M76.36, M76.37 y M76.39
M76.4	M76.40, M76.46 y M76.49
M76.5	M76.50, M76.55, M76.56 y M76.59
M76.6	M76.60, M76.66, M76.67 y M76.69
M76.7	M76.70, M76.76, M76.77 y M76.79
M76.8	M76.80, M76.85, M76.86, M76.87 y M76.89
M76.9	M76.90, M76.95, M76.96, M76.97 y M76.99

Tabla 6.7: Ejemplo códigos de segundo y tercer nivel de M76

Como se puede apreciar, existen diferentes formas de combinarse, y en cada una se le puede asignar una descripción. Por ello, se han realizado y comprobado cada código y término estándar, para que coincidan con el documento, y la página web del Ministerio de Sanidad, Asuntos Sociales e Igualdad.

Duplicación de datos:

Sobre los problemas hallados también resulta interesante detallar las descripciones que contenían información dentro de paréntesis o corchetes. Los corchetes, generalmente, hacen referencia a siglas de enfermedades, como, por ejemplo, "[VIH] Virus de Inmunodeficiencia Humana" o a nombres de enfermedades, bacterias, hongo, etc.

Como se muestra a continuación:

- Shigelosis grupo A [disenteria de Shiga-Kruse]
- Intoxicación alimentaria debida a Clostridium perfringens [Clostridium welchii]
- Carbunco [antrax]

En tanto en cuanto a los paréntesis se refiere, estos añaden información adicional a la descripción, y que en su combinación puede variar a la misma. Con el interés de ayudar a simplificar lo explicado, se añade el siguiente ejemplo:

- Hiperqueratosis palmar o plantar (precoz) (tardía) debida a frambesia
  - Hiperqueratosis palmar o plantar precoz debida a frambesia
  - Hiperqueratosis palmar o plantar tardía debida a frambesia

Este primer ejemplo es una de las muestras más sencillas. Sin embargo, a medida que la cantidad de palabras entre paréntesis aumentan, las combinaciones aumentan y se complejiza su obtención,

tal y como se puede observar en el siguiente extracto:

- Glaucoma (primario) (residual)(con) capsular con seudo-exfoliación del cristalino
  - Glaucoma (primario) capsular con seudo-exfoliación del cristalino
  - Glaucoma(residual) capsular con seudo-exfoliación del cristalino
  - Glaucoma (primario) (residual) capsular con seudo-exfoliación del cristalino

Por tanto, para obtener una información más completa y un mayor número de posibilidades, se tienen en cuenta estas diferencias entre las descripciones, y se realizan todas las combinaciones posibles.

Es importante precisar como limitación que las combinaciones que se generen no tenga un rigor médico; sin embargo, al ser un ámbito profesional distinto y desconocido, se decide no descartar ninguna combinación. De hecho, para poder hacer estas distinciones se requeriría de una persona experta de este ámbito.

6.1.10. Resultado Final CIE-10.csv

El resultado final obtenido es un archivo en formato *csv*. Este, precisamente, contiene: 1) una columna con los códigos, y 2) en la columna contigua los términos estándar, información auxiliar y los datos de los incluyes correspondientes a los códigos, así como los datos que han sido duplicados.

A04	Otras infecciones intestinales bacterianas
	Excluye:
	enteritis tuberculosa (A18.3)
	intoxicacion alimentaria clasificada en otra parte
A04.0	Infeccion debida a Escherichia coli enteropatogena
A04.1	Infeccion debida a Escherichia coli enterotoxigena
A04.2	Infeccion debida a Escherichia coli enteroinvasiva
A04.3	Infeccion debida a Escherichia coli enterohemorragica
A04.4	Otras infecciones intestinales debidas a Escherichia coli
	Enteritis debida a Escherichia coli SAI
A04.5	Enteritis debida a Campylobacter
A04.6	Enteritis debida a Yersinia enterocolitica
	Excluye:
	yersiniosis extraintestinal (A28.2)
A04.7	Enterocolitis debida a Clostridium difficile
	Colitis seudomembranosa
	Intoxicacion alimentaria por Clostridium difficile
A04.8	Otras infecciones intestinales bacterianas especificadas
A04.9	Infeccion intestinal bacteriana, no especificada
	Enteritis bacteriana SAI
A05	Otras intoxicaciones alimentarias bacterianas, no clasificadas en otra parte
	Excluye:
	efectos toxicos de comestibles nocivos (T61-T62)
	infeccion e intoxicacion alimentaria debida a salmonela (A02.-)
	infeccion por Escherichia coli(A04.0-A04.4)
	listeriosis (A32.-)
A05.0	Intoxicacion alimentaria estafilococica

Figura 6.20: Datos del diccionario en formato csv

Antes de usar este documento es preciso eliminar las *stop words*.En concreto, son aquellas palabras que no aportan información al término, así como las preposiciones, conjunciones.... Sin embargo, en este caso las palabras como “sin”, “no”,“otros” y “otras” -generalmente *stop words*-, no son eliminadas, ya que pueden especificar datos sobre la enfermedad a clasificar. Asimismo, al disponer de un “diccionario” en francés, se realiza también un preproceso de sus datos, para homogeneizar los datos eliminando aquellos caracteres que puedan introducir ruido a la hora de calcular las similitudes. Este preproceso es similar al que se explica en el siguiente punto.

Como también se dispone de un “diccionario” en francés, se realizara también un preproceso de sus datos, para homogeneizar los datos eliminando aquellos caracteres que puedan introducir ruido a la hora de calcular las similitudes. Este preproceso es similar al que se explica en el siguiente punto.

6.1.11. Preprocesado términos no-estándar

Ademas de los datos obtenidos de la clasificación internacional de enfermedades, también es necesario preprocesar los términos no-estándar de los que se quiere obtener su código, y así eliminar los caracteres especiales, así como los *stop words* que aparezcan en estos términos.

La herramienta que nos ofrece Python con el módulo “nltk” facilita este proceso, además, es importante considerar que en este proyecto trabajamos con datos en francés y castellano, por ello, se hace el preproceso correspondiente para estos dos idiomas.

El preproceso de los términos en castellano no resulta complicado, comparado con el francés, ya que el abecedario del segundo consta de una amplia variedad de caracteres especiales, como pueden ser tildes y apostrofes, que son necesarias modificar.

Respecto a los datos en castellano se eliminan las *stop words* y las palabras acentuadas, así como el carácter “ñ” y símbolos como pueden ser: guiones (-) o (_), paréntesis o corchetes, interrogaciones y exclamaciones, etc.

En cambio, como se ha comentado, el idioma francés consta de diferentes símbolos y apostrofes que son necesarios eliminar. Como se muestra en la siguiente imagen:

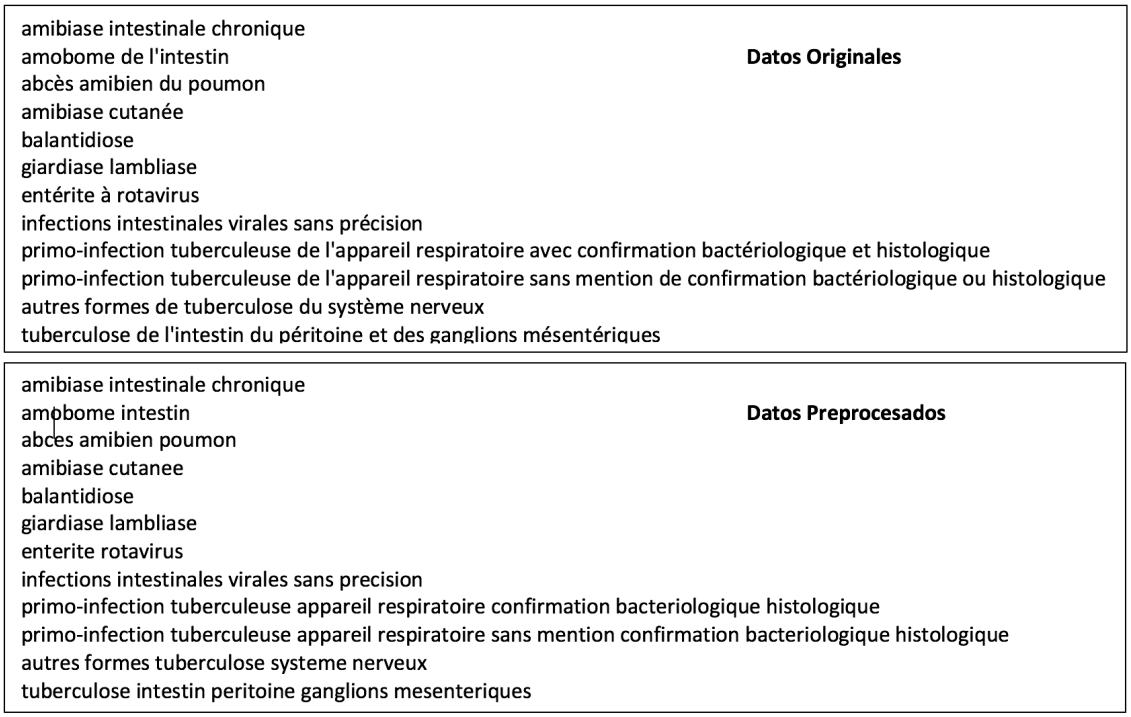


Figura 6.21: Datos antes y después del preproceso, ejemplo datos francés

El diagrama de clases que se muestra a continuación pertenece a este proceso:

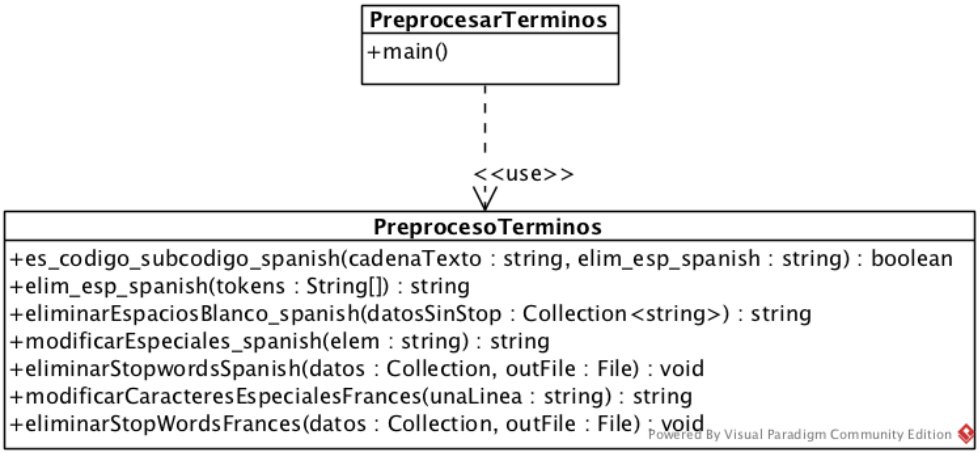


Figura 6.22: Diagrama de clases preproceso términos.

6.2. Prototipo 2: Obtención de similitudes Mediante DKPro Similarity Lexical.

El objetivo principal de este prototipo es la aplicación de distintos algoritmos del módulo *lexical* que proporciona la herramienta "DKPro Similarity". Concretamente, este módulo, facilita la posibilidad de trabajar con cadenas de texto y con n-gramas, de palabras y caracteres.

Para poder aplicar los algoritmos, se analiza la estructura de cada uno, para así, comprender su funcionamiento y poder generar los datos que requieren como parámetros de entrada.

Una vez analizado e identificado el funcionamiento de los algoritmos, se procede, a implementar los pasos necesarios para cargar en la aplicación los datos que se utilizan en el proyecto. Para ello, se deciden las estructuras de datos que almacenaran la información del "diccionario" CIE-10 y los términos no-estándar, en la aplicación. Una vez cargada la información, se recorren los términos que se van a etiquetar y se obtiene su similitud con cada una de las lineas del "diccionario".

A continuación, se presentan los algoritmos explicados en el apartado de análisis de antecedentes y que han sido aplicados en este módulo:

- BoundedSubstringMatchComparator, devuelve un valor respecto a la similitud de dos "strings" según los siguientes criterios:
  - Si son iguales, el valor de similitud indicado 1.0
  - Si comienzan o finalizan las cadenas de texto con la misma palabra, el valor de similitud que indica es 1.0
  - En cualquier otro caso, el valor obtenido es 0.0

- ExactStringMatchComparator, realiza la comparación de dos cadenas de texto o palabras que se indiquen:
  - Si son exactamente iguales, se obtiene el valor 1.0
  - Si no 0.0

- SubstringMatchComparator, compara dos cadenas de texto devolviendo como resultado
  - Si las cadenas son iguales, o una contiene a la otra, el valor es 1.0
  - Si no 0.0

- SimMetricsComparatorImplBase, se encarga de la gestión de los siguientes cálculos de similitud:
  - DiceSimMetricComparator, mediante la aplicación del cálculo de coeficiente Dice (3.4).
  - OverlapCoefficientSimMetricComparator, mediante la aplicación del cálculo de Overlap Coefficient (3.6).
  - CosineSimMetricsComparator.

- SecondStringComparatorImplBase.
  - LevenshteinSecondStringComparator.
  - JaroSecondStringComparator, formula indicada en (3.2).
  - JaroWinklerSecondStringComparator.
  - MongeElkanSecondStringComparator, la siguiente ecuación describe el cálculo que realiza:

$$sim_{mongeElkman}(x,y) = \frac{1}{|x|}\sum_{i=1}^{|x|} \underset{j=1,|y|}{max}\, sim^1(x[i],y[j]) \tag{6.1}$$

Donde, $|x|$ representa el número de elementos en x y sim^1 una función se similitud que utiliza internamente (E.j.: Levenshtein).
Si la cadena de texto únicamente contiene un elemento, la ecuación se simplificaría de la siguiente manera:

$$sim_{mongeElkman}(x,y) = sim^1(x[i],y[j]) \tag{6.2}$$

- LevenshteinComparator, calcula la similitud entre las cadenas de texto indicadas, utilizando la "Distancia Levenshtein", el conjunto mínimo de operaciones de edición necesarias para transformar la cadena A en B (3.1).

- LongestCommonSubsequenceComparator, consta en hallar la subsecuencia común mas larga entre dos cadenas.

- WordNGramJaccardMeasure, calcula la similitud entre dos términos basándose en el coeficiente de Jaccard (3.5)

- WordNGramContainmentMesure, Es similar al cálculo anterior, pero con una variación en la fórmula:

$$S_{Jaccard}(x,y) = \frac{tok(x) \cap tok(y)}{|tok(y)|} = \frac{|tok(x) \cap tok(y)|}{|tok(y)|} \tag{6.3}$$

- characterNGramMeasure: Hace uso de un alfabeto $\sum = \{a,\cdots,\cdots,z,0,\cdots,9\}$ siendo todos los demás caracteres descartados, y se obtienen los n-gramas de ambos términos. Finalmente, se obtiene la similitud de dichos términos utilizando la medida de similitud del coseno. [7]

  Para poder implementar el uso de "CharacterNGramMeasure", la herramienta "DKPro Similarity" hace uso de una estructura de datos que contiene la información de todos los n-gramas, que se pueden generar a partir de los datos que se van a utilizar para comparar. Es decir, en el caso del proyecto, es necesario crear un documento, uno por cada valor de "n", que contenga todos los n-gramas que se puedan formar con los datos del "diccionario". Asignándoles un peso, a estos n-gramas, según su frecuencia de aparición en los datos del "diccionario".

6.3. Prototipo 3: Obtención de similitudes Mediante DKPro Similarity LSA.

Este prototipo se encarga de aplicar el algoritmo *LSA* (Latent semantic analysis) que ofrece "DKPro Similarity", y además genera el espacio semántico necesario para poder ejecutarlo.

Como con los algoritmos del módulo *lexical*, este algoritmo también es analizado, para así identificar los parámetros de entrada que necesita y su funcionamiento.

LSA (Latent Semantic Analysis), en castellano denominado Análisis de la semántica latente. Esta técnica, representa el texto en forma de vectores, estos contienen el contenido semántico de dicho texto, y calcula la similitud de los textos, según el valor del angulo del coseno que forman sus vectores [8] .

Generación Campo Semántico
El campo semántico es necesario para poder realizar el uso de este apartado de "DKPro Similarity", es un documento con la extensión (.sspace). Para crear este documento se hace uso de la librería externa "sspace-2.0.4". Esta librería ofrece la posibilidad de generar un espacio semántico de manera rápida y sencilla, pudiéndole indicar ciertos parámetros. En nuestro caso, los parámetros son los siguientes:

- El documento o documentos, que se utilizaran para generar el espacio semántico:
  En este caso, estos documentos se generan por cada uno de los códigos del "diccionario" CIE-10. Es decir, se agrupan los códigos según su código principal, generando un documento que contenga la información de cada uno de estos. Por ejemplo, tomamos el código "A00", para la creación del documento correspondiente a este código, obteniendo todos los datos del "diccionario" pertenecientes a este código. Para que los datos del documento sean mas consistentes, en el mismo documento, se almacenan también los datos de los códigos de segundo y tercer nivel tiene la misma raiz que el principal, en el caso del ejemplo aquellos que contengan el código "A00". Esto se realiza de la misma manera para los datos en castellano como en francés.

- El algoritmo SVD:
  El espacio semántico al estar generado por vectores, se genera una matriz. En la matriz, las filas representan a cada una de las palabras del texto y las columnas, a su vez, los párrafos. Este algoritmo se encarga de reducir el número de columnas, preservando la similitud estructural de cada una de las filas, es decir, redimensiona la matriz..

6.4. Prototipo 4: Exportar y evaluar resultados

Este módulo se encarga de almacenar los resultados que nos ofrecen los distintos algoritmos de "DKPro Similarity" previamente analizados, o bien de exportar el valor de las figuras de mérito con la evaluación de los resultados, si así se requiere y se dispone de los datos necesarios para realizar la evaluación. Estos resultados se exportan en formato de texto.

Estos archivos contienen la información de los resultados obtenidos para que el usuario pueda entenderla y hacer uso de ella. En estos documentos la información que se puede encontrar es:

- Término no-estándar etiquetado.
- Número del resultado, es decir, si se obtienen mas de uno, cual es su posición, ordenado según su valor de similitud (de mayor a menor).
- Código, es decir, el código que la aplicación sugiere como candidato al término.
- Descripción con la que se ha calculado la similitud del término.
- Valor de similitud obtenido.

Para obtener estos resultados, se tiene en cuenta que la similitud que calculan los algoritmos devuelven un valor comprendido entre 0 y 1, este valor cuanto mas próximo sea a 1, mayor similitud indica el algoritmo que existe entre los términos comparados. En algunos casos, como bien se puede observar en la explicación de los algoritmos previamente realizada, alguno de ellos únicamente devuelven un valor 0 o 1, de manera que estos resultados no aportan una información tan veraz como otros, lo cual se tendrá en cuenta en la evaluación de los resultados.

Si el usuario lo requiere y dispone de los datos necesarios para realizar la evaluacion podrá ejecutar la aplicación y obtener los resultados como la evaluación de los mismos, para poder examinar con estos resultados, el funcionamiento de estos algoritmos.

Para poder evaluar los resultados, se requiere de un corpus de datos, con los cuales poder realizar la evaluación y determinar el funcionamiento de los algoritmos, es decir, disponer de datos con términos no-estándar correctamente etiquetados con sus respectivos códigos CIE-10 correspondientes a información clínica real. Debido a la poca información disponible en castellano y su dificultad de obtención, puesto que, estos datos se encuentran protegidos por la ley orgánica de protección de datos, ha sido necesaria la creación manual de un corpus, y mediante este se ha podido realizar la evaluación con los datos en castellano.

Es necesario puntualizar que este corpus no contendrá una gran cantidad de datos, ya que los datos con los que se genera este, son artículos médicos o documentos relacionados con la medicina; ya que el logro de información médica, y posteriormente su etiquetado en castellano no ha sido posible.

6.4.1. Creación Corpus No-Estándar

Descarga de datos Internet

Considerando lo anterior, los datos utilizados son aportados por los directores y estos, concretamente son descargados de la siguiente página web: Mantra-GSC [9].

Esta información se dispone de la siguiente manera:

- Mantra-GSC (directorio principal)
  - Dutch
  - English
  - French
  - German
  - Spanish

Se utilizan los datos de la carpeta “Spanish”, ya que contiene los datos en el idioma requerido. Dentro de la carpeta, se encuentran las carpetas, “EMEA-ec22-cui-best-ma” y “Medline-EN-ES-ec22-cui-best-ma“’, de donde se obtienen los datos para la creación del corpus. La carpeta “EMEA” posee documentos de la “Agencia Europea de Medicina” y la de “Medline”, aglutina artículos médicos.

Para obtener los datos se aplica un comando a través del terminal, mostrando una lista con los distintos datos de los documentos, la descripción o término de una enfermedad o síndrome y el código CUI (código único de identificación de la enfermedad o síndrome al que hace referencia). He aquí un par imágenes que resultan ilustrativas, de la aplicación del comando y su resultado:

```
cd tu_ruta/EMEA_ec22-cui-best_man
egrep ’DIS’ *ann|egrep -v ’Finding’
```

```
MacBook-Pro-de-Unai:Spanish unai$ EMEA_ec22-cui-best_man egrep 'DIS' *ann|egrep -v 'Finding'
-bash: EMEA_ec22-cui-best_man: command not found
MacBook-Pro-de-Unai:Spanish unai$ cd EMEA_ec22-cui-best_man/
MacBook-Pro-de-Unai:EMEA_ec22-cui-best_man unai$ egrep 'DIS' *ann|egrep -v 'Finding'
0007_d348.u313.ann:#5    AnnotatorNotes T5  "C0029944", "overdose", "Injury or Poisoning" ,"DISO"
0008_d42.u386.ann:#13    AnnotatorNotes T13 "C0026769", "Multiple Sclerosis", "Disease or Syndrome" ,"DISO"
0009_d42.u513.ann:#7     AnnotatorNotes T7  "C0235050", "Tingling of skin", "Sign or Symptom" ,"DISO"
0009_d42.u513.ann:#8     AnnotatorNotes T8  "C0522057", "Numbness of skin", "Sign or Symptom" ,"DISO"
0013_d334.u473.ann:#21   AnnotatorNotes T21 "C0344232", "Blurred vision", "Sign or Symptom" ,"DISO"
0013_d334.u473.ann:#24   AnnotatorNotes T24 "C0012833", "Dizziness", "Sign or Symptom" ,"DISO"
0013_d334.u473.ann:#30   AnnotatorNotes T30 "C0041755", "Adverse reaction to drug", "Injury or Poisoning" ,"DISO"
0016_d347.u428.ann:#11   AnnotatorNotes T11 "C0024131", "Lupus Vulgaris", "Disease or Syndrome" ,"DISO"
0016_d347.u428.ann:#12   AnnotatorNotes T12 "C0409974", "Lupus Erythematosus", "Disease or Syndrome" ,"DISO"
0016_d347.u428.ann:#13   AnnotatorNotes T13 "C0024138", "Lupus Erythematosus, Discoid", "Disease or Syndrome" ,"DISO"
```

Figura 6.23: Resultados del comando para datos EMEA

```
cd tu_ruta/Medline_EN_ES_ec22-cui-best_man
egrep ’DIS’ *ann|egrep -v ’Finding’
```

```
MacBook-Pro-de-Unai:Spanish unai$ cd Medline_EN_ES_ec22-cui-best_man/
MacBook-Pro-de-Unai:Medline_EN_ES_ec22-cui-best_man unai$ egrep 'DIS' *ann|egrep -v 'Finding'
0201_d1967049.u1.ann:#1 AnnotatorNotes T1  "C0339901", "Acute respiratory infections", "Disease or Syndrome" ,"DISO"
0201_d1967049.u1.ann:#2 AnnotatorNotes T2  "C0729531", "Viral respiratory infection", "Disease or Syndrome" ,"DISO"
0203_d2278738.u1.ann:#1 AnnotatorNotes T1  "C0205929", "Anal Fistula", "Acquired Abnormality" ,"DISO"
0203_d2278738.u1.ann:#3 AnnotatorNotes T3  "C0149889", "Anorectal fistula", "Pathologic Function" ,"DISO"
0204_d15612520.u1.ann:#1 AnnotatorNotes T1 "C0752124", "Spinocerebellar Ataxia Type 6 (disorder)", "Disease or
Syndrome" ,"DISO"
0204_d15612520.u1.ann:#2 AnnotatorNotes T2 "C0752124", "Spinocerebellar Ataxia Type 6 (disorder)", "Disease or
Syndrome" ,"DISO"
0206_d4660995.u1.ann:#4 AnnotatorNotes T4  "C0038580", "Substance Dependence", "Mental or Behavioral Dysfunction"
,"DISO"
0206_d4660995.u1.ann:#5 AnnotatorNotes T5  "C1510472", "Drug Dependence", "Mental or Behavioral Dysfunction"
```

Figura 6.24: Resultados del comando para datos Medline

Tras este proceso, se obtienen los datos de interés para generar el corpus, tal y como muestra la siguiente imagen:

EMEA Document	CUI	Description	Medline Document		
0007_d348	C0029944	overdose	0201_d1967049	C0339901	Acute respiratory infections
0008_d42	C0026769	Multiple Sclerosis	0201_d1967049	C0729531	Viral respiratory infection
0009_d42	C0235050	Numbness of skin	0203_d2278738	C0205929	Anal Fistula
0009_d42	C0522057	Tingling of skin	0203_d2278738	C0149889	Anorectal fistula
0013_d334	C0344232	Blurred vision	0204_d15612520	C0752124	Spinocerebellar Ataxia Type 6 (disorder)
0013_d334	C0012833	Dizziness	0204_d15612520	C0752124	Spinocerebellar Ataxia Type 6 (disorder)
0013_d334	C0041755	Adverse reaction to drug	0206_d4660995	C0038580	Substance Dependence
0016_d347	C0024131	Lupus Vulgaris	0206_d4660995	C1510472	Drug Dependence
0016_d347	C0409974	Lupus Erythematosus	0208_d12887862	C0007102	Malignant tumor of colon
0016_d347	C0024138	Lupus Erythematosus, Discoid	0208_d12887862	C0686619	Secondary malignant neoplasm of lymph node
0016_d347	C0024141	Lupus Erythematosus, Systemic	0208_d12887862	C0346629	Malignant neoplasm of large intestine
0016_d347	C0919715	Lupus-like syndrome	0208_d12887862	C0699790	Colon Carcinoma
0024_d218	C0022660	Kidney Failure, Acute	0210_d19174097	C0018193	Granuloma, Foreign-Body
0024_d218	C1565662	Acute Kidney Insufficiency	0210_d19174097	C0012634	Disease
0027_d691	C0002874	Aplastic Anemia	0211_d14502942	C0343401	MRSA - Methicillin resistant Staphylococcus aureus infection
0028_d459	C0439857	Dependence	0212_d2518961	C0401151	Chronic diarrhea
0029_d231	C0029944	overdose	0213_d4408652	C0019189	Hepatitis, Chronic
0030_d101	C0012634	Disease	0215_d13459728	C0000924	Accidents
0031_d61	C0848332	Spots on skin	0218_d11412527	C0278504	Non-small cell lung cancer stage I
0031_d61	C0849850	Skin blotches	0219_d6554001	C0025289	Meningitis
0031_d61	C0033774	Pruritus	0220_d4214540	C0684516	Benign bone neoplasm
0031_d61	C0013404	Dyspnea	0221_d16185630	C0020538	Hypertensive disease
0031_d61	C0240211	Lip swelling	0221_d16185630	C1144799	Hypertensive cardiomyopathy
0031_d61	C0549386	Sensation of warmth	0222_d19558921	C0243069	Hypoplasia
0033_d353	C0032285	Pneumonia	0223_d1981479	C0010417	Cryptorchidism
0033_d353	C0019360	Herpes zoster disease	0224_d3112890	C0011981	Diaphragmatic Eventration
0033_d353	C0019348	Herpes Simplex Infections	0224_d3112890	C0025160	Megacolon
0033_d353	C0041912	Upper Respiratory Infections	0226_d2719189	C0008373	Cholesteatoma
0033_d353	C0037199	Sinusitis	0227_d2486966	C0020514	Hyperprolactinemia
0033_d353	C0006849	Oral candidiasis	0227_d2486966	C0554400	Lactation problem

Figura 6.25: Muestra de los datos guardados en corpus

Habiendo almacenado los datos de interés, se obtienen los datos de CIE-10 correspondientes a los CUIs. Para ello se utiliza el buscador *Metathesaurus Browser*.

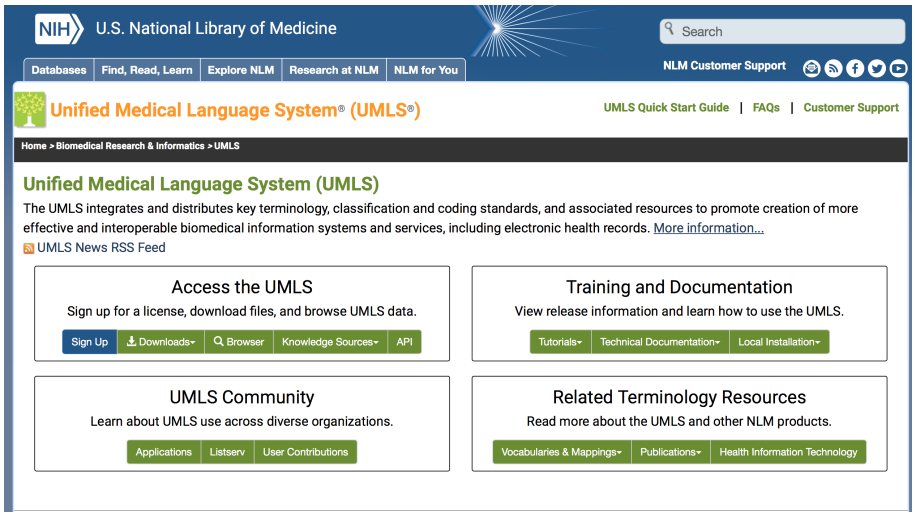


Figura 6.26: Página web https://www.nlm.nih.gov/research/umls/

Para acceder a el buscador, en el apartado de “Access the UMLS” se selecciona la opción “Browser”, siendo necesario tener una cuenta para poder hacer uso de este buscador, y se accederá a la siguiente pantalla:

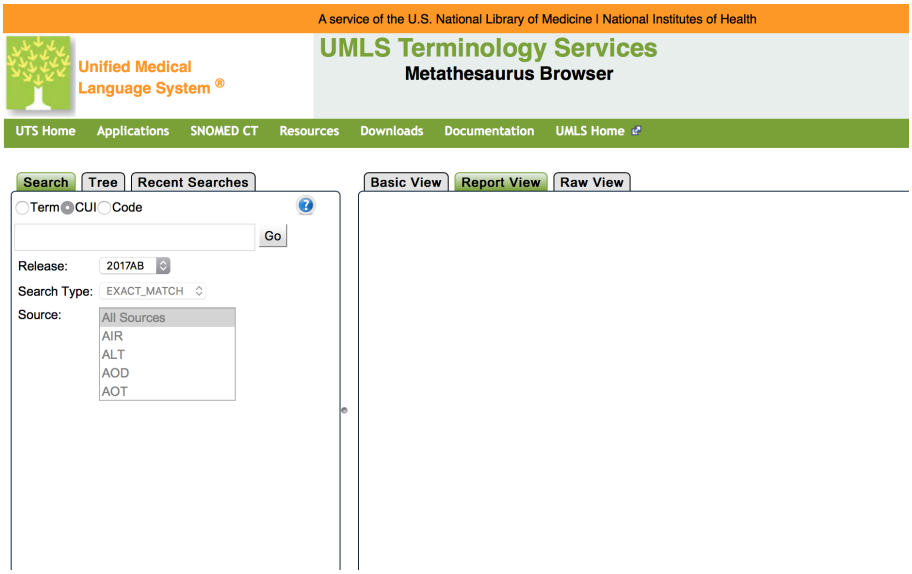


Figura 6.27: Metathesaurus Browser en página web https://www.nlm.nih.gov/research/umls/

En el buscador se introducen los códigos CUI y se comprueba si este dispone de un código CIE-10 correspondiente. Esta información aparece, en la pestaña “Report View” en el apartado “Contexts”; al igual que se puede observar en la imagen adjunta:

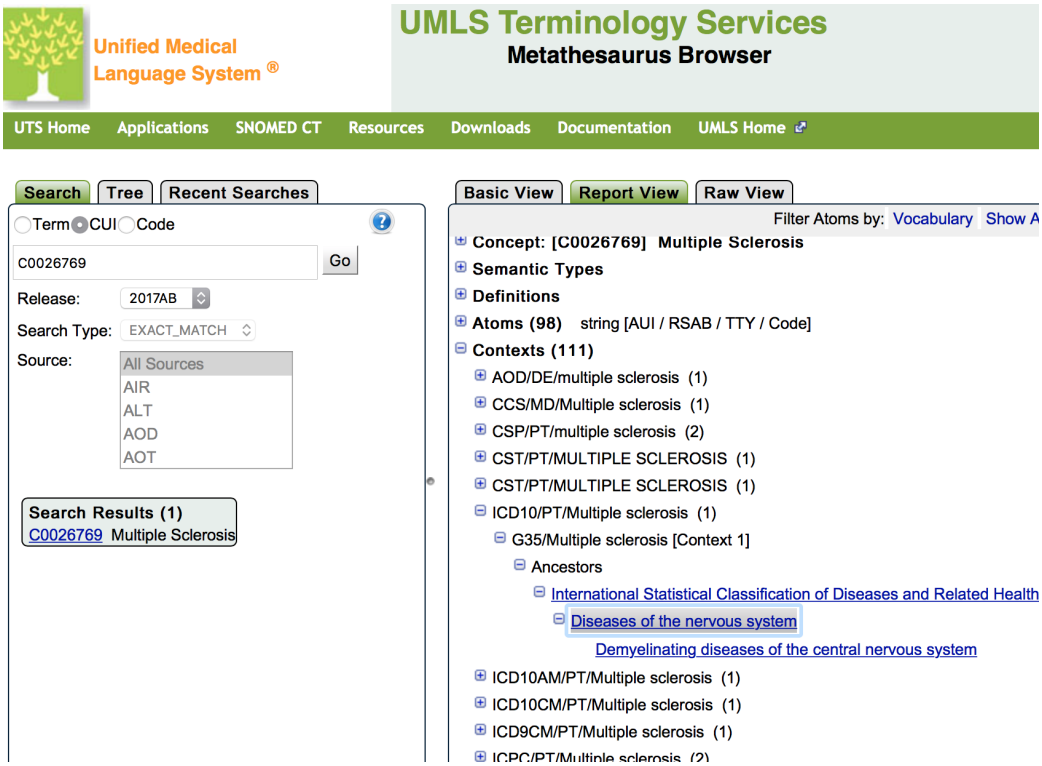


Figura 6.28: Metathesaurus Browser en página web https://www.nlm.nih.gov/research/umls/

Como se puede observar a continuación, al realizar todas las búsquedas, no todos los códigos obtienen su código asociado CIE-10.

0007_d348	C0029944	overdose	
0008_d42	C0026769	Multiple Sclerosis	G35
0009_d42	C0235050	Numbness of skin	
0009_d42	C0522057	Tingling of skin	
0013_d334	C0344232	Blurred vision	
0013_d334	C0012833	Dizziness	
0013_d334	C0041755	Adverse reaction to drug	
0016_d347	C0024131	Lupus Vulgaris	
0016_d347	C0409974	Lupus Erythematosus	L93
0016_d347	C0024138	Lupus Erythematosus, Discoid	L93.0
0016_d347	C0024141	Lupus Erythematosus, Systemic	M32/.9
0016_d347	C0919715	Lupus-like syndrome	
0024_d218	C0022660	Kidney Failure, Acute	N17/.9
0024_d218	C1565662	Acute Kidney Insufficiency	
0027_d691	C0002874	Aplastic Anemia	D61/.9
0028_d459	C0439857	Dependence	
0029_d231	C0029944	overdose	
0030_d101	C0012634	Disease	
0031_d61	C0848332	Spots on skin	
0031_d61	C0849850	Skin blotches	
0031_d61	C0033774	Pruritus	L29/.9
0031_d61	C0013404	Dyspnea	R06.0
0031_d61	C0240211	Lip swelling	
0031_d61	C0549386	Sensation of warmth	
0033_d353	C0032285	Pneumonia	J18.9
0033_d353	C0019360	Herpes zoster disease	B02
0033_d353	C0019348	Herpes Simplex Infections	B00

Figura 6.29: Pequeña muestra resultados tras buscar en metathesaurus

Por tanto, se procede a obtener el código correspondiente a través de la descripción. Para ello, se utilizan los términos en inglés en los siguientes buscadores:

- apps.who.int/classifications/icd10/browse/2016/en
- ecieMaps, este último gestionado por el Ministerio de Sanidad, Igualdad y Servicios Sociales del Gobierno de España.

Con estos recursos, el primero es utilizado para un primer reconocimiento del término ya que se encuentra en inglés, es decir, si es una enfermedad; y el segundo para traducir el término al castellano una vez que se ha comprobado que se atiende a una enfermedad.

Los resultados obtenidos con el buscador pueden aparecer de distintas maneras. Específicamente, puede ser una enfermedad a la que haga referencia un código exacto o una palabra clave que aparezca en más de uno. Para razonar lo comentado se utiliza como ejemplo “Esclerosis Múltiple”.

Atendemos a una enfermedad compuesta por dos palabras “Esclerosis” y “Múltiple”. Sin embargo, al introducir la enfermedad se obtiene lo siguiente:

Figura 6.30: Página eCiemaps del MSSSI, resultados esclerosis múltiple

Como se muestra en la captura, el buscador prioriza “Esclerosis”, y en vinculación a ella se añaden otras expresiones; y entre ellas se puede localizar “Múltiple”, haciendo referencia al código “G35”. De este modo, se obtiene el código para añadir al corpus.

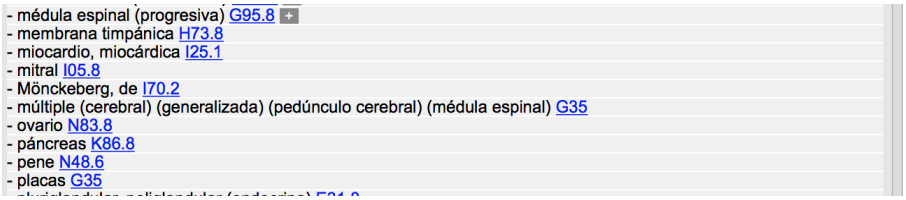


Figura 6.31: Página eCiemaps del MSSSI, resultados esclerosis múltiple

En el análisis de estas variantes, se encuentra que a un mismo término se le pueda asociar más de un código. Esto sucede debido a que el término no especifica una enfermedad concreta, y puede referirse a más de una. En estos casos, se seleccionan para el corpus la mayor variedad de códigos referenciados. Volviendo al ejemplo anterior, si no se hiciese referencia a “Múltiple” no se podría especificar y seleccionar el código exacto, por tanto, es necesario tener en cuenta todas las posibilidades.

Una vez realizado estos pasos, con todas las opciones, obtenemos el corpus que denominaremos “GoldStandard”, el cual se adjunta en esta memoria.

Por otro lado, a los documentos con los que se ha generado el corpus, también les ha sido aplicada la herramienta de etiquetado de enfermedades denominada *Perceptron*, la cual ha sido facilitada por los directores de este TFG, con el fin de obtener un segundo corpus a partir de las etiquetas que esta herramienta detecte en los distintos documentos utilizados.

Este etiquetador analiza el contenido del documento que se le introduce como parámetro y genera como resultado otro documento con las entidades que haya detectado como enfermedad, tal y como las siguientes imágenes hacen referencia:

```
vomitos B-Grp_Enfermedad
, O
reacciones O
alergicas O
, O
nauseas B-Grp_Enfermedad
, O
artralgia O
, O
hipotension B-Grp_Enfermedad
arterial I-Grp_Enfermedad
```

Figura 6.32: Resultados de un análisis *Perceptron*

Una vez recogidas todas las entidades detectadas por *Perceptron*, se procede a obtener los códigos correspondientes. Para ello, se hace uso de las mismas herramientas que se mencionaron previamente.

A este corpus se le denominara “PerceptronCorpus”, el cual se adjunta con este proyecto.

6.4.2. Uso del corpus

Una vez obtenemos el corpus, y, por tanto, los términos no-estándar correspondientes y sus respectivos códigos, se procede a obtener los resultados de similitud entre los términos que forman el corpus y el CIE-10.

En este momento se tendrá la opción a evaluar los resultados, ya que, al disponer de los datos con sus respectivos códigos correctos, es posible comparar si los resultados que nos ofrecen los algoritmos tienen alguna relación con los originales, y poder determinar el funcionamiento de los mismos en la tarea de etiquetado.

En el ultimo tramo de este proyecto se pudo tener acceso a un corpus que contiene datos reales etiquetados por personas expertas. Con el único inconveniente, de que este corpus se encuentra en el idioma francés, aun así, se decide evaluar la aplicación haciendo uso de este corpus y así poder obtener una evaluación más real. Este corpus se denomina *CepiDC Causes of Death Corpus*

Capítulo 7

Verificación y Evaluación

7.1. Verificación de Prototipos

En este apartado se exponen las diferentes pruebas realizadas para verificar el funcionamiento de los prototipos, comprobando si estos funcionan de forma correcta, o no. Para facilitar su comprensión, se exponen las pruebas realizadas a través de tablas que detallan su objetivo y los resultados del prototipo.

7.1.1. Pruebas Prototipo1

En este apartado se muestran las pruebas del Prototipo 1. Este se encarga de obtener los datos del documento *pdf* que contiene la información del diccionario con la Clasificación Internacional de Enfermedades en su versión 10. Y también, del su preprocesamiento y el de los términos no-estándar.

Prueba Prototipo1: Correcta ejecución del comando	
Salida Esperada:	El software se ejecuta sin problemas, accediendo a los directorios correspondientes que contiene el Volumen1.pdf
Resultado obtenido:	Correcto

Tabla 7.1: Prueba Prototipo1: Correcta ejecución del comando

Prueba Prototipo 1: Extracción y exportación de los datos a formato de texto	
Salida Esperada:	Se accede al contenido del archivo Volumen1.pdf del CIE-10 y exporta los datos en formato de texto en el directorio establecido por defecto.
Resultado obtenido:	Correcto

Tabla 7.2: Prueba Prototipo 1: Extracción y exportación de los datos a formato de texto

Prueba Prototipo1: Lectura de los datos y modifica los caracteres mal codificados	
Salida Esperada:	Se accede al archivo que contiene los datos exportados, del documento *pdf*, en formato de texto plano, y modifica los caracteres erróneamente codificados.
Resultado obtenido:	Correcto

Tabla 7.3: Prueba Prototipo1: Lectura de los datos y modificación de los caracteres especiales

Prueba Prototipo 1: Obtención de los códigos principales	
Salida Esperada:	Se localizan los códigos principales en los datos, generando como resultado un documento con estos códigos perfectamente reconocibles y los datos pertenecientes a el.
Resultado obtenido:	Correcto

Tabla 7.4: Prueba Prototipo 1: Obtención códigos principales

Prueba Prototipo 1: Generar los códigos de segundo y tercer nivel	
Salida Esperada:	Se generan los archivos que contienen los códigos de tercer nivel creados, y otro archivo que contiene los códigos principales diferenciados de los de segundo nivel.
Resultado obtenido:	Correcto

Tabla 7.5: Prueba Prototipo 1: Generar los códigos de segundo y tercer nivel

Prueba Prototipo 1: Unir Códigos	
Salida Esperada:	Se genera un archivo con los datos diferenciados según los códigos previamente identificados.
Resultado obtenido:	Correcto

Tabla 7.6: Prueba Prototipo 1: Unir Códigos

Prueba Prototipo 1: Exportar en formato csv	
Salida Esperada:	Se genera un documento en formato *csv*, que contiene los distintos códigos de la clasificación y su correspondiente información.
Resultado obtenido:	Correcto

Tabla 7.7: Prueba Prototipo 1: Exportar en formato csv

Prueba Prototipo 1: Preproceso de los datos CIE-10	
Salida Esperada:	Documento en formato *csv*, que contiene los datos del CIE-10 preprocesados.
Resultado obtenido:	Correcto

Tabla 7.8: Prueba Prototipo 1: Preproceso de los datos CIE-10

Prueba Prototipo 1: Preproceso términos	
Salida Esperada:	Si los parámetros introducidos son correctos, se genera un archivo con los términos preprocesados.
Resultado obtenido:	Correcto

Tabla 7.9: Prueba Prototipo 1: Preproceso términos

Prueba Prototipo 1: Comprobación de los parámetros introducidos en preproceso de términos	
Salida Esperada:	Si alguno de los parámetros indicados son erróneos, se muestra un mensaje de error.
Resultado obtenido;	Correcto

Tabla 7.10: Prueba Prototipo 1: Comprobación parámetros

7.1.2. Pruebas Prototipo2

Pruebas del prototipo 2, encargado de aplicar el modulo *lexical* de "DKPro Similarity".

Prueba Prototipo 2: Parámetros introducidos erróneos	
Salida Esperada:	Se muestra por consola un ejemplo de como se han de indicar los paramentos.
Resultado obtenido:	Correcto

Tabla 7.11: Prueba Prototipo 2: Parámetros introducidos erróneos

Prueba Prototipo 2: Parámetros introducidos correctos	
Salida Esperada:	Se ejecuta la aplicación.
Resultado obtenido:	Correcto

Tabla 7.12: Prueba Prototipo 2: Parámetros introducidos correctos

Prueba Prototipo 2: No es posible la creación de los datos necesarios	
Salida Esperada:	Mensaje informando por consola que no se han podido obtener los datos.
Resultado obtenido:	Correcto

Tabla 7.13: Prueba Prototipo 2: No es posible la creación de los datos necesarios

Prueba Prototipo 2: Datos necesarios para la ejecución creados	
Salida Esperada:	Se generan los datos y se procede con la ejecución.
Resultado obtenido:	Correcto

Tabla 7.14: Prueba Prototipo 2: Datos necesarios para la ejecución creados

7.1.3. Pruebas Prototipo3

Pruebas del prototipo 2, encargado de aplicar el modulo *LSA* de "DKPro Similarity".

Prueba Prototipo 3: Parámetros introducidos erróneos	
Salida Esperada:	Se muestra un mensaje explicativo por consola de como han de introducirse los parámetros.
Resultado obtenido:	Correcto

Tabla 7.15: Prueba Prototipo 3: Parámetros introducidos erróneos

Prueba Prototipo 3: Parámetros introducidos correctos	
Salida Esperada:	Se ejecuta la aplicación.
Resultado obtenido:	Correcto

Tabla 7.16: Prueba Prototipo 3: Parámetros introducidos correctos

Prueba Prototipo 3: Creación de los datos para la ejecución no posible	
Salida Esperada:	Mensaje de información indicando que no se han podido generar los datos.
Resultado obtenido:	Correcto

Tabla 7.17: Prueba Prototipo 3: Creación de los datos para la ejecución no posible

Prueba Prototipo 3: Creación de los datos necesarios correcta	
Salida Esperada:	Se generan los datos necesarios para ejecutar *LSA*, es espacio semántico y los archivos requeridos para su creación.
Resultado obtenido:	Correcto

Tabla 7.18: Prueba Prototipo 3: Creación de los datos necesarios correcta

7.1.4. Pruebas Prototipo4

Comprobación de los resultados obtenidos con los datos de los distintos corpus utilizados.

Prueba Prototipo 4: Parámetros introducidos incorrectamente	
Salida Esperada:	Se muestra un mensaje de como se han de introducir los parámetros o que parámetro no ha sido correctamente indicado.
Resultado obtenido:	Correcto

Tabla 7.19: Prueba Prototipo 4: Parámetros introducidos incorrectamente

Prueba Prototipo 4: Parámetros introducidos correctamente	
Salida Esperada:	Se ejecuta la aplicación.
Resultado obtenido:	Correcto

Tabla 7.20: Prueba Prototipo 4: Parámetros introducidos correctamente

Prueba Prototipo 4: Generar los archivos con los resultados	
Salida Esperada:	Se crean los diferentes documentos de formato de texto con los resultados.
Resultado obtenido:	Correcto

Tabla 7.21: Prueba Prototipo 4: Generar los archivos con los resultados

7.1.5. Pruebas Interfaz

Pruebas realizadas con la interfaz implementada.

Pruebas Interfaz: No se han introducido todos los datos	
Salida Esperada:	Mensaje informativo de faltan datos.
Resultado obtenido:	Correcto

Tabla 7.22: Pruebas Interfaz: No se han introducido todos los datos

Pruebas Interfaz: No se encuentran los datos del CIE-10 situados en la carpeta por defecto.	
Salida Esperada:	Mensaje informativo no se encuentran el diccionario.
Resultado obtenido:	Correcto

Tabla 7.23: Pruebas Interfaz: No se encuentran los datos del CIE-10 situados en la carpeta por defecto

Pruebas Interfaz: Todos los datos necesarios seleccionados y correctamente indicados	
Salida Esperada:	Se generan los datos
Resultado obtenido:	Correcto

Tabla 7.24: Pruebas Interfaz: Todos los datos correctos

7.2. Evaluación

En este apartado se exponen y analizan los resultados obtenidos al realizar la evaluación de los algoritmos implementados en los distintos datos de los corpus utilizados. Posteriormente, se finalizan con unas conclusiones de los resultados conseguidos.

Antes de comenzar con la evaluación, es importante recoger los métodos de evaluación utilizados:

- *precission@k*: donde los valores que toma k son {1,5,7}. Esta evaluación se realiza con los datos tanto en castellano, como en francés. En el caso de los datos de francés se utiliza únicamente el conjunto de datos dev, ya que son los más similares a los de castellano.
- *precission at 1* con el mejor *threshold* obtenido: se escogen los dos mejores algoritmos que ofrecen el resultado más óptimo, según la figura de mérito *precission at 1* de la anterior evaluación. Este *threshold*, sirve para establecer un valor mínimo de similitud a la hora de escoger un resultado. Puntualizar que este apartado únicamente se realiza con los datos en francés por ser los más completos y fiables.

Antes de exponer y comentar los resultados se realiza una introducción de los distintos datos que son utilizados para la realización de la evaluación.

Datos Para Evaluación

Para realizar la evaluación se han utilizado tres corpus diferentes, dos de ellos en castellano y uno en francés. Para facilitar la identificación de los corpus que se han utilizado, se establece un nombre específico a cada uno de ellos, utilizado de aquí en adelante:

- Castellano:
  - SP-EGold: contiene los datos del corpus generado de forma manual, cuyo proceso de obtención se ha presentado en un capítulo previo de este documento.
  - SP-EAuto: este corpus contiene los datos obtenidos tras aplicar el etiquetador de entidades medicas denominado *Perceptron*, a los mismos datos con los que se genera el corpus SP-EGold.
- Francés:
  - FR-Gold: contiene los datos en francés que se utilizan en la evaluación. Estos datos contienen términos no-estándar, es decir, escritos en lenguaje natural de manera espontánea por expertos. Se dividen en tres conjuntos de datos distintos:
    - Train
    - Dev
    - Test
  - FR-Auto: con el idioma francés no se dispone de este corpus, puesto que no se precisa de un etiquetador de entidades médicas que identifique las de este idioma.

Una vez establecidos los nombres con los que se hará referencia a los distintos corpus, procedemos a describir su estructura, la cual consiste en dos documentos de texto:

Por un lado, un documento en el que cada una de sus líneas hace referencia a un término no-estándar que puede representar una enfermedad, síntoma o término médico. Y, por otro lado, un documento que contiene los códigos pertenecientes al CIE-10, que corresponden a cada uno de los términos del primer documento. El orden del contenido de ambos documentos resulta importante y necesario a mantener, puesto que los pares de término-código se encuentran relacionados por su posición en el documento, como se puede observar a continuación:

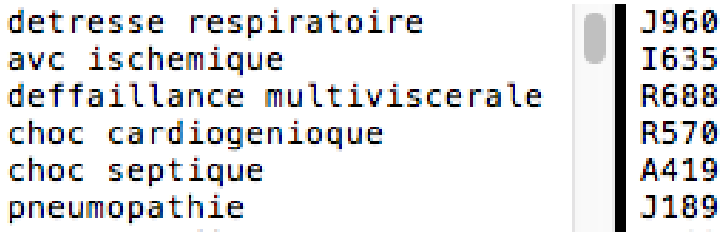


Figura 7.1: Muestra documento FR-Gold

En la figura 7.1,se ilustra una muestra de los términos no-estándar tal y como aparecen en el documento, y, a la derecha, se presenta la muestra correspondiente al documento que contiene los códigos del CIE-10. Por tanto, si tomamos como ejemplo el término no-estándar “choc septique” se puede observar que le corresponde el código “A419”.

En casos especiales, al generar los datos de los corpus en castellano, a los términos no-estándar que se les ha tenido que buscar y asignar su código CIE-10, no hacen referencia a una enfermedad de forma concreta. Por tanto, no se les ha podido asignar un único código, y se les ha asignado más de uno, como se muestra a continuación:

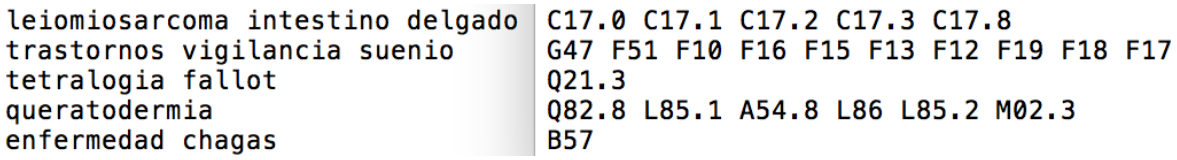


Figura 7.2: Muestra documento SP-EGold

Situándonos en la figura 7.2, escogemos como ejemplos los términos “enfermedad chagas” y “queratodermia”, y si se observan sus respectivos códigos, se puede detectar como a la primera le corresponde “B57” y a la segunda le corresponden seis “Q82.8, L85.1, A54.8, L86, L85.2 y M02.3”.

Además de los diferentes corpus, se hace uso de dos “diccionarios” con la información del CIE-10, uno en castellano y otro en francés, dependiendo del idioma de los datos que se utilizan para evaluar. Estos diccionarios contienen los diferentes códigos que forman parte del CIE-10, así como su correspondiente término estándar e información auxiliar. Su función es la de aportar los datos de la Clasificación Internacional de Enfermedades, y, así, poder realizar la comparación entre el término no-estándar -del que se quiere conocer su código-, con los términos estándar e información adicional que contenga el “diccionario”.

Como sucede con los corpus de castellano y francés, también existe una diferencia de información importante entre ambos. De tal modo, el CIE-10 en francés contiene alrededor de 207.795 términos estándar con su respectivo código, mientras que el CIE-10 en castellano dispone de 47.282 datos, entre los que constan los términos estándar e información auxiliar de estos. La

principal diferencia consta que en el diccionario francés aparecen diferentes posibilidades de denominar a una misma enfermedad a las que le corresponde el mismo código, y en castellano únicamente contamos con el término estándar e información adicional. Es decir, los datos en francés contienen sinónimos o diferentes nombres con los que denominar a la misma enfermedad y en el de castellano únicamente contamos con los datos que forman el documento original.

En la figura 7.3 se pueden observar algunos términos estándar con los que se hace referencia al código "Z824".

```
antecedents familiaux cardiopathies ischemiques autres maladies appareil circulatoire;Z824
trouble rythme cardiaque familial;Z824
terrain familial pro-thrombotique;Z824
terrain familial coronarien;Z824
terrain familial artherosclerose;Z824
terrain familial atheromateux;Z824
risque cardio-vasculaire familial;Z824
insuffisance coronarienne familiale;Z824
hta familiale;Z824
heredite vasculaire;Z824
heredite vasculaire maternelle;Z824
heredite vasculaire paternelle;Z824
heredite familiale cardiaque;Z824
heredite familiale cardio-vasculaire;Z824
heredite familiale vasculaire;Z824
heredite cardio-vasculaire familiale;Z824
heredite cardio-vasculaire;Z824
heredite coronarienne;Z824
facteurs risques cardio-vasculaires familiaux;Z824
facteurs hereditaires familiaux cardio-vasculaires;Z824
coronaropathie familiale;Z824
contexte familial maladie cardio-vasculaire;Z824
antecedent vasculaire familial;Z824
antecedent tdr familial;Z824
antecedents vasculaires familiaux;Z824
antecedents tdr familiaux;Z824
antecedents paternels idm;Z824
antecedents familiaux vasculaires;Z824
antecedents familiaux tachycardie ventriculaire;Z824
antecedents familiaux rupture anevrisme cerebral;Z824
antecedents familiaux rupture anevrisme;Z824
antecedents familiaux problemes cardiaques;Z824
antecedents familiaux necrose myocardique;Z824
antecedents familiaux cardiopathie ischemique;Z824
antecedents familiaux cardiopathie;Z824
antecedents familiaux cmno;Z824
antecedents familiaux coronariens;Z824
antecedents familiaux coronaropathie;Z824
antecedents familiaux crises cardiaques;Z824
antecedents familiaux embolies pulmonaires;Z824
antecedents familiaux idm;Z824
antecedents familiaux infarctus myocarde;Z824
antecedents familiaux infarctus;Z824
antecedents familiaux maladie cardio-vasculaire;Z824
antecedents familiaux maladie thrombo-embolique;Z824
```

Figura 7.3: Términos estándar del código Z824 en diccionario francés.

Preproceso Realizado a Datos del Corpus

A los distintos documentos que contienen los términos no-estándar de los distintos corpus, es necesario aplicarles un preproceso antes de comenzar con la obtención de los resultados y su posterior evaluación. Con el objetivo de conseguir los datos de la manera más limpia posible, se utiliza el preprocesamiento de términos desarrollado en el prototipo1, y así, eliminar palabras innecesarias o que no aportan información las (*stop Words*) y se modifican o elimina los caracteres especiales que aparecen en los datos.

También, de cada corpus se eliminan los datos de aquellas cadenas de texto que no tenga un código CIE-10 asignado, en el caso de los corpus SP-EGold y SP-EAuto aquellas cuyo código este representado por el valor "000" y en el corpus FR-Gold por el valor "UNK" (*unknown* en ingles, desconocido en castellano). Esto se realiza para no introducir desde un comienzo fallos que afecten al resultado de la evaluación.

En ciertos casos del corpus FR-Gold, a una cadena de texto le corresponde más de un código CIE-10. En la mayoría de líneas que se da esta situación se hace referencia a una o más enfermedades, y, por tanto, al desconocer el idioma se decide eliminar las líneas en las que esto sucede, para no realizar divisiones erróneas que introduzcan fallos y ensucien los datos.

A través de la siguiente tabla 7.25 se presentan la cantidad de términos que forman los distintos documentos de los corpus en su origen y tras realizar los cambios antes mencionados. Como se puede apreciar el conjunto de datos train es el que mayor cantidad de términos disminuye, esto se debe a que, en este conjunto, además de eliminar los datos previamente mencionados, únicamente nos quedamos con las cadenas de texto que son únicas, es decir, que no están repetidas en el documento y que también comparten el mismo código.

Corpus	Términos No-Estándar Iniciales	Términos No-Estándar Finales
SP-EGold	202	190
SP-EAuto	98	78
FR-Gold Train	195.203	28.281
FR-Gold Dev	80.899	60.323
FR-Gold Test	91.962	64.924

Tabla 7.25: Cantidad Términos No-Estándar en los corpus para evaluación.

7.2.1. Precisión según valor de K

Como se ha comentado previamente, k toma los valores {1,5,7} con el objetivo de analizar los resultados en relación a la cantidad de códigos obtenidos en cada término del corpus. Para ello, por cada término no-estándar del que se desea obtener su código, se logra su similitud con los datos del diccionario CIE-10; y dependiendo del valor de k, se obtiene el código o códigos del diccionario que obtengan mayor similitud, tantos como indique el valor de k. Una vez obtenidos el código o códigos, se procede a comparar cuántos de estos coinciden con los códigos correctos de cada término del corpus, obteniendo, así, la precisión según el valor de k. De esta manera, se facilita la identificación del algoritmo o algoritmos que mejor resultados consigan.

Este método de evaluación es válido con que únicamente un resultado de los obtenidos por cada término sea correcto, es decir, si el valor de k=5 se obtienen los 5 mejores resultados, y si entre ellos se encuentre el correcto, es un resultado correcto.

Para realizar el calculo de la precisión según k, se hace uso de la siguiente fórmula:

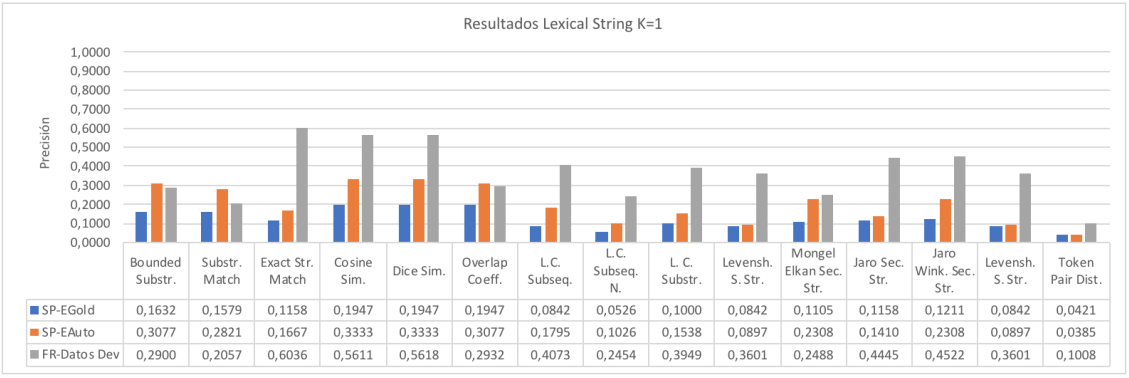
$$Precission@K = \frac{Resultados\ Correctos}{Total\ Resultados\ correctos + Total\ Resultados\ Incorrectos} \tag{7.1}$$

Para realizar esta evaluación se utilizan el corpus SP-EGold, SP-EAuto y FR-Gold, de este ultimo únicamente el conjunto de datos dev, como se ha mencionado al comienzo de este apartado.

Resultados Precission at K={1,5,7}

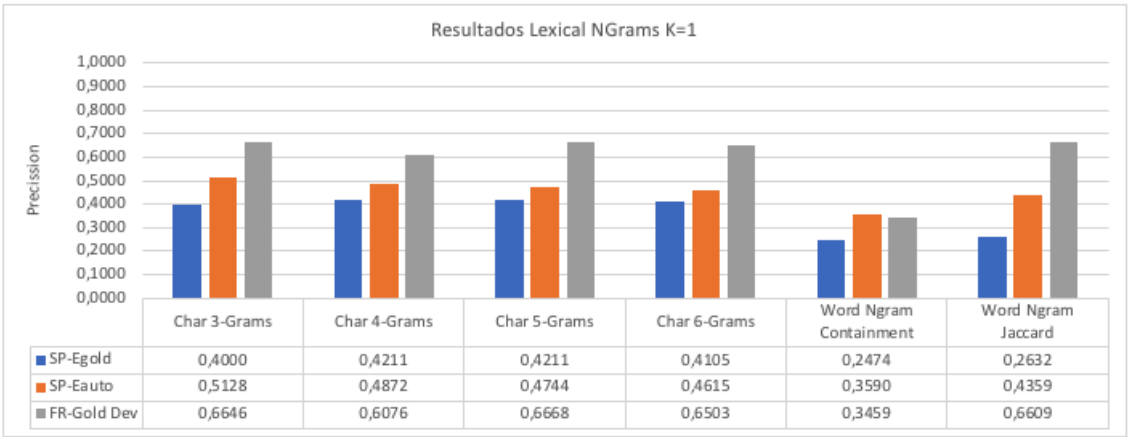
En este apartado se exponen los resultados obtenidos con los distintos valores de K calculados, según la figura de mérito *Precission*. Los resultados se exponen según el valor de k al que corresponden y al módulo de DKPro que pertenecen sus algoritmos.

Resultados K=1



	Bounded Substr.	Substr. Match	Exact Str. Match	Cosine Sim.	Dice Sim.	Overlap Coeff.	L.C. Subseq.	L.C. Subseq. N.	L. C. Substr.	Levensh. S. Str.	Mongel Elkan Sec. Str.	Jaro Sec. Str.	Jaro Wink. Sec. Str.	Levensh. S. Str.	Token Pair Dist.
SP-EGold	0,1632	0,1579	0,1158	0,1947	0,1947	0,1947	0,0842	0,0526	0,1000	0,0842	0,1105	0,1158	0,1211	0,0842	0,0421
SP-EAuto	0,3077	0,2821	0,1667	0,3333	0,3333	0,3077	0,1795	0,1026	0,1538	0,0897	0,2308	0,1410	0,2308	0,0897	0,0385
FR-Datos Dev	0,2900	0,2057	0,6036	0,5611	0,5618	0,2932	0,4073	0,2454	0,3949	0,3601	0,2488	0,4445	0,4522	0,3601	0,1008

Figura 7.4: Resultados Lexical String con K=1



	Char 3-Grams	Char 4-Grams	Char 5-Grams	Char 6-Grams	Word Ngram Containment	Word Ngram Jaccard
SP-Egold	0,4000	0,4211	0,4211	0,4105	0,2474	0,2632
SP-Eauto	0,5128	0,4872	0,4744	0,4615	0,3590	0,4359
FR-Gold Dev	0,6646	0,6076	0,6668	0,6503	0,3459	0,6609

Figura 7.5: Resultados Lexical NGram con K=1

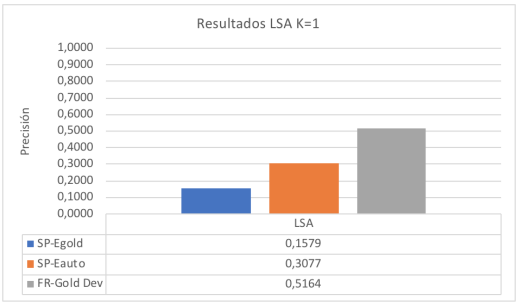
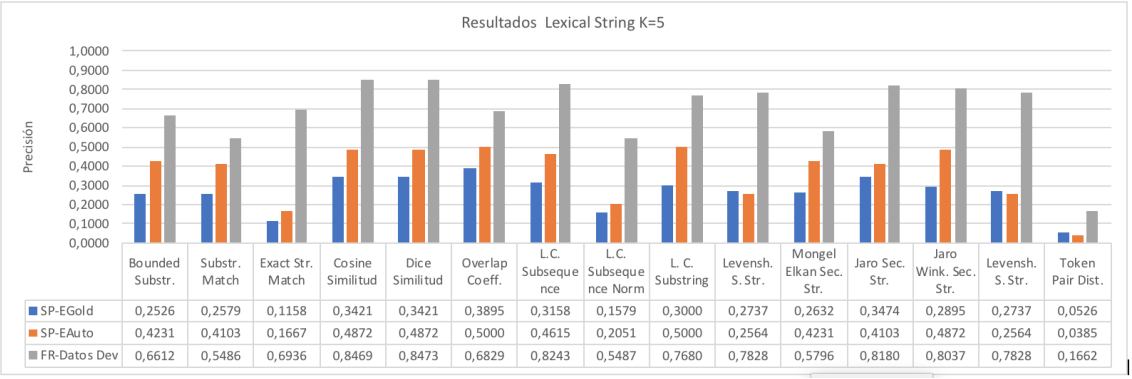


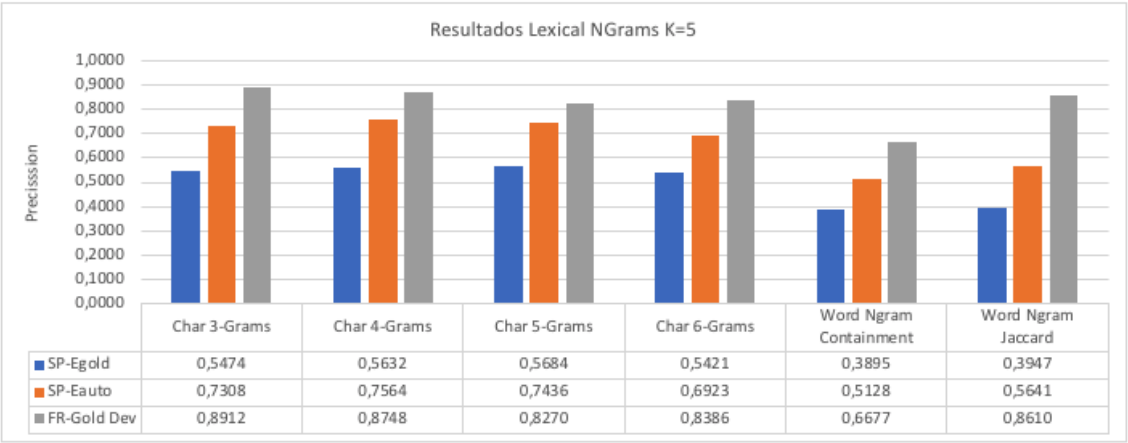
Figura 7.6: Resultados LSA con K=1

Resultados K=5



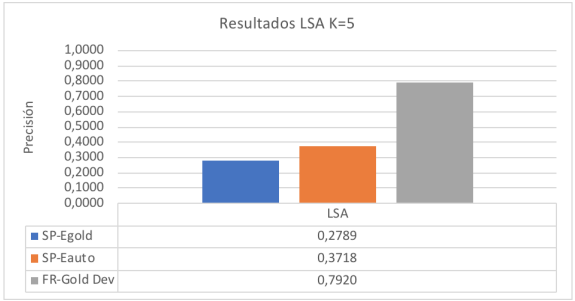
	Bounded Substr.	Substr. Match	Exact Str. Match	Cosine Similitud	Dice Similitud	Overlap Coeff.	L.C. Subsequence	L.C. Subsequence Norm	L. C. Substring	Levensh. S. Str.	Mongel Elkan Sec. Str.	Jaro Sec. Str.	Jaro Wink. Sec. Str.	Levensh. S. Str.	Token Pair Dist.
SP-EGold	0,2526	0,2579	0,1158	0,3421	0,3421	0,3895	0,3158	0,1579	0,3000	0,2737	0,2632	0,3474	0,2895	0,2737	0,0526
SP-EAuto	0,4231	0,4103	0,1667	0,4872	0,4872	0,5000	0,4615	0,2051	0,5000	0,2564	0,4231	0,4103	0,4872	0,2564	0,0385
FR-Datos Dev	0,6612	0,5486	0,6936	0,8469	0,8473	0,6829	0,8243	0,5487	0,7680	0,7828	0,5796	0,8180	0,8037	0,7828	0,1662

Figura 7.7: Resultados Lexical String con K=5



	Char 3-Grams	Char 4-Grams	Char 5-Grams	Char 6-Grams	Word Ngram Containment	Word Ngram Jaccard
SP-Egold	0,5474	0,5632	0,5684	0,5421	0,3895	0,3947
SP-Eauto	0,7308	0,7564	0,7436	0,6923	0,5128	0,5641
FR-Gold Dev	0,8912	0,8748	0,8270	0,8386	0,6677	0,8610

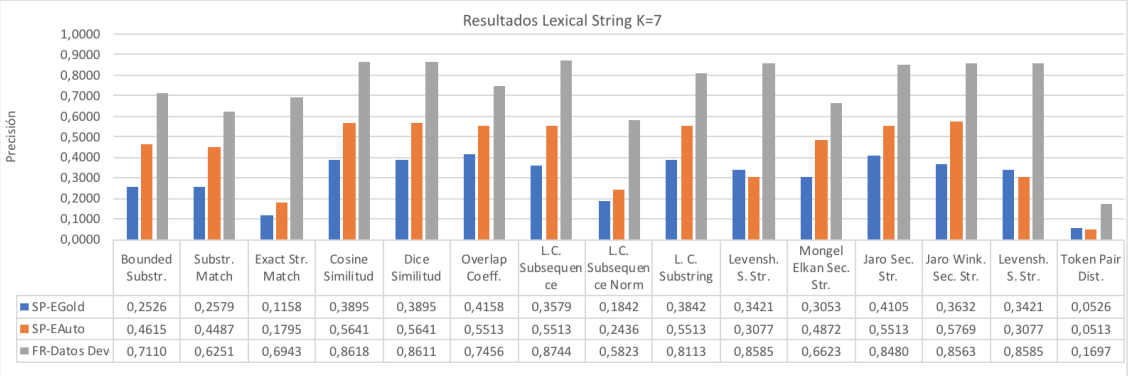
Figura 7.8: Resultados Lexical NGram con K=5



	LSA
SP-Egold	0,2789
SP-Eauto	0,3718
FR-Gold Dev	0,7920

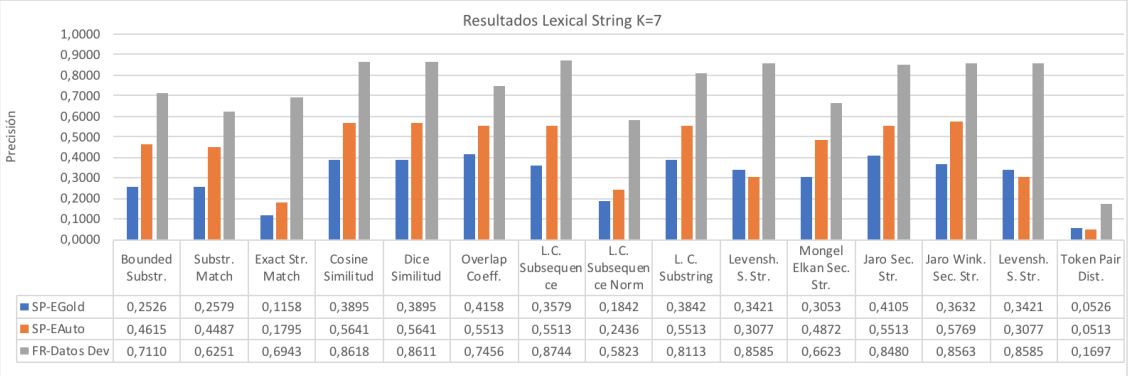
Figura 7.9: Resultados LSA con K=5

Resultados K=7



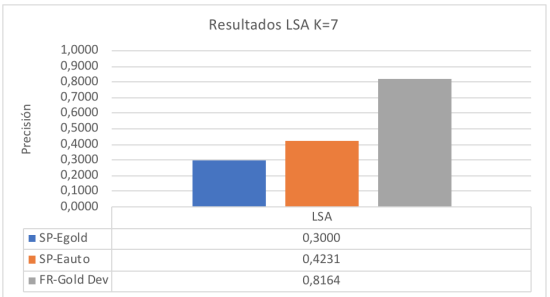
	Bounded Substr.	Substr. Match	Exact Str. Match	Cosine Similitud	Dice Similitud	Overlap Coeff.	L.C. Subsequence	L.C. Subsequence Norm	L. C. Substring	Levensh. S. Str.	Mongel Elkan Sec. Str.	Jaro Sec. Str.	Jaro Wink. Sec. Str.	Levensh. S. Str.	Token Pair Dist.
SP-EGold	0,2526	0,2579	0,1158	0,3895	0,3895	0,4158	0,3579	0,1842	0,3842	0,3421	0,3053	0,4105	0,3632	0,3421	0,0526
SP-EAuto	0,4615	0,4487	0,1795	0,5641	0,5641	0,5513	0,5513	0,2436	0,5513	0,3077	0,4872	0,5513	0,5769	0,3077	0,0513
FR-Datos Dev	0,7110	0,6251	0,6943	0,8618	0,8611	0,7456	0,8744	0,5823	0,8113	0,8585	0,6623	0,8480	0,8563	0,8585	0,1697

Figura 7.10: Resultados Lexical String con K=7



	Bounded Substr.	Substr. Match	Exact Str. Match	Cosine Similitud	Dice Similitud	Overlap Coeff.	L.C. Subsequence	L.C. Subsequence Norm	L. C. Substring	Levensh. S. Str.	Mongel Elkan Sec. Str.	Jaro Sec. Str.	Jaro Wink. Sec. Str.	Levensh. S. Str.	Token Pair Dist.
SP-EGold	0,2526	0,2579	0,1158	0,3895	0,3895	0,4158	0,3579	0,1842	0,3842	0,3421	0,3053	0,4105	0,3632	0,3421	0,0526
SP-EAuto	0,4615	0,4487	0,1795	0,5641	0,5641	0,5513	0,5513	0,2436	0,5513	0,3077	0,4872	0,5513	0,5769	0,3077	0,0513
FR-Datos Dev	0,7110	0,6251	0,6943	0,8618	0,8611	0,7456	0,8744	0,5823	0,8113	0,8585	0,6623	0,8480	0,8563	0,8585	0,1697

Figura 7.11: Resultados Lexical NGram con K=7



	LSA
SP-Egold	0,3000
SP-Eauto	0,4231
FR-Gold Dev	0,8164

Figura 7.12: Resultados LSA con K=7

Análisis de resultados *Precission at K*

En primer lugar, se puede analizar una diferencia entre los resultados obtenidos en castellano y en francés. Esto se debe a la diferencia existente entre los corpus y su contenido, así como la cantidad de datos que conforman cada uno de ellos.

En este sentido, es necesario comentar los resultados significativos obtenidos con los datos que contienen los corpus en castellano SP-EGold y SP-EAuto. Estos se han generado haciendo uso de la misma información, por lo tanto, se puede presuponer que los resultados de ambos deberían ser parecidos o los de SP-EGold mejores. Así, los datos han sido seleccionados de forma manual, en cambio, en SP-EAuto los términos son obtenidas de forma automática mediante un etiquetador de entidades médicas. Sin embargo, como se puede observar en los resultados, esto no sucede debido a que los resultados que se obtienen con SP-EAuto aparentemente son mejores.

Para poder comprender lo mencionado, es necesario tener en cuenta la diferencia existente entre los términos no-estándar que contienen SP-EGold y SP-EAuto:

En la tabla 7.26 se muestran la cantidad de términos que contiene cada uno, y cuantos de ellos coinciden exactamente, es decir, cuantos términos no-estándar a detectado el etiquetador de entidades médicas y se encuentran en SP-EGold.

Corpus	Número Total Términos no-estándar	Términos no-estándar Exactos Compartidos
SP-EGold	190	50
SP-EAuto	78	

Tabla 7.26: Tabla *Exact Match* entre SP-EGold y SP-EAuto

Con estos datos, podemos obtener la precisión de los términos identificados por el detector de entidades médicas que coinciden en ambos corpus y el *Recall* respecto a la cantidad de datos de ambos:

$$Precission = \frac{Term\ no-standar\ Coincidentes}{Term\ no-standar\ SPEAuto} = \frac{50}{78} = 0,641025641 \tag{7.2}$$

$$Recall = \frac{Term\ no-standar\ SPEAuto}{Term\ no-standar\ SPEGold} = \frac{78}{190} = 0,410526316 \tag{7.3}$$

$$\tag{7.4}$$

Observando el resultado de la precisión de los datos obtenidos en SP-EAuto respecto a los términos que coinciden en ambos, podemos deducir que a poco que estos 50 resultados comunes aporten resultados exitosos afectaran de manera más directa al resultado de SP-EAuto que a los de SP-EGold. Esto se puede justificar haciendo uso de la figura de mérito *Recall*, ya que esta nos sirve como guía para conocer la diferencia entre la cantidad de términos que contiene SP-EAuto respecto a SP-EGold, es decir, SP-EAuto no contiene ni la mitad de términos que SP-EGold, y por tanto, a poco que los 50 términos que comparten sean correctos, los resultados de SP-EAuto se verán más afectados que los de SP-EGold, ya que este último depende de los resultados de bastantes más términos.

La diferencia de términos no-estándar que forman un corpus y el otro, se debe a la llamada *subespecificación*que surge cuando se trabaja con texto, es decir, los términos no-estándar identificados mediante el etiquetador, en su mayoría hacen referencia a enfermedades, obviando la información que pueda añadir datos específicos o ignorando síntomas u otros elementos que si se tienen en cuenta en SP-EGold. Esto se debe a que el etiquetador únicamente detecta enfermedades como tal, y al generar el SP-EGold, y al realizarse de manera manual por el humano, este puede suponer que por el contexto de un término se hace referencia a una enfermedad, creándose de esta manera el efecto de *subespecificación*. Utilizando los datos de la tabla 7.27 se puede observar con el término "cáncer pulmón no microcitico", en SP-EGold se toma todo el contexto al que se refiere. En cambio, en SP-EAuto únicamente se queda con la enfermedad en si "cáncer pulmón". De esta manera, es mucho más probable que en el CIE-10 aparezca la enfermedad en sí, que todo su contexto, obteniendo mejores resultados.

La tabla 7.27 contiene una pequeña muestra de como pueden variar los datos que contienen SP-EGold y SP-EAuto, según lo comentando previamente.

Diferencia Términos no-estándar SP-EGold y SP-EAuto	
SP-EGold	**SP-EAuto**
hiperplasia nodular regenerativa higado	hiperplasia nodular
leiomiosarcoma intestino delgado	leiomiosarcoma
anemia aplasica	anemia
insuficiencia renal	insuficiencia renal hepatica
insuficiencia hepatica	
cancer pulmon no microcitico	cancer pulmon
efectos adversos	-

Tabla 7.27: Ejemplo diferencia entre datos SP-EGold y SP-EAuto

Por otro lado, haciendo referencia a los resultados de ambos idiomas, se puede concluir que, entre los distintos algoritmos utilizados, en los que mejores resultados se obtienen son aquellos del módulo *lexical*, que trabajan con n-gramas, ya sean de caracteres o de palabras. Como se espera, a medida en que se aumenta el valor de K, se obtienen mayor cantidad de códigos posibles, y, por ende, la precisión de los resultados aumenta.

Otro resultado que resulta relevante es la diferencia entre las precisiones obtenidas con *LSA*, en el que se destaca una gran diferencia entre los idiomas. Esto se debe a la cantidad de datos generados a partir de los diccionarios para utilizar este método. Es decir, para poder aplicar *LSA*, es necesario generar un espacio semántico lo más amplio posible. Este espacio semántico es creado a partir de documentos que contienen los distintos datos de cada uno de los códigos que forman el diccionario CIE-10. Al igual que se ha precisado al principio de este capítulo, el diccionario que contiene la información del CIE-10 francés alberga una información más completa que el de castellano, obteniendo, así, un espacio semántico mayor y más completo, y que, por ende, aporta mejores resultados.

Para terminar, se puede apuntar que con k=1 los resultados de *Bounded Substr.* y *Substr. Match*, con el corpus SP-EAuto se obtiene un leve mejor resultado, debido al funcionamiento de estos dos algoritmos. Esto se debe a como calculan la similitud, puesto que, al contener este corpus menos cantidad de datos y sus términos no-estándar ser más concretos, facilita que se cumplan las condiciones necesarias para que estos algoritmos indiquen una similitud optima.

7.2.2. Evaluación Dos Mejores Resultados K=1

Dado que el corpus más completo y con mayor cantidad de datos correctamente etiquetados del que se dispone es el francés, se procede a seleccionar las dos opciones que aportan los dos mejores resultados en el apartado previo, con el valor de k=1.

Para realizar este apartado, se utilizan los datos train, dev y test del corpus FR-Gold, de la siguiente manera: los datos del train se disponen para calcular el mejor *threshold*, y una vez obtenido cual es el que maximiza el valor de *f-score* es seleccionado para evaluar los resultados obtenidos con el conjunto de datos dev y test. Cada conjunto de datos contiene datos de diferentes años, el train datos de los años 2006 al 2012, dev del año 2013 y test del año 2014. Para realizar el análisis de los resultados de esta evaluación , se utilizaran los valores obtenidos de las siguientes figuras de mérito:

$$Precission = \frac{Resultados\ Correctos}{Resultados\ Correctos + Resultados\ Incorrectos} \tag{7.5}$$

$$Recall = \frac{Resultados\ Correctos}{Datos\ Totales} \tag{7.6}$$

$$F - Score = 2 * \frac{(Precission * Recall)}{(Precission + Recall)} \tag{7.7}$$

El valor de la precisión representa, cuantos de los resultados obtenidos han sido correctos. Por otro lado, el valor del *recall* representa cuantos códigos correctos se han identificado teniendo en cuenta el total de los datos, tengan o no tengan un resultado. Finalmente *f-score* es la media armónica de estas dos métricas.

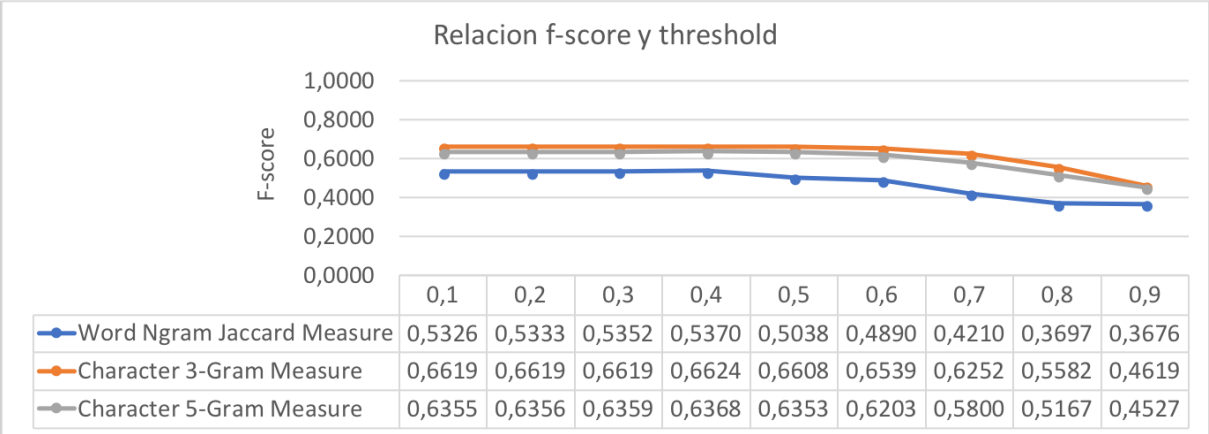
Los resultados obtenidos con *Precission@1*, en que se utilizan los datos del corpus FR-Gold Dev, indican que los dos mejores resultados que se obtienen son con *CharacterNGram*, utilizando n-gramas de 3 y 5 caracteres. Al tratarse de la misma técnica, se decide realizar este apartado también aplicando *WordNGramJaccardMeasure* con el que tercer mejor resultado se ha obtenido.

Elección mejor *Threshold*

El *threshold* que se va a obtener esta comprendido entre el siguiente rango de valores {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}. Este *threshold* representa el valor mínimo de similitud que se dará por valido a la hora de obtener un resultado, es decir, el valor de similitud que se obtenga al comparar el término no-estándar del que se desea obtener su código y el término estándar del diccionario CIE-10, deberá de ser igual o superior al valor del *threshold* indicado.

Una vez que se hayan obtenido el *precission, recall y f-score* con cada uno de los valores posibles del *threshold*, se selecciona el que mayor valor de *f-score* obtenga, y con dicho valor del *threshold* se realiza la evaluación de los resultados con los datos del conjunto FR-Gold dev y test.

Los resultados obtenidos son los siguientes:



	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9
Word Ngram Jaccard Measure	0,5326	0,5333	0,5352	0,5370	0,5038	0,4890	0,4210	0,3697	0,3676
Character 3-Gram Measure	0,6619	0,6619	0,6619	0,6624	0,6608	0,6539	0,6252	0,5582	0,4619
Character 5-Gram Measure	0,6355	0,6356	0,6359	0,6368	0,6353	0,6203	0,5800	0,5167	0,4527

Figura 7.13: Resultados del valor *f-score*

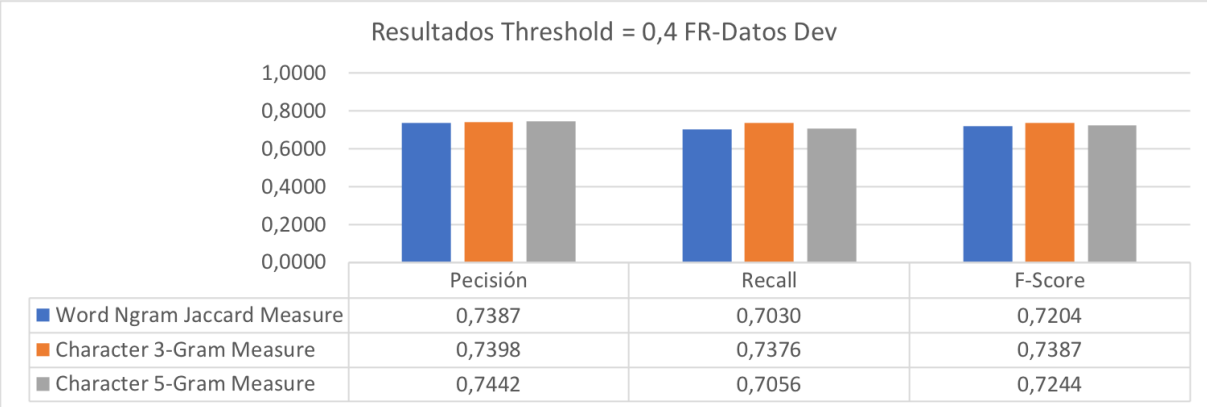
Como se puede apreciar en el gráfico 7.13, los resultados obtenidos con *CharacterNGramMeasure* son ligeramente mejores que los resultados de *WordNGramJaccardMeasure*. Estos resultados son esperados, puesto que con la evaluación de *precission at k*, los resultados obtenidos con n-gramas de caracteres son mejores que los obtenidos con n-gramas de palabras.

Respecto al objetivo de este apartado, para la elección del mejor *threshold* podemos observar cómo el valor de este límite aumenta, y el *f-score* disminuye. Por tanto, según el valor de este se escoge como *threshold*el valor "0.4", ya que representa el que mayor equilibro tiene respecto a su *recall* y precisión.

Se puede afirmar, pues, que no existe mucha variación entre los valores de *f-score* obtenidos. Así, según el límite va en aumento, este comete menos errores al etiquetar los términos no-estándar, pero obtiene muchos menos resultados. En cambio, cuando el valor del límite va disminuyendo se cometen más errores, pero se obtienen más resultados.

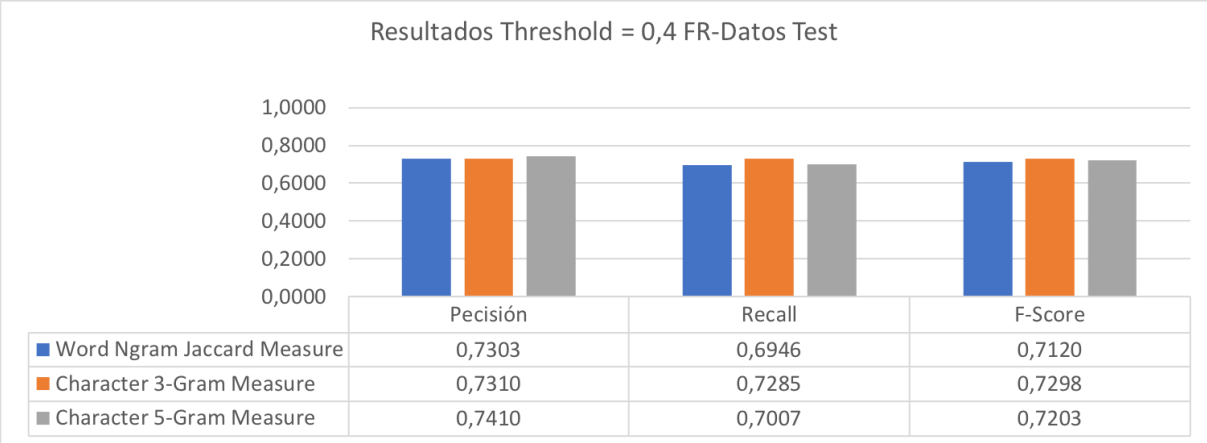
Resultados con el mejor *threshold* obtenido

Esta evaluación se realiza con el conjunto de datos dev y test pertenecientes al corpus FR-Gold. A continuación, se muestran los resultados obtenidos para ambos conjuntos al aplicar las técnicas antes mencionadas, con un valor mínimo de similitud entre el término estándar y no-estándar indicado por el *threshold*, de "0.4".



	Pecisión	Recall	F-Score
Word Ngram Jaccard Measure	0,7387	0,7030	0,7204
Character 3-Gram Measure	0,7398	0,7376	0,7387
Character 5-Gram Measure	0,7442	0,7056	0,7244

Figura 7.14: Resultados *threshold* 0.4 FR-Datos dev



	Pecisión	Recall	F-Score
Word Ngram Jaccard Measure	0,7303	0,6946	0,7120
Character 3-Gram Measure	0,7310	0,7285	0,7298
Character 5-Gram Measure	0,7410	0,7007	0,7203

Figura 7.15: Resultados *threshold* 0.4 FR-Datos test

Los resultados obtenidos en este apartado han resultado ser muy parejos entre ellos. Se puede observar como el valor de *f-score* no varia mucho entre una opción u otra.

7.2.3. Conclusión Evaluación

Una vez expuestos los diferentes resultados obtenidos se procede a realizar una conclusión de los mismos, analizando los aspectos mas relevantes.

Para comenzar, se puede observar en los resultados obtenidos con *precission at k*, que con cualquiera de los valores de k calculados (1,5 y 7) los sistemas que mejores resultados aportan son aquellos que hacen uso de n-gramas, como son *character NGram measure* y *word Ngram*. Esto se debe a la forma en que estos algoritmos realizan los cálculos internamente. Al generar todas las combinaciones posibles de palabras o caracteres, dependiendo del algoritmo, y según el tamaño de n-gramas, aumenta la probabilidad de obtener coincidencias entre ambos términos, y que sea mayor la similitud entre ellos.

También, es necesario destacar que los resultados obtenidos con el módulo *LSA*, han sido mejores de lo esperado con los datos correspondientes al idioma en francés. Por tanto, si se decidiese seguir una línea de investigación utilizando este tipo de técnicas, se podría lograr una mayor cantidad de datos posibles, y, así, poder generar un campo semántico lo más completo posible.

Por otro lado, y no menos importante, apuntar para futuras investigaciones, que hay ciertos algoritmos utilizados que implementan medidas de similitud como son la del *Coseno* o *Dice Coefficient*, y que, dependiendo de los datos utilizados para realizar la evaluación, el valor de la precisión obtenida varia bastante (esto se puede observar en la figura **??**). De tal modo, los valores obtenidos con los datos del corpus francés son superiores a los del castellano. Lo cual nos permite deducir que si se aumentasen los datos con los que trabajar en castellano estos podrían mejorar. Puesto que, al trabajar directamente con los términos completos, tener una amplia variedad de estos, aumentaría la probabilidad de obtener resultados.

Respecto a la obtención del mejor *threshold*, en el establecimiento de un valor mínimo de similitud que cumplan los resultados, se puede observar qué a medida que aumenta el valor de este *threshold*, la precisión de los resultados aumenta y disminuye el *recall*. Dicho de otro modo, el aumento del valor del *threshold*, implica el aumento de aciertos por cada término no-estándar etiquetado, pero, a su vez, disminuye la cantidad de términos no-estándar que obtienen algún resultado. En cambio, cuando el valor del *threshold* disminuye se obtienen mayor cantidad de resultados, pero se cometen más errores en el etiquetado, es decir, aumenta el *recall* y disminuye la precisión obtenida.

Finalmente, comentar que a medida que se aumenta el número de resultados a obtener por cada término no-estándar que se desea etiquetar, la aplicación obtiene mejores resultados. Esto implica que para hacer uso de esta aplicación como guía para obtener los códigos CIE correspondientes al término o términos no-estándar requeridos obtendría mejores resultados y más fiables, que hacer uso de ella como un identificador exacto de los códigos. Es cierto que los resultados obtenidos en el logro de un único resultado en el término no-estándar han sido superiores a lo esperado con algunos sistemas; no obstante, aún está muy lejos de ser completamente fiable, siendo necesaria la intervención de una persona experta.

Capítulo 8

Conclusiones y Trabajo Futuro

Una vez finalizado el proyecto, se procede a realizar un análisis crítico tanto del proyecto en general, como de carácter personal y de líneas futuras.

8.1. Conclusiones Generales

En lo que respecta al proyecto, se puede destacar la creación del "diccionario" como uno de los procesos de mayor valor. Sin el "diccionario" no se podrían obtener los códigos correspondientes a los términos que se quieren etiquetar y, por ello, los esfuerzos personales y recursos del proyecto han estado centrados en esta dirección. Además, como bien se puede observar en el plan de gestión de riesgos, muchos de los riesgos que se mencionan se han dado, afectando, por consecuencia, al desarrollo del proyecto. De esta forma, ha resultado imposible aplicar técnicas relacionadas con las redes neuronales o *word embeddigs* (aunque de igual modo hubiese sido un proceso difícil, debido a la falta de datos). Esto es, uno de los procesos vitales de este proyecto es esa recolección de daros, ya que sin datos no seria posible avanzar en este ámbito. No obstante, cabe precisar que al programar los planes de contingencia ha sido posible superarlos.

Otro aspecto importante en lo que se refiere a los datos, se plantea al realizar la evaluacion de las técnicas implementadas, ya que en esta es necesario disponer de términos no-estándar referentes al ámbito clínico correctamente etiquetados. De hecho, al estar estos datos protegidos por la ley orgánica de protección de datos, ha sido imprescindible la creación artificial de dichos datos.

Una vez que se han conseguido los datos mínimos necesarios para evaluar los resultados del proyecto, estos han sido mejores de lo esperado, tal como se comenta en el apartado de evaluación. Aun así, comentar que el software funciona mejor como guía y apoyo a la hora de identificar las enfermedades -proponiendo los mejores candidatos para esta-, que como herramienta que automatiza todo el proceso.

Finalmente, es importante destacar que, tal y como se incide en la evaluación de los datos en francés, los resultados resultan más positivos, óptimos y, por ende, mejores, cuando se cuenta con un número más amplio de datos. Es decir, generar y/o contener corpus con un volumen importante de datos -y que sean de calidad-, ayudan a albergar cuotas de investigación más ambiciosas y ricas (además, de completas).

8.2. Conclusiones personales

En este apartado se van a realizar valoraciones más personales respecto al trabajo realizado, así como de los objetivos personales fijados por uno mismo y el cumplimiento de ellos.

En primer lugar, me gustaría destacar el enriquecimiento personal que ofrece realizar un trabajo de esta índole. Durante el proceso académico se fomenta el trabajo en grupo, que, sin duda, en una profesión como la informática se percibe como un requisito sumamente necesario; sin embargo, afrontar un trabajo de estas características de forma individual (aunque se esté guiado y tutorizado) se convierte, en cuanto menos, en un reto personal, ya que se precisa de una búsqueda de herramientas propias.

También, es preciso reconocer el aporte de conocimiento que posibilita este trabajo. Concretamente, este ha favorecido conocer dos herramientas útiles; por un lado, otro lenguaje de programación -distinto al que se utiliza en la carrera-, Python, y, por el otro, el editor de texto Latex. Además de consolidar estos dos procesos de aprendizaje, también se deben destacar las habilidades personales ganadas en este enfrentamiento a "nuevos espacios", ya que me ha permitido desarrollar habilidades y destrezas personales para resolver y gestionar problemas que,sin ninguna duda, me resultarán útiles para el futuro.

Finalmente, me gustaría poner en relevancia la impresión general obtenida de este proceso. Durante la carrera se pone énfasis en los procesos de gestión de los datos, sin embargo, pocas veces, como alumno, me había cuestionado la importancia de obtener dichos datos. Sin datos, no se pueden generar procesos de programación y diferentes tareas. De hecho, este proceso me ha ayudado a comprender, y vivir en primera persona la dificultad y la complejidad con la que se obtienen los datos. Estas vivencias han estado especialmente presentes en la creación del "diccionario" y los corpus, en los que se ha tenido que aprender a gestionar los diferentes problemas surgidos del uso de la lengua, como son la ambigüedad, subespecificación, errores en el uso de la misma, etc.

Esta dificultad y complejidad en muchas ocasiones ha generado frustraciones y vivencias negativas con respecto al proyecto, ya que ha sido un proceso largo y en el que se han tenido que superar diferentes obstáculos; ejemplo de ello es la creación del Corpus que se ha explicado. Así, si a veces uno se cuestiona si no se han dado pasos relevantes para la investigación, este trabajo enseña cuán importante son los procesos previos a una investigación o proyecto, y que en muchas ocasiones quedan invisibilizados, ya que nos quedamos con "la carátula final" o "el proyecto que se vende" sin prestar atención en los esfuerzos silenciados previamente dados.

8.3. Trabajo Futuro

A lo largo del desarrollo del proyecto para poder cumplir los distintos objetivos fijados, han ido surgiendo diferentes planteamientos que no han podido ser implementadas en este proyecto, debido a que nos acotamos a un determinado tiempo.

En esta línea, primero, exponer, y como se ha mencionado previamente, que por motivos relacionados con la falta de datos para realizar este proyecto, se ha destinado bastante tiempo a la obtencion de ellos, y, por tanto, algunas técnicas que se pensaban incorporar a la aplicación no se han integrado, como son las redes neuronales y el uso de word embeddings. De este modo, resulta especialmente interesante poder abrir líneas futuras de trabajo en esta dirección.

También, resultaría sumamente ejemplar poder usar datos de otros idiomas. En este momento la aplicación solo permite seleccionar y utilizar datos en castellano y francés, pero seria una propuesta interesante ampliarlo para el uso de mas idiomas, siempre y cuando se dispongan de los datos. De hecho, recordemos que atendemos a criterios internacionales (el mismo CIE).

Otro aspecto a seguir explorando, seria, por ejemplo, el preprocesado de los datos en otros idiomas, como es el caso del francés. Estos datos, incluían en la misma cadena de texto una o mas enfermedades y por consiguiente, más de un código. Al desconocer el idioma y por motivos de tiempo, estas son eliminadas, por lo que una ampliación seria identificar cada una de estas enfermedades y asignarles su código correspondiente, aumentando, así, los datos y obteniéndolos de una forma más limpia.

Además de todo lo comentado, dado el proceso del propio proyecto, se considera especialmente importante seguir ampliando el "diccionario" CIE-10. La posibilidad de ampliar el"'diccionario" con información más completa, posibilitaría mejorar los resultados. Es decir, tal y como se expresaba anteriormente, considerando los resultados logrados en francés, se considera que resulta más rico tener una mayor fuente de datos, ya que posibilita correlacionar más términos a un mismo código. Así, continuar con la ampliación de este "diccionario" se convierte, en sí misma, en una futura línea de trabajo, pudiendo generar candidatos, y, por ende, lograr e incorporar sinónimos u otros términos equivalentes que se usan de manera asidua en el ámbito clínico y que delimitan una misma enfermedad. En definitiva, aumentar los datos del "diccionario" en castellano, posibilitaría lograr resultados más óptimos, permitiéndonos comparar los términos estándar, con los no-estándar de los que se quiere obtener su código en la Clasificación Internacional de Enfermedades.

Por último, y en el mismo sentido que se proponía en la idea anterior, se cree que poder contar con datos reales que contengan términos no-estándar y estén etiquetados con su código CIE correcto por expertos en castellano, habilitaría realizar pruebas mucho más fiables en este idioma.

Anexos

Anexos A

Casos de uso Extendidos

A.1. Prototipo 1: Obtención, Creación y Prepocesamiento de los Datos

A.1.1. Caso de uso Extraer datos del CIE-10

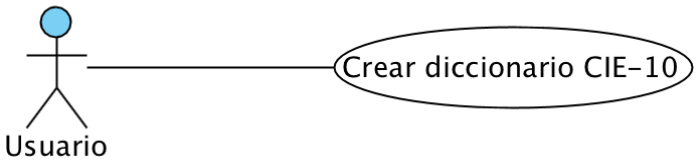


Figura A.1: Caso de Uso extendido: Extraer datos CIE-10

Nombre : Crear diccionario CIE-10	
Descripción	El usuario al ejecutar el archivo Python encargado de este proceso, se extraen los datos del documento pdf original y se realizan los pasos para obtener los datos en formato csv.
Actores	Usuario
Precondición	Disponer del archivo inicial con los datos del CIE en formato pdf.
Postcondición	Se genera el documento en formato csv que contiene los datos extraídos. Por otro lado, se generar archivos de texto que contienen los datos de pasos intermedios del proceso.

Tabla A.1: Caso de uso extendido: Extraer datos del archivo CIE-10

A.1.2. Preprocesar CIE

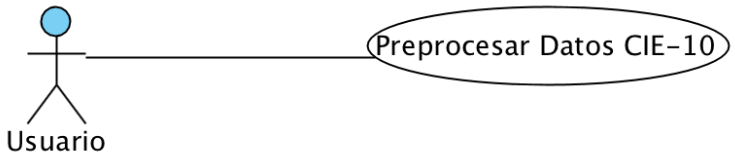


Figura A.2: Caso de Uso extendido: Preprocesar CIE

Nombre: Preprocesar datos CIE-10	
Descripción	Se preprocesan los datos del archivo en formato csv obtenido en el paso de extracción de datos, es decir, se eliminan las preposiciones, y se duplican los datos que se encuentren entre corchetes y paréntesis.
Actores	Usuario
Precondición	Que exista el archivo csv con los datos del "diccionario" en el directorio especificado.
Postcondición	Se obtiene un archvio cie10E.csv con los datos finales.

Tabla A.2: Caso de Uso extendido: Preprocesar CIE

A.1.3. Preprocesar Términos no-entandar

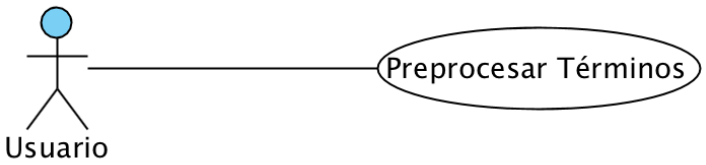


Figura A.3: Caso de Uso extendido: Preprocesar Términos

Nombre: Preprocesar etiquetas	
Descripción	Se encarga de preprocesar los distintos datos de los términos no-estándar que se quieren etiquetar, eliminando caracteres especiales y *stop words*.
Actores	Usuario
Precondición	Que exista el documento con los términos no-estándar que indique el usuario y se indique el idioma.
Postcondición	Se obtienen los términos introducidos por el usuario preprocesados.

Tabla A.3: Caso de Uso extendido: Preprocesar Etiquetas

A.2. Prototipo 2: Obtención de similitudes Mediante DKPro Similarity Lexical

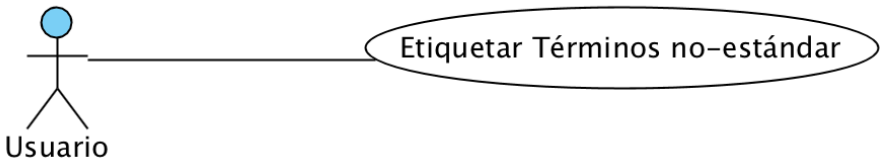


Figura A.4: Caso de Uso extendido: Obtener similitud Lexical

Nombre: Etiquetar Términos no-estándar con Lexical	
Descripción	Calcula la similitud entre los términos no-estándar y los datos del diccionario, haciendo uso de distintas métricas de similitud de textos. Asignándole al término el código correspondiente al dato o datos del diccionario con los que mayor similitud logre.
Actores	Usuario
Precondición	El usuario introduce la ruta del archivo que contiene los términos que quiere etiquetar.
Postcondición	Se generan los resultados de los datos etiquetados.

Tabla A.4: Caso de Uso extendido: Obtener similitud Lexical

A.3. Prototipo 3: Obtención de similitudes Mediante DKPro Similarity LSA

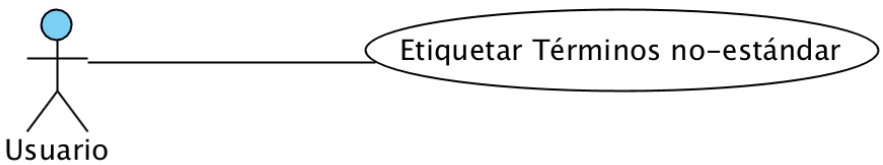


Figura A.5: Caso de Uso extendido: Obtener similitud LSA

Nombre: Etiquetar Términos no-estándar con LSA	
Descripción	Calcula la similitud entre los términos no-estándar y los datos del diccionario, según “DKPro Similarity” y su algoritmo LSA. Asignándole al término el código correspondiente al dato o datos del diccionario con los que mayor similitud logre.
Actores	Usuario
Precondición	El usuario introduce la ruta del archivo que contiene los términos que quiere etiquetar.
Postcondición	Se generan los resultados de los datos comparados.

Tabla A.5: Caso de Uso extendido: Obtener similitud LSA

A.4. Prototipo 4: Evaluar resultados

Figura A.6: Caso de Uso extendido: Evaluar resultados

Nombre: Evaluación Resultados	
Descripción	Facilita al usuario, datos con los que poder interpretar el funcionamiento de las distintas métricas de similitud de textos aplicadas al etiquetar los términos no-estándar.
Actores	Usuario
Precondición	Disponer de los datos que se desean etiquetar, así como, los códigos correctos correspondientes a estos datos.
Postcondición	Se exportan los datos etiquetados, asi como, los datos de la evaluación. Términos correctamente etiquetados, cuantos fallos y las figuras de mérito correspondientes.

Tabla A.6: Caso de Uso extendido: Evaluar resultados

Anexos B

Diagramas de secuencia

B.1. Prototipo 1: Obtención, Creación y Prepocesamiento de los Datos

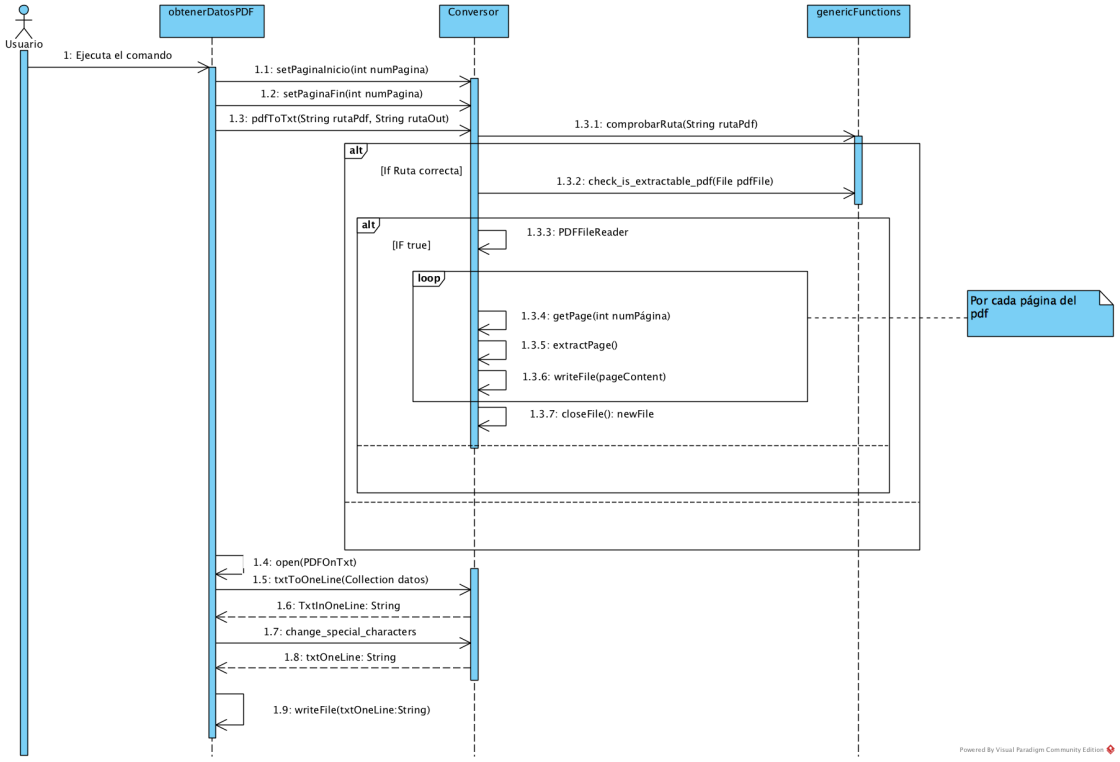


Figura B.1: Diagrama de secuencia para obtener el pdf en formato de texto

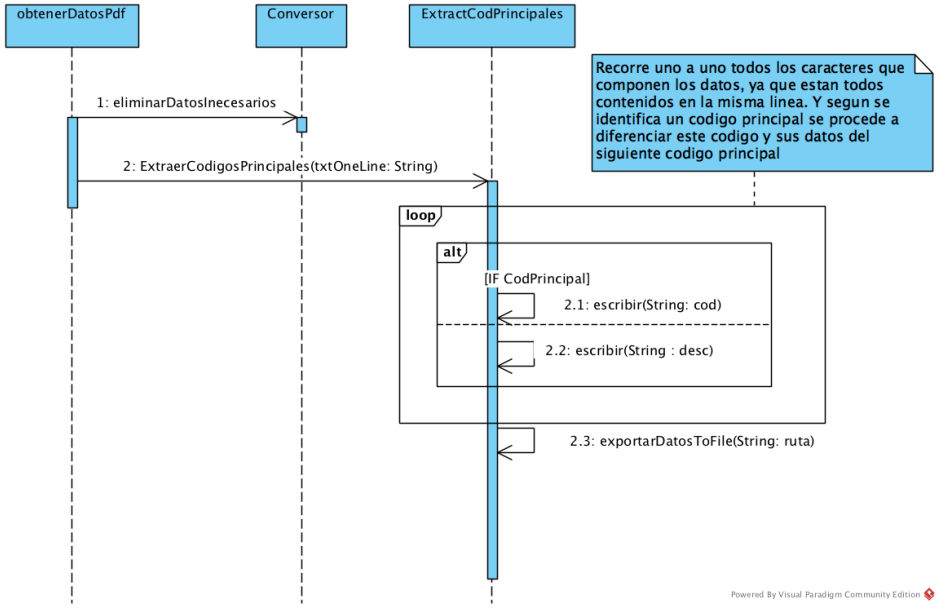


Figura B.2: Diagrama de secuencia para obtener los datos diferenciados entre Código Principal y Descripciones

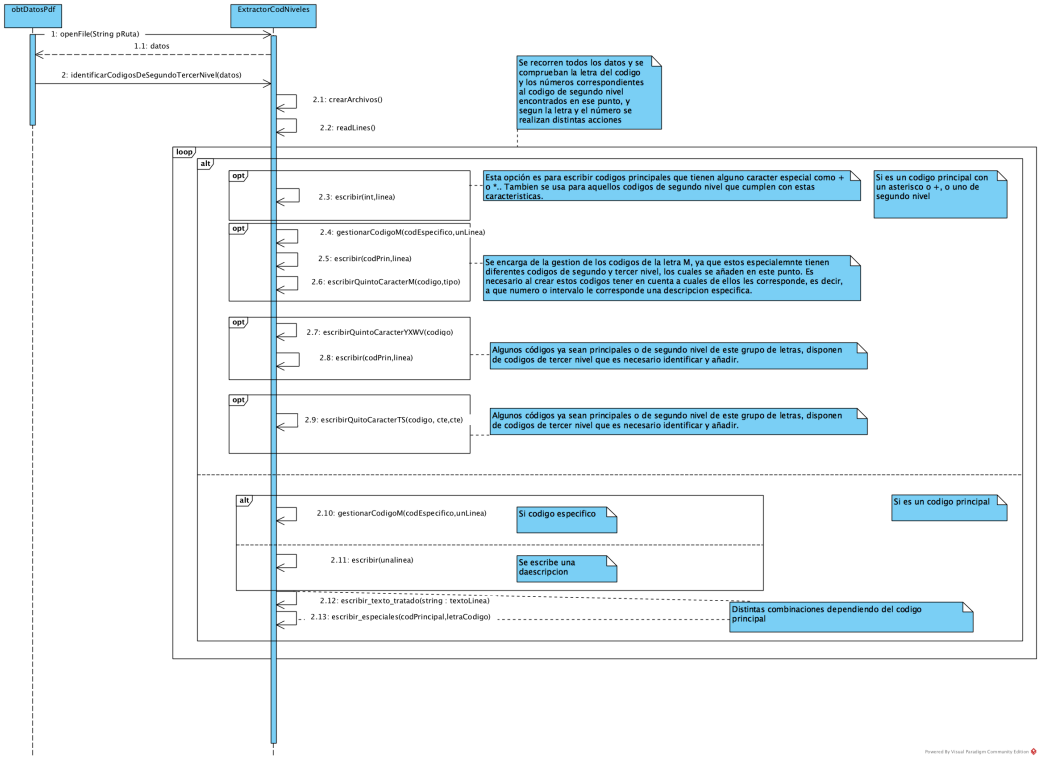


Figura B.3: Diagrama de secuencia para generar los códigos segundo o tercer nivel

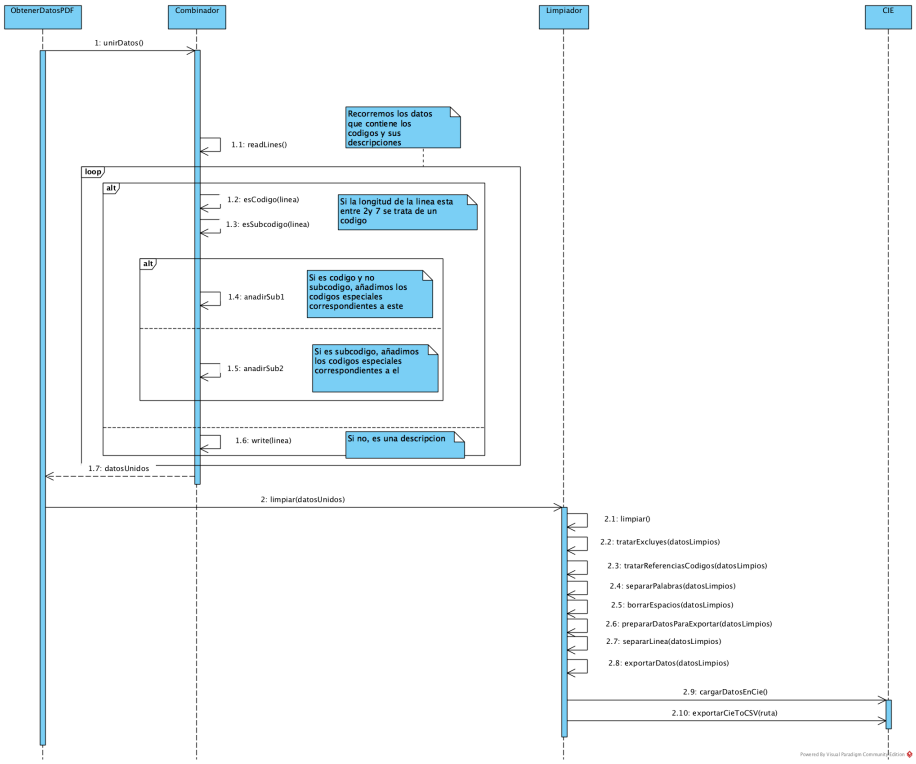


Figura B.4: Diagrama de secuencia para combinar los datos obtenidos del archivo, con los códigos generados manualmente

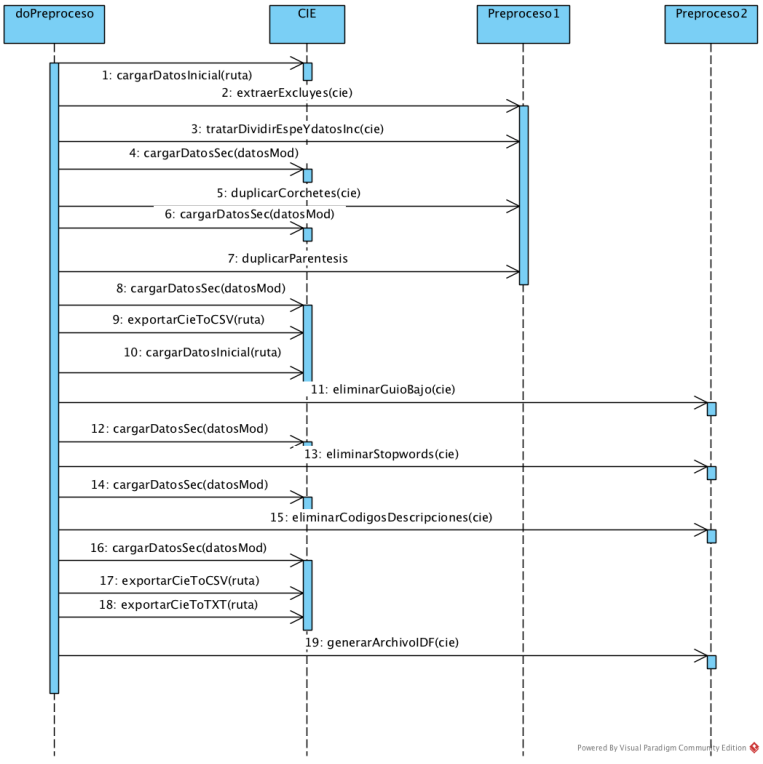


Figura B.5: Diagrama de secuencia para preprocesar los datos

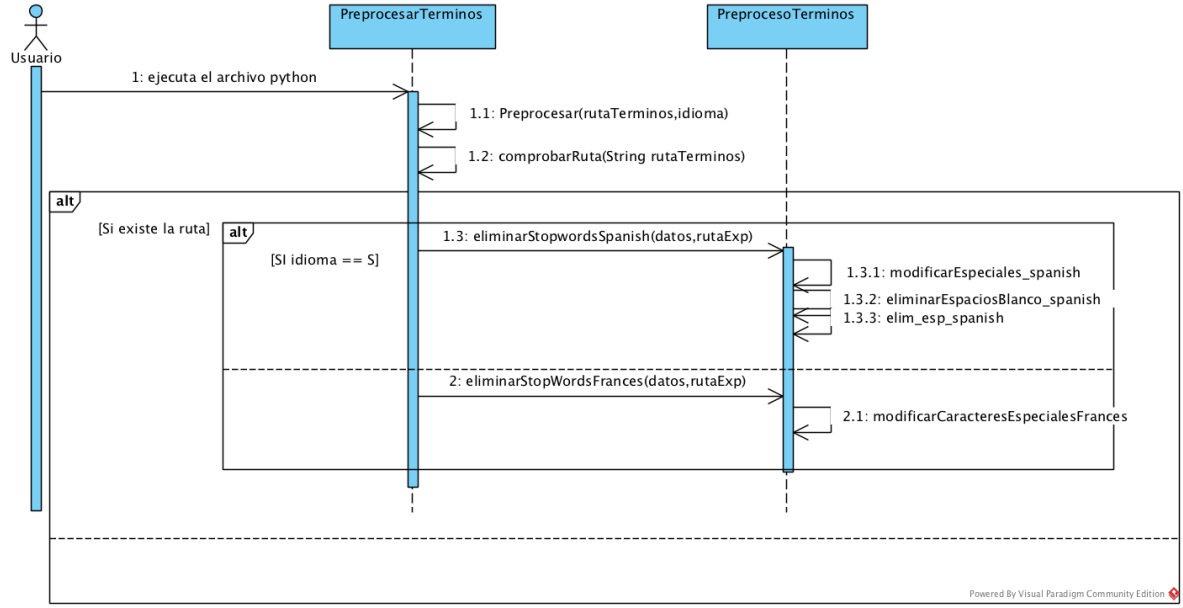


Figura B.6: Diagrama de secuencia preprocesar etiquetas

B.2. Prototipo 2: Obtención de similitudes Mediante DKPro Similarity Lexical

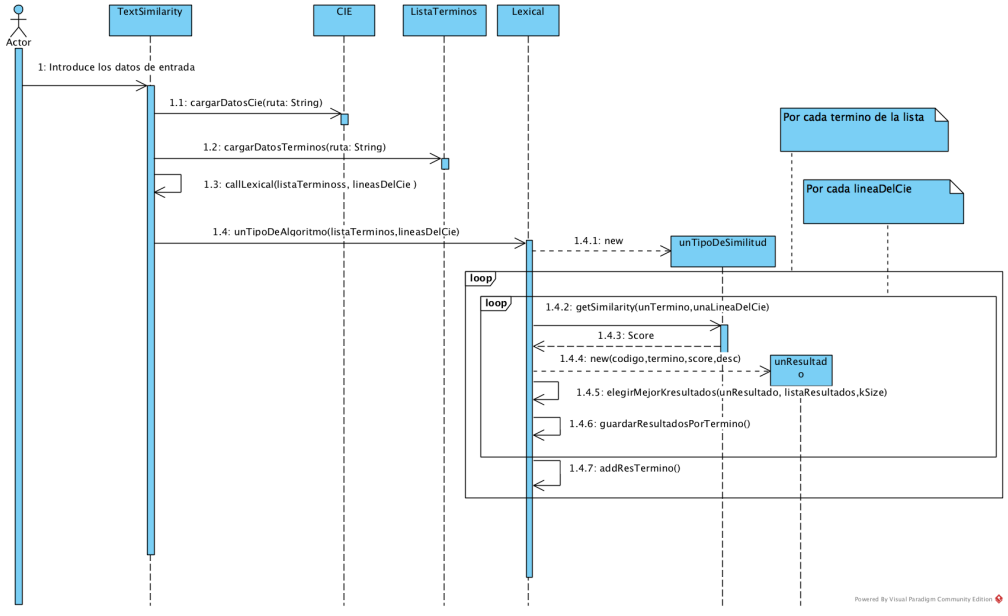


Figura B.7: Diagrama de secuencia de Lexical, forma genérico

El diagrama anterior representa de forma general de aplicar las distintas medidas de similitud utilizadas del modulo lexical. En su uso, varían el tipo de “TextSimilarityMeasure”. Por ejemplo, para obtener el valor de similitud de *Levenshtein* se utiliza el tipo “LevenshteinComparator” y la de *Dice Coefficient* con “DiceSimMetricComparator”.

B.3. Prototipo 3: Obtención de similitud Mediante DKPro Similarity LSA

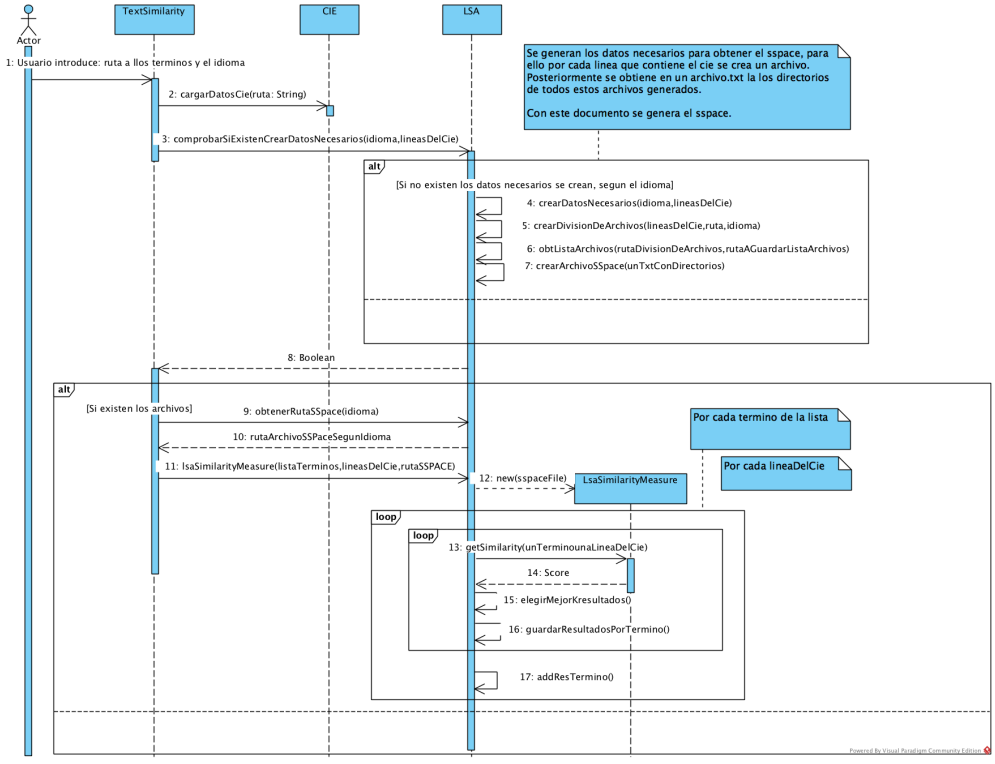


Figura B.8: Diagrama de secuencia obtencion similitud LSA

B.4. Prototipo 4: Exporta y evaluar resultados

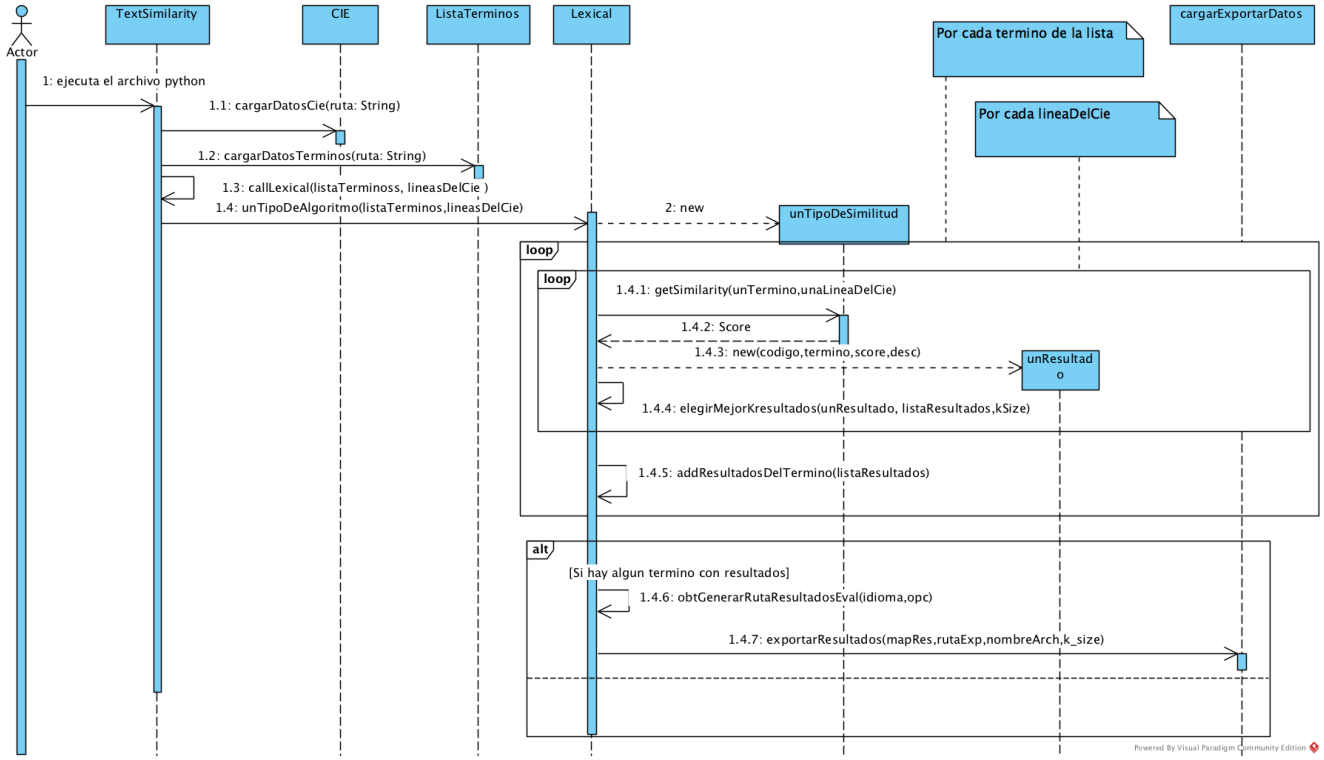


Figura B.9: Diagrama de secuencia con funcionalidad de exportar datos

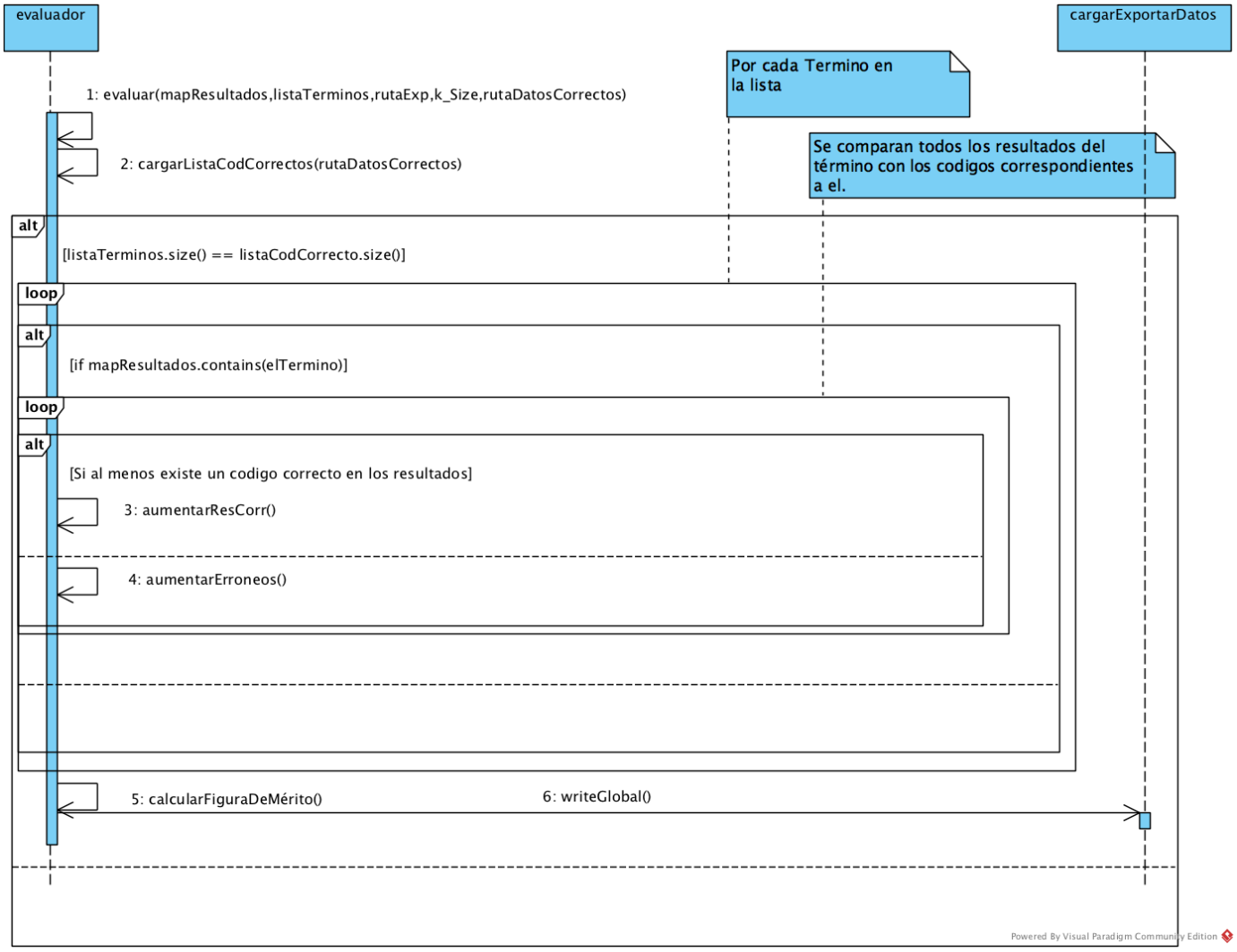


Figura B.10: Diagrama de secuencia método evaluar

Anexos C

Manual de usuario

A continuación, se procede a explicar como hacer uso del software desarrollado en este proyecto, así como, los requisitos que ha de cumplir el sistema para poder ejecutarlos.

Para poder ejecutar este software principalmente se hace uso de la terminal. No obstante, en el apartado correspondiente al cálculo de similitudes y etiquetado términos, también se ha diseñado una interfaz de usuario para aplicar las herramientas de una manera mas intuitiva, el único inconveniente de la interfaz, es hacer uso de ella a través de un servidor remoto, ya que interactuar con ella de esta manera puede ser lento y tedioso.

Todo lo necesario para poder hacer uso del software, es decir, los archivos de Python, los ejecutables ".jar" y los datos, se encuentran disponibles en la carpeta con el nombre TFG-Unai.

Antes de proceder a la explicación de la ejecución del software, el sistema en el que se vaya a ejecutar debe cumplir ciertos requisitos. A continuación, se detallan estos requisitos y si es necesario, como realizar la instalación de las distintas herramientas o módulos requeridos:

- Sistema operativo unix: Ubuntu, Linux, MacOSX.
- Python3: sudo apt-get install Python3
- pip3: sudo apt-get install python3-pip
- pdfminer: sudo pip3 install pdfminer.six
- PyPDF2: sudo pip3 install PyPDF2
- re: sudo pip3 install re
- ntlk: sudo pip3 install ntlk o sudo pip3 install -U ntlk, elegir una opción u otra.
- Descargar los datos de ntlk, para que no de error a la hora de eliminar las *stop words* introduciendo los siguientes comandos en la terminal pasos:
  - python3
  - import nltk
  - nltk.download()

  Se mostrara una interfaz, pulsamos *Download* y se procederá con la descarga los datos necesarios.
- Java, si no se encuentra instalado en la maquina se puede descargar e instalar a traves del siguiente enlace: (clic aquí).
- Disponer de la utilidad por línea de comando SVD, para realizar su instalación seguir los siguientes pasos:

- "Doug Rohde's SVD C library", la carpeta con el contenido para su instalación se adjunta en la carpeta TFG-Unai con el nombre svdlibc-master, aun así, se puede descargar en (clic aquí).
- Una vez descargada o localizada la carpeta svdlibc-master, descomprimirla y acceder a ella mediante la terminal con el comando: cd svdlibc-master. Una vez se accede a la carpeta, introducir el comando *make* y posteriormente *make install*. Si todo se ha realizado correctamente, se instalará en la maquina. Para mas información de como realizar la instalación, en la pagina web indicada previamente, se encuentra este proceso en mas detalle.

Una vez se cumplen los requisitos previamente mencionados, se procede a detallar como realizar la ejecución del software.

C.1. Obtención Datos CIE-10

Para obtener los datos del documento CIE-10 y de esta manera crear el "diccionario", hay que hacer uso del archivo Python *CrearDiccionario.py*. Este archivo se encuentra dentro de la carpeta TFG-Unai en el directorio "CrearDiccionario", así como, los datos que este archivo utiliza.

Este archivo realiza los siguientes procesos:

- Extrae los datos del documento *Volume1.pdf* que contiene los datos de la Clasificación Internacional de Enfermedades.
- Realizar la corrección de los caracteres erróneamente codificados en su extracción.
- Identifica los códigos principales y los datos de estos.
- Identifica los códigos de segundo nivel que se encuentran entre los datos de los principales, y crea los códigos de tercer nivel.
- Se realiza una limpieza de los datos, seleccionando aquellos que aportan información.
- Exportar los datos a formato *.csv*, creando el "diccionario" inicial.

Objetivo. Extraer los datos del documento *pdf* original, identificando los distintos datos que contiene y crear el archivo con los datos que formaran parte del "diccionario" en formato *csv*.

Parámetros: no requiere parámetros, los datos de entrada y de salida se indican internamente.

Salida: Este archivo crea una carpeta con el nombre *TxtToCsv* y en esta, se almacenan los resultados que se van generando a lo largo de su ejecución, facilitando de este modo, obtener los datos de los distintos procesos. En esta carpeta como resultado final se crea el "diccionario" con el nombre *CIE10_RAW.csv*.
Ejecución:

```
python3 /rutaAl/ObtenerDatosPDF.py
```

En el momento de exportar los archivos creados, si no existe la carpeta destino, esta se crea. En el caso de que esta ya exista, si esta contiene archivo y estos comparten el nombre con los que se van a generar, son sustituidos por los nuevos.

C.2. Preprocesar Datos

Como se hace uso de distintos datos, por un lado los "diccionarios" en castellano y francés, y por el otro los términos no-estándar, se requiere de dos tipos de preproceso.

C.2.1. Preprocesar Diccionario en Castellano

En este apartado, se hace uso del archivo Python *DoPreprocesoCast.py*, el archivo se encuentra en la carpeta TFG-Unai en el directorio "PreprocesarCIECast".

Este archivo realiza los siguientes procesos:

- Elimina los *excluye* y por tanto su información.
- Realiza el duplicado de aquellos datos que contienen corchetes y paréntesis.
- Eliminación de *stop words*
- Crea el archivo con el que posteriormente se generan los *idf values*.
- Exporta el diccionario preprocesado en formato *csv*.

Objetivo: preprocesar el contenido del "diccionario" con los datos en castellano.

Parámetros: no requiere, los datos de entrada y salida indican internamente.

Salida: Este archivo crea tres carpetas en las que exportara los datos:
- DatosPreproceso: en esta carpeta se almacenan archivos que contienen los datos en procesos intermedios previos a la exportación del archivo final.
- CIEFINAL: se exporta el "diccionario" ya preprocesado, tanto en formato de texto como en csv, con el nombre *cie10E.csv* y *cie10E.txt*.
- GenerarIDF: Se crea el archivo *PARA_IDF_ICD10E.txt*.

Ejecución:

```
python3 /rutaAl/DoPreprocesoCast.py
```

Si el directorio donde se guardan los datos contiene archivos con el mismo nombre, estos serán sustituidos por los nuevos.

C.2.2. Preprocesar Diccionario Francés

En este apartado, se hace uso del archivo Python *DoPreprocesoFrench.py*, el archivo se encuentra en la carpeta TFG-Unai en el directorio “PreprocesarCIEFrances”.

Este archivo realiza los siguientes procesos:

- Se unifica un archivo.csv que contiene los datos del CIE-10 francés descargado de Internet y los del conjunto de datos francés.
- Elimina los *stop words*
- Crea el archivo con el que posteriormente se generan los *idf values*.
- Exporta el diccionario preprocesado en formato *csv*.

Objetivo: preprocesar el contenido del “diccionario” con los datos en francés.

Parámetros: no requiere, los datos de entrada y salida se indican internamente.

Salida: Este archivo genera los siguientes documentos de salida:
- El “diccionario” ya preprocesado, en formato csv, con el nombre *cieF.csv*.
- El archivo, *PARA_IDF_ICD10F.txt*.

Ejecución:

```
python3 /rutaAl/DoPreprocesoFrench.py
```

Si el directorio donde se guardan los datos contiene archivos con el mismo nombre, estos serán sustituidos por los nuevos.

C.2.3. Preprocesar Términos

En este apartado, se hace uso del archivo Python *PreprocesarTerminos.py*, el archivo se encuentra en la carpeta TFG-Unai en el directorio “PreprocesoTerminos”.

Este archivo realiza los siguientes procesos:

- Preprocesa los datos de los términos no-estándar que quieren ser etiquetados.

Parámetros:

- Parámetro 1: Ruta del archivo que contiene los términos.
- Parámetro 2: Idioma de los términos, S si en castellano o F si en francés.

Salida: Este archivo genera un documento de salida, asignándole al nombre del archivo a preprocesar la palabra "PreProcesado", con los términos preprocesados. Ej.: si el archivo que se desea preprocesar se denomina *prueba.txt* , el archivo que se creara se denominara *PreProcesadoprueba.txt*.

Ejecución:

```
python3 /rutaAl/PreprocesarEtiquetas.py /tu/ruta/alarchivo.txt S
```

Nota
A los términos no-estándar del francés se les realiza un preproceso más, de la siguiente forma:

Limpieza datos francés

Para realizar este proceso, se ha de tener en cuenta, que el conjunto de dato en francés consta de tres archivos diferentes, el train, el dev y el test.

Al conjunto train se eliminaran ademas de los datos cuyo código consta como desconocido "UNK" y aquellos que disponen de mas de un código, los términos repetidos. En cambio, a los datos del dev y test, únicamente se les eliminan los desconocidos y aquellos que les corresponda mas de un código.

Los archivos encargados de realizar este proceso son los siguiente, y se encontrar junto a los datos de los que hacen uso, ya que estos datos son confidenciales, y no se pueden distribuir:

- *LimpiarTrain.py*
- *LimpiarDev.py*
- *LimpiarTest.py*

Parámetros:
- Ruta del archivo con los términos.
- Ruta del archivo con los códigos correspondientes a los términos.

Salida:
- Train: trainDesc.txt y trainICD.txt.
- Dev: DevDesc.txt y DevICD.txt.
- Test: TestDesc.txt y TestICD.txt.

Ejecución:
La ejecución de los tres archivos es similar, únicamente cambia el nombre del archivo Python y los parámetros:

```
python3 /rutaAl/LimpiarTrain.py /ruta/trainTerm.txt /ruta/trainICD.txt

    python3 /rutaAl/LimpiarDev.py /ruta/devTerm.txt /ruta/devICD.txt

    python3 /rutaAl/LimpiarTest.py /ruta/testTerm.txt /ruta/testICD.txt
```

C.3. DKPro Similarity Algorithms

En este apartado se explica como ejecutar el software correspondiente a los distintos algoritmos de la herramienta "DKPro Similarity" implementados en el proyecto.

C.3.1. Ejecutables sin interfaz

En este apartado, se han generado tres ejecutables distintos para hacer uso del software implementado.

Los ejecutables correspondientes a la evaluación, *TextSimilarityEvaluar.jar* y *TextSimilarityBestThreshold*, son los utilizados para realizar la evaluación del software programado. Se decide explicar su funcionamiento, en caso de que el usuario final quiera hacer uso de ellos.

Los "diccionarios" que se utilizan en este apartado no hay que introducirlos como parámetros, ya que se adjuntan con el trabajo y en el software se indica su ruta de manera interna.

TextSimilarityResultados.jar

Objetivo: etiquetar los términos no-estándar, según los parámetros de entrada que escoja.

Parámetros: los parámetros son los siguientes:

- Parámetro 1: Ruta del archivo que contiene los términos.
- Parámetro 2: Idioma de los términos, S si castellano o F si francés.
- Parámetro 3: Módulos de DKPro Similarity que se desea ejecutar: 1 si *lexical* o 2 si *LSA*.
- Parámetro 4: En el caso de elegir 1 (*lexical*), se podrá seleccionar entre una de las siguientes opciones indicando su número correspondiente, si se ha seleccionado 2 (*LSA*) es indiferente que valor indicar, el único requisito es introducir uno:
  - 0: Se ejecutan todas las posibilidades.
  - 1: BoundedSubstringMatchComparator
  - 2: ExactStringMatchComparator
  - 3: SubstringMatchComparator
  - 4: wordNgramJaccardMeasure
  - 5: wordNGramContainmentMesure
  - 6: characterNGramMeasure con N=3,4,5 y 6
  - 7: LongestCommonSubsequenceComparator
  - 8: LongestCommonSubsequenceNormComparator
  - 9: LongestCommonSubstringComparator
  - 10: Levenshtein
  - 11: LevenshteinSecondStringComparator
  - 12: JaroSecondStringComparator
  - 13: JaroWinklerSecondStringComparator
  - 14: MongeElkanSecondStringComparator
  - 15: tokenPairDistanceMeasure
  - 16: DiceSimMetricComparator
  - 17: OverlapCoefficientSimMetricComparator

    - 18: CosineSimMetricsComparator
- Parámetro 5: El número de k, es decir, la cantidad de resultados a obtener por cada término.

Salida: Como resultado se obtendrá el archivo o archivos en formato de texto de la opción seleccionada. Si se ha seleccionado el valor 0 (Se ejecutan todas las posibilidades)y, por tanto,se generara un archivo con el nombre correspondiente a cada una de ellas,así como, con el valor de k indicado: por ejemplo si se escoge la opción 4 con k=5, el archivo de salida seria *wordNgramJaccardK5.txt*.

Ejemplo de ejecución *lexical*:

```
java -jar /rutaAl/TextSimilarityResultados.jar /tu/ruta/archivoTerm.txt S 1 4 5
```

Ejemplo de ejecución *LSA*: en esta opción el parámetro 4, es indiferente, el único requisito es introducir un valor.

```
java -jar /rutaAl/TextSimilarityResultados.jar /tu/ruta/archivoTerm.txt S 2 4 5
```

Nota

Cuando se generan los resultados, estos se almacenan en la carpeta de resultados que se encuentra dentro de */TFG-Unai/DKProSimilarity/Lexical/idioma/Resultados/* o */TFG-Unai/DKProSimilarity/LSA/idioma/Resultados/*. Si en esta carpeta existe un archivo con el mismo nombre, los datos del archivo previo se sobrescribirán con los nuevos.

TextSimilarityEvaluar.jar

Objetivo: el objetivo de este ejecutable, es permitir al usuario ademas de obtener los resultados según los parámetros que elija, evaluar los resultados obtenidos con el algoritmo o algoritmos utilizados.

Parámetros: los parámetros son los siguientes:

- Parámetro 1: Ruta del archivo que contiene los términos.
- Parámetro 2: Idioma de los términos, S si castellano o F si francés.
- Parámetro 3: Módulos de DKPro Similarity que se desea ejecutar: 1 si *lexical* o 2 si (*LSA*).
- Parámetro 4: En el caso de elegir 1 (*lexical*), se podrá seleccionar entre una de las siguientes opciones indicando el número, si se ha seleccionado 2 (*LSA*) es indiferente que valor indicar, el único requisito es introducir uno:
    - 0: Se ejecutan todas las posibilidades.
    - 1: BoundedSubstringMatchComparator

- 2: ExactStringMatchComparator
- 3: SubstringMatchComparator
- 4: wordNgramJaccardMeasure
- 5: wordNGramContainmentMesure
- 6: characterNGramMeasure con N=3,4,5 y 6
- 7: LongestCommonSubsequenceComparator
- 8: LongestCommonSubsequenceNormComparator
- 9: LongestCommonSubstringComparator
- 10: Levenshtein
- 11: LevenshteinSecondStringComparator
- 12: JaroSecondStringComparator
- 13: JaroWinklerSecondStringComparator
- 14: MongeElkanSecondStringComparator
- 15: tokenPairDistanceMeasure
- 16: DiceSimMetricComparator
- 17: OverlapCoefficientSimMetricComparator
- 18: CosineSimMetricsComparator

- Parámetro 5: El número de k, es decir, la cantidad de resultados a obtener por cada término.

- Parámetro 6: ruta del archivo que contiene los códigos correctos correspondientes a los términos a etiquetar.

Salida: este ejecutable como salida, ademas de los resultados, también exporta los datos correspondientes a la evaluación segun *precission@k*.

Ejemplo de ejecución *lexical*:

```
java -jar /rutaAl/TextSimilarityEvaluar.jar /tu/ruta/archivoTerm.txt S 1 4 5
/tu/ruta/codTerm.txt
```

Ejemplo de ejecución *LSA*: en esta opción el parámetro 4, es indiferente el valor que se le indique, es requisito introducir un valor.

```
java -jar /rutaAl/TextSimilarityEvaluar.jar /tu/ruta/archivoTerm.txt S 2 4 5
/tu/ruta/codTerm.txt
```

Nota

Al igual que sucede con los archivos del ejecutable *TextSimilarityResultados.jar*, en este apartado, también se almacenan los archivos que contienen la información de los resultados y sus evaluaciones (estos se almacenan en al carpeta “Evaluación”). Por tanto, si en estas carpetas existe archivos que compartan el nombre con los nuevos, son sustituidos.

TextSimilarityBestThreshold.jar

Objetivo: este ejecutable permite realizar la evaluación de los datos, según un valor mínimo previamente obtenido, que indica el valor de similitud mínimo para que los resultados sean validos.

Parámetros:

- Parámetro 1: Ruta al archivo train con los términos.
- Parámetro 2: Ruta al archivo train con los códigos.
- Parámetro 3: Ruta al archivo dev con los términos.
- Parámetro 4: Ruta al archivo dev con los códigos.
- Parámetro 5: Ruta al archivo test con los términos.
- Parámetro 6: Ruta al archivo test con los códigos.
- Parámetro 7: Idioma, S castellano o F francés.
- Parámetro 8: Opción a utilizar:
  - 1: character5GramMeasure
  - 2: character3GramMeasure
  - 3: wordNgramJaccardMeasure

Salida: los archivos de texto correspondientes a los resultados según el valor del *threshold* y el archivo con el cálculo de las figuras de mérito.

Ejemplo de ejecución: en estos momentos únicamente se disponen de datos necesarios para realizar este apartado en el idioma francés.

```
java -jar /rutaAl/TextSimilarityBestThreshold.jar /tu/ruta/trainTerms.txt
/tu/ruta/trainCodes.txt
/tu/ruta/devTerms.txt /tu/ruta/devCodes.txt
/tu/ruta/testTerms.txt /tu/ruta/testCodes.txt F 2
```

Nota

Al exportar los archivos si estos ya existen en el directorio en el que se almacenan sobrescribirán al ya existente con los datos nuevos, tal y como sucede con los otros ejecutables.

C.3.2. Ejecutable Interfaz Gráfica: TextSimilarityIU.jar

Objetivo: Facilitar al usuario una interfaz gráfica amigable para hacer uso de las mismas funciones que los ejecutables *TextSimilarityResultados.jar* y *TextSimilarityEvaluar.jar*.
Parámetros:

- Elegir entre *lexical* o *LSA*.
- Seleccionar el idioma.
- Seleccionar el algoritmo en caso del módulo *lexical*.
- Seleccionar el archivo con los términos.
- Seleccionar el valor de k.

- Si se desea evaluar.

Salida: Según las opciones elegidas se exportaran los archivos correspondientes, tanto los resultados o las evaluaciones.

Ejemplo de ejecución: se puede ejecutar directamente el archivo ".jar" haciendo doble clic sobre el, o mediante el uso de la terminal, utilizando el siguiente comando.

```
java -jar /rutaAl/TextSimilarityIU.jar
```

Al ejecutar el comando o hacer doble clic sobre el ejecutable, se abrirá la aplicación y se mostrará la siguiente pantalla:

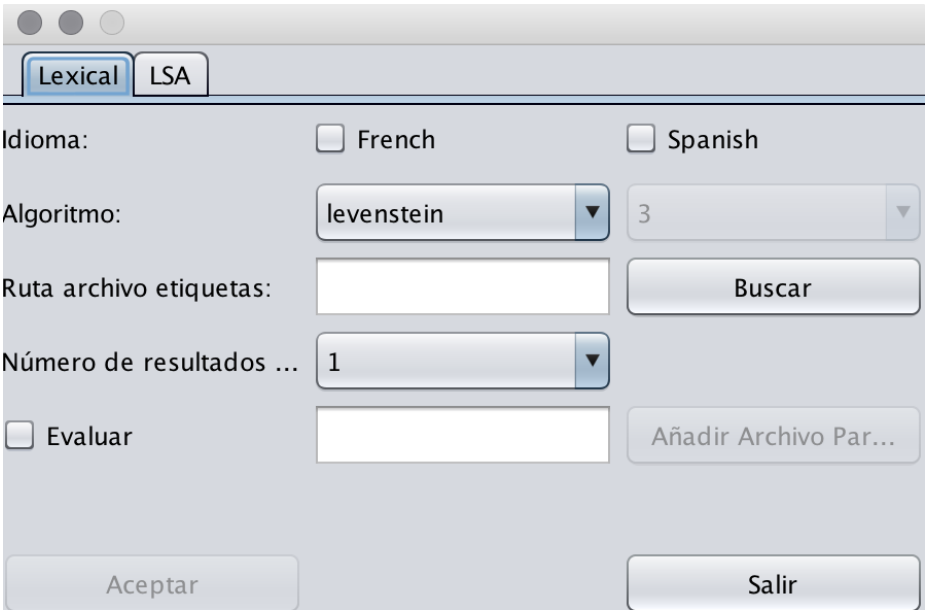


Figura C.1: Interfaz Usuario Lexical

Como podemos observar en la imagen, la pantalla consta de dos pestañas: una, en la que podremos seleccionar las distintas opciones del módulo *lexical* implementadas, y en la otra las opciones de *LSA*.

Como se ve en la imagen, para ejecutar la opción deseada la interfaz consta de diferentes elementos que pueden ser seleccionados. Sin seleccionar el idioma, el algoritmo y la ruta del archivo que contenga las etiquetas, no se activará el botón aceptar. También, es posible la elección del número de resultados a obtener.

Si se desea evaluar, será necesario seleccionar el *checkbox*, e indicar la ruta del archivo que contenga los códigos correctos correspondientes a cada término.

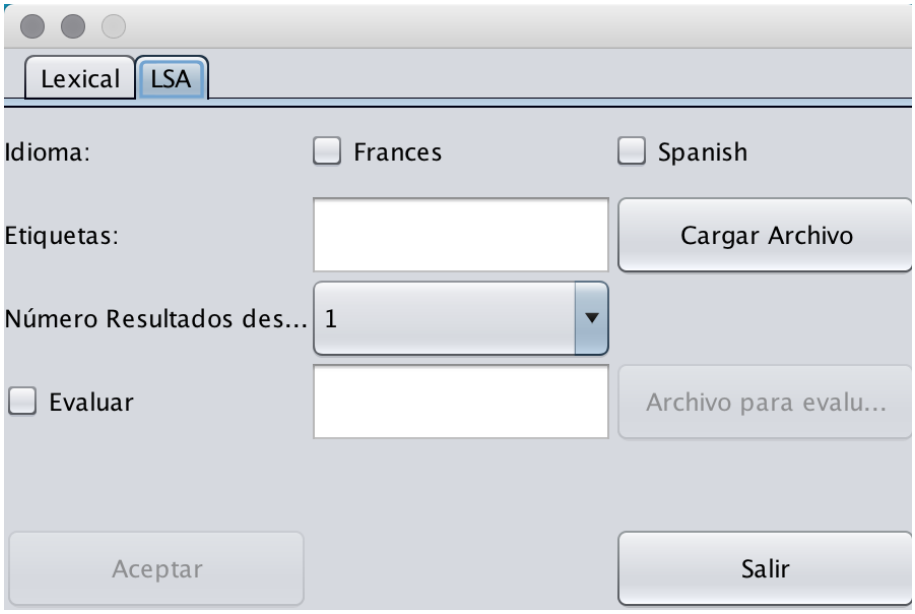


Figura C.2: Interfaz Usuario LSA

La interfaz correspondiente a la opción *LSA* funciona igual que la explicada previamente, con la diferencia de no tener que seleccionar un algoritmo.

Cuando se selecciona el botón de buscar o añadir archivo, se muestra el siguiente buscador para facilitar al usuario la busqueda y selección de los archivos:

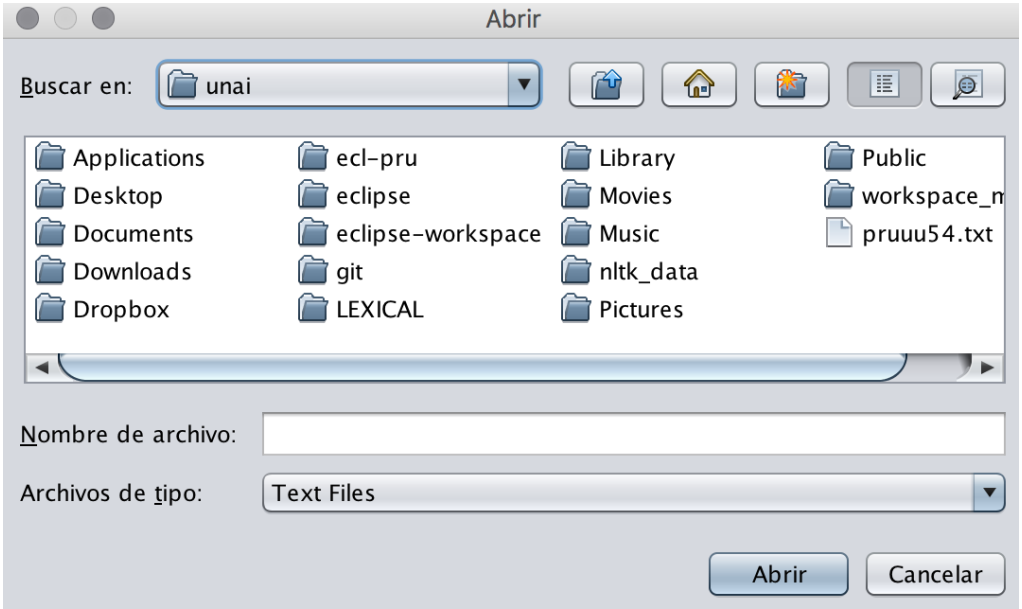


Figura C.3: Interfaz Usuario selección de documento

Cuando se pulse el botón aceptar, se procederá a obtener los resultados, estos serán exportados en el directorio del proyecto en las carpetas correspondientes al módulo seleccionado, *LSA* o *lexical*. Mientras se este ejecutando una opción, la pantalla sera la siguiente:

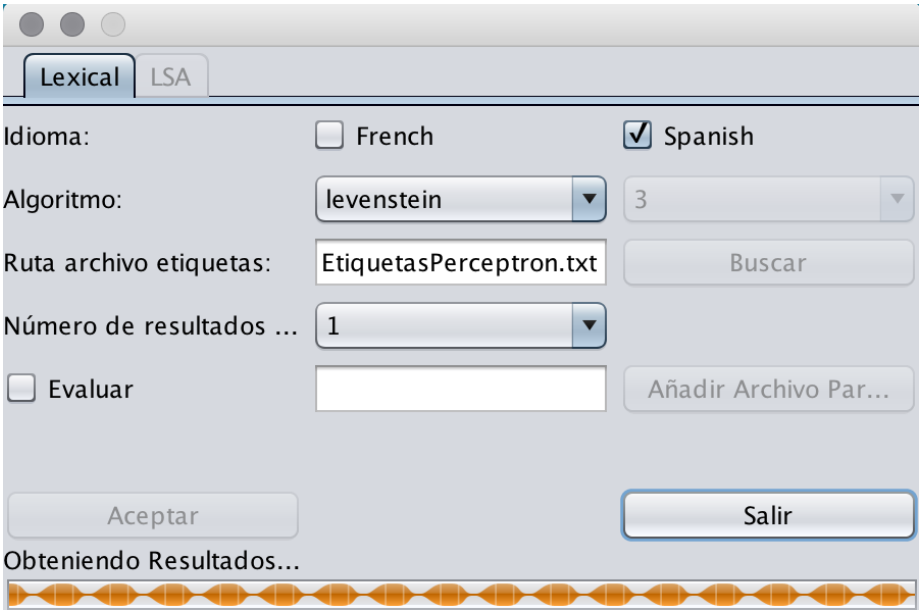


Figura C.4: Interfaz Usuario En ejecución

AVISO

Se recomienda que si en algún instante de la ejecución de LSA o la opción CharacterNGram de *lexical*, se suspende o se cierra la aplicación, acceder a las siguientes carpetas para eliminar posibles documentos que no se hayan generado correctamente y provoquen fallos en los resultados finales:

- Lexical CharacterNGram: En esta situación, acceder a la carpeta DKProSimilarity, y dentro de esta a Lexical (si se ha ejecutado mínimo una vez algún algoritmo), y en la carpeta idfvalues eliminar el archivo o archivos que se encuentren vacíos.
- LSA: Acceder a la carpeta DKProSimilarity y en esta eliminar la carpeta LSA.

Aun así, en la carpeta TFG-Unai, estarán todos los datos necesarios para realizar la ejecución. Salvo los datos en francés, los cuales al no ser públicos, únicamente se podrá hacer uso de ellos si se accede a un servidor remoto, en el cual, se encontraran disponibles el proyecto y todos los datos, incluidos los del francés.

Finalmente, tener en cuenta, que dependiendo de la cantidad de términos que se desean etiquetar y el idioma seleccionado, puede variar el tiempo de ejecución del proyecto. Por ejemplo, con los datos que se proporcionan para ejecutar el software, los datos en castellano no requieren de mucho tiempo para su obtención, en cambio, los del francés pueden demorarse horas.

Anexos D

Bibliografía

[1] Clef eHealth 2018. Task multilingual information extraction - icd-10 coding.

[2] Daniel Bär, Torsten Zesch, and Iryna Gurevych. Dkpro similarity: An open source framework for text similarity. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 121–126, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[3] Marc Damashek. Gauging similarity with n-grams: Language-independent categorization of text. *Science, New Series*, Vol. 267, No. 5199:pp. 843–848, Feb. 10, 1995.

[4] Thabet Slimani. Description and evaluation of semantic similarity measures approaches. *Computer Science Department Taif University and LARODEC Lab*, pages 1–10, 2013.

[5] M.K.Vijaymeena and K.Kavitha. A survey on similarity measures in text mining. *An International Journal (MLAIJ)*, 3:19–22, 2016.

[6] Wael H. Gomaa and Aly A. Fahmy. A survey of text similarity approaches. *International Journal of Computer Applications*, Volume 68– No.13:13–15, 2013.

[7] Alberto Barrón-Cedeño. Plagiarism detection across distant language pairs. Proceedings of the 23rd International Conference on Computational Linguistics:37–45, 2010.

[8] Peter Wiemer-Hastings. Latent semantic analysis.

[9] Jan A Kors, Simon Clematide, Saber A Akhondi, Erik M van Mulligen, and Dietrich Rebholz-Schuhmann. A multilingual gold-standard corpus for biomedical concept recognition: the mantra gsc. *Journal of the American Medical Informatics Association*, 22(5):948–956, 2015.

[10] A. Severyn and A. Moschitti. Learning to rank short text pairs with convolutional deep neural networks. 2015.

[11] Felix Naumann. Similarity measures.

[12] ml4a. https://ml4a.github.io/ml4a/es/neural_networks/, Redes Neuronales.