UNIVERSIDAD DE CONCEPCIÓN
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

CODIFICACIÓN AUTOMÁTICA DE EFECTOS ADVERSOS PRESENTES EN TEXTO LIBRE

POR

Bastián Mercado Ocares

Memoria de Titulo presentada a la Facultad de Ingeniería de la Universidad de Concepción
para optar al título profesional de Ingeniero Civil Biomédico

Profesor(es) Guía
Rosa Figueroa Iturrieta

Profesional Supervisor
Jaime Jiménez Ruiz

Julio 2022
Concepción (Chile)

Resumen

Con las nuevas tecnologías aplicadas en el campo de la medicina, se ha logado muchos avances en cuanto a entregar servicios de salud. La mayoría de los cambios están orientados por la inclusión de la informática, en especial en temas de información. Las fichas clínicas electrónicas están cada vez más presentes en el mundo con un sinfín de datos que pueden ser utilizados para apoyar la toma de decisiones con respecto a tratamientos de paciente o bien en la gestión de recursos, pero aún falta avanzar en estandarizar la información con el fin de facilitar su recuperación, extracción y análisis. Con la idea de apoyar los esfuerzos en estandarizar la información de fichas clínicas, se presenta este trabajo de memoria de título que desarrolla un sistema de recodificación del corpus clínico MADE 1.0 utilizando la terminología MedDRA. El uso de la terminología MedDRA permitirá obtener etiquetas más precisas, que permitan profundizar en los conceptos clínicos y con esto facilitar la extracción y recuperación de información. La recodificación se realiza utilizando la distancia de Levenshtein y con la implementación de 3 algoritmos de emparejamiento entre la información en MADE 1.0 y la terminología de MedDRA utilizando tanto términos preferidos como los términos asociados a éste.

Este algoritmo recodificador se presentó en 3 diferentes versiones, la primera es la más cruda de todas, aplicada de manera directa al corpus. La segunda ya con mejoras apuntando a la normalización, tanto del corpus como de la terminología. La última versión, tomaba lo hecho en la segunda versión, pero se mejoró con tratamientos especiales a palabras cortas y abreviaciones, además de integrar 2 niveles de búsqueda al utilizar la distancia máxima de Levenshtein (igual a “0” e igual a “1”). Con estas mejoras se logró cerca de un 60% de codificación de los efectos adversos presentes en MADE 1.0, dentro de estos un 97,43% fue clasificado de manera correcta, lo que se logró mantener desde el primer algoritmo. Como conclusión, se evidencia la potencia que posee la distancia de Levenshtein como codificador, acompañado de una adecuada terminología. Se trataron todos los objetivos planteados a lo largo de este trabajo, llegando a aborda tanto la terminología MedDRA como el corpus MADE 1.0, detallando sus características, como ambas afectaron en los resultados finales y como pueden ser mejoradas para futuros usos. Por otra parte, a pesar de las complicaciones antes mencionadas, el algoritmo final posee un porcentaje alto de términos bien clasificados (97,43%) en relación al total de términos que logro identificar en MADE 1.0, lo que nos permite ver que, a medida que el algoritmo se vaya mejorando, la positividad de clasificación esperada será similar o superior.

Abstract

With the new technologies applied in the field of medicine, many advances have been made in the delivery of health services. Most of the changes have been driven by the inclusion of information technology, especially in the area of information. Electronic medical records are increasingly present in the world with an endless amount of data that can be used to support decision making with respect to patient treatment or resource management, but there is still a lack of progress in standardizing the information in order to facilitate its retrieval, extraction and analysis. With the idea of supporting efforts to standardize the information in clinical records, this thesis develops a recoding system for the MADE 1.0 clinical corpus using the MedDRA terminology. The use of MedDRA terminology will allow more precise labels to be obtained, which will make it possible to go deeper into clinical concepts and thus facilitate the extraction and retrieval of information. The recoding is performed using the Levenshtein distance and with the implementation of 3 matching algorithms between the information in MADE 1.0 and the MedDRA terminology using both preferred terms and the terms associated with it.

This recoding algorithm was presented in 3 different versions, the first being the crudest of all, applied directly to the corpus. The second one, with improvements aimed at normalization, both of the corpus and of the terminology. The last version took what was done in the second version, but was improved with special treatments for short words and abbreviations, as well as integrating 2 search levels by using the maximum Levenshtein distance (equal to "0" and equal to "1"). With these improvements, about 60% of the adverse effects present in MADE 1.0 were coded, of which 97.43% were correctly classified, which was maintained from the first algorithm. In conclusion, the power of the Levenshtein distance as a coder, together with an adequate terminology, is evident. All the objectives set out throughout this work were addressed, and both the MedDRA terminology and the MADE 1.0 corpus were dealt with, detailing their characteristics, how both affected the final results and how they can be improved for future use. On the other hand, despite the complications mentioned above, the final algorithm has a high percentage of well-classified terms (97,43%) in relation to the total number of terms identified in MADE 1.0, which allows us to see that, as the algorithm is improved, the expected classification positivity will be similar or higher.

Tabla de Contenidos

Lista de Tablas

Lista de Figuras

Abreviaciones

Mayúsculas

ADE	:	Adverse Drug Event (efectos adversos de medicamentos).
CIE-9	:	Clasificación internacional de enfermedades – Novena edición
CIE-9-CM	:	Clasificación internacional de enfermedades – Novena edición, con modificaciones clínicas.
COSTART	:	Coding Symbols for a Thesaurus of Adverse Reaction Terms (Símbolos para codificar un tesauro de términos de reacciones adversas).
CUI	:	Concept unique identifier (Identificador de concepto único).
HLGT	:	high level group term (Grupo de términos de alto nivel).
HLT	:	high level term (término de alto nivel).
J-ART	:	Japanese Adverse Reaction Terminology (Terminología de reacción adversa de Japón).
LLT/ LT	:	Lowest Level Term (Término del nivel más bajo).
MedDRA	:	Medical Dictionary for Regulatory Activities (Diccionario médico para actividades reglamentarias).
MEDIS	:	Medical Information System Development Center (sistema japonés de información de salud).
PT	:	Preferred Term (Término preferente).
SOC	:	System Organ Class (clase de Sistemas del cuerpo humano).
SSLIF	:	Other Signs, Symptoms and Diseases (otros signos, síntomas y enfermedades).
WHO-ART	:	World Health Organization Adverse Reaction Terminology (Terminología de reacción adversa de la Organización mundial de la salud).
NLP	:	Natural Language Processing (procesamiento de lenguaje natural).

Capítulo 1. Introducción

1.1. Introducción General

La estandarización y estructuración de términos médicos es una necesidad presente desde hace muchos años. Como respuesta a esta necesidad, en 1990 surge MedDRA (*Medical Dictionary for Regulatory Activities*) como una terminología médica normalizada, específica a productos médicos, cuyo fin era apoyar la estandarización de los registros y su regulación [1]. El auge de esta terminología con 81.000 términos, estuvo marcado por la intención de estandarizar un diccionario medico compuesto de 5 niveles de clasificación jerárquica [2][3]. Estos niveles se caracterizan por tener una subordinación y una supra-ordinación, el termino más supra-ordenado será aquel de carácter más amplio y que a medida que se baja de nivel, se vuelven cada vez más específicos. Partiendo por el grupo más supra-ordenado tenemos al *System Organ Class* (SOC), indica el sistema del cuerpo humano al que se asocia el término. Luego sigue el *high level group term* (HLGT) que indica si pertenece a un síntoma, una condición, una enfermedad, etc. El siguiente es *high level term* (HLT) que especifica con un poco más de detalle a que hacer referencia el término. Como nivel subordinado se encuentra *preferred term* (PT) el termino más distintivo medicamente hablando, entrega con precisión el término identificado. Para terminar en la última clasificación que sería *lowest level term* (LLT), términos con mayor detalle que el nivel anterior, pero no son los más adecuado con los cuales referirse al término o, que solo hacen referencia al *preferred term* [4].

Esta terminología ha evolucionado con énfasis en el registro, recuperación, análisis y despliegue de datos. Aplica a todas las fases del desarrollo de productos como los medicamentos, así como también a sus efectos en la salud y los eventos adversos por malfuncionamiento de dispositivos. Dado lo anterior, MedDRA fue utilizado desde 2006 en la codificación de los ensayos clínicos sobre el VIH, con resultados favorables en términos de la factibilidad de la codificación, pero ya mostrando las dificultades generadas por la falta de precisión de los textos médicos estudiados y la alta especificidad y sensibilidad de MedDRA en ese entonces [5]. En 2009, se estudió la influencia de MedDRA en los resultados de la extracción de datos de farmacovigilancia, así como también, en el estudio clínico de “Riesgo similar de malignidad con la insulina glargina y la insulina Hagedorn de protamina neutra (NPH) en pacientes con diabetes de tipo 2” [6].

La codificación de información clínica no solo es importante en las labores de vigilancia de dispositivos clínicos o medicamentos, sino que también se convierte en una herramienta útil para estandarizar el registro de información de otras fuentes como lo son el registro clínico electrónico de los pacientes. En nuestro país cada vez se masifica más el uso de registro de la información clínica utilizando medios digitales, junto con ello emerge la necesidad de poder analizar la información registrada en favor de mejorar la calidad de la atención y la salud de los pacientes.

Por ejemplo, en el 2018 tenemos estudios vinculados al "impacto de la extensión del postnatal en la adherencia a la lactancia materna, estudio basado en la extracción de datos de fichas electrónicas en el Centro Médico San Joaquín entre el 2009 y 2013 [7]. Otro estudio realizado en Chile es el de "Caracterización de la enfermedad celiaca en niños atendidos en hospitales públicos chilenos", donde la utilización de datos extraídos de fichas electrónicas fue la base de la investigación, aportando resultados muy detallados, gracias a la gran cantidad de información presentes en estas fichas [8].

Ahora si bien estos estudios han demostrado que las fichas clínicas son una fuente importante de información, la mayoría de la información clave para su análisis, se encuentra no-estructurada o bien en un formato mixto mezclando campos estructurados y de texto libre. Aún queda por avanzar en la codificación de estos aspectos claves para poder hacer facilitar la extracción, recuperación y análisis para uso secundario con el propósito de entregar un mejor apoyo a las decisiones de gestión o clínicas [9].

Este trabajo propone utilizar un algoritmo basado en la distancia de Levenshtein como eje principal en la recodificación. Se harán diferentes pruebas, como por ejemplo analizar el aporte de la normalización del texto libre en este proceso o el de otras modificaciones hechas al algoritmo a medida que detectamos limitaciones dentro de la recodificación de MADE 1.0. Todo esto con el objetivo de aumentar la asertividad de la recodificación y maximizar el alcance de recodificación del algoritmo dentro de la información etiquetada en MADE 1.0.

1.2. Objetivos

1.2.1 Objetivo General

Desarrollar un algoritmo para la recodificación automática de efectos adversos presentes en el Corpus MADE.

1.2.2 Objetivos Específicos

- Pre-procesar el corpus de texto libre del cual se extraerá la información a recodificar.
- Analizar la terminología con la cual se realizará la codificación de los eventos adversos.
- Desarrollar algoritmo recodificador de eventos adversos utilizando herramientas del procesamiento natural del lenguaje y extracción de información.
- Evaluar y documentar el funcionamiento de los algoritmos al momento de recodificar afectos adversos en texto libre etiquetado.

1.3. Alcances y Limitaciones

El algoritmo a desarrollar se limitará a recodificar eventos adversos etiquetados en el corpus MADE, que contiene información de-identificada de registros clínicos electrónicos de pacientes con cáncer. Los eventos adversos a recodificar son los que han sido etiquetados en MADE como otros signos o síntomas y "ADE" (*adverse drug events*).

1.4. Temario

- **Capítulo 1:** Se introduce el tema a tratar definiendo el objetivo general y el específico junto con los alcances o limitaciones de este trabajo.
- **Capítulo 2:** Presenta el marco teórico en el que se sustenta este trabajo. Se entregan definiciones de conceptos relevantes, se presenta la revisión bibliográfica realizada, un análisis de software existentes y las bases de datos relacionadas al algoritmo a desarrollar.
- **Capítulo 3:** Presenta la metodología utilizada en este trabajo, centrándose en el algoritmo, detallando las 3 versiones realizadas y explicando su funcionalidad dentro de los efectos secundarios.
- ***Capítulo 4:*** Se presentan los resultados obtenidos y el respectivo análisis de limitaciones, para luego dar paso a las mejoras pertinentes dentro del algoritmo inicial y los nuevos resultados. Terminando con una versión más detallada del código, acompañado de su análisis y conclusiones.
- **Capítulo 5:** Se presenta una discusión del trabajo y el resumen de conclusiones.

Capítulo 2. Marco Teórico

2.1. Introducción

Este capítulo presenta la teoría que sustenta este trabajo, entregando una visión global más extensa del problema que se aborda. Se analiza cómo las necesidades inmediatas de estandarización dentro de cada país dieron paso a la búsqueda de un consenso en terminologías y las respectivas fuentes de datos que necesitan de dichas codificaciones para ser utilizadas dentro de la medicina como información masiva relevante. Además, se entregan nociones básicas del tratamiento de datos que se extiende a lo largo del informe, acompañadas de la descripción de las bases de datos a utilizar, sumándole las librerías y el software a utilizar.

2.2. Codificación

La codificación de eventos adversos constituye una temática fundamental que debe detectarse, prevenirse y vigilarse. Dado lo anterior, las organizaciones internacionales como, por ejemplo, la Organización Mundial de la Salud, han utilizado terminologías médicas para recodificar la información de eventos adversos facilitando así su procesamiento y análisis.

En Europa, se ha utilizado una combinación de la Terminología de Reacciones Adversas de la Organización Mundial de la Salud (WHO-ART) y la Novena Enmienda de la Clasificación Internacional de Enfermedades (CIE9). En los Estados Unidos, el término de la Administración de Alimentos y Medicamentos conocido como COSTART (símbolos de codificación para los términos y sinónimos de reacciones adversas) se usa comúnmente con la modificación clínica de CIE-9 (CIE-9-CM). Los japoneses han desarrollado sus propias versiones de estas terminologías, la terminología japonesa de reacciones adversas (J-ART) y el sistema japonés de información de salud (MEDIS). Además, muchas organizaciones han modificado estos términos para satisfacer sus necesidades. Los términos establecidos carecen de términos específicos que puedan profundizar los conceptos lo que proporciona opciones limitadas para la recuperación de datos (por ejemplo, muy pocos niveles en la jerarquía) y no nombran los síndromes de manera efectiva. Las organizaciones con buenos recursos han desarrollado su propia terminología interna para abordar algunas o todas las deficiencias [10].

Dentro de todas estas diferentes codificaciones las complicaciones son múltiples. La falta de estandarización en el uso de las terminologías genera que, por ejemplo, la industria tenga que adaptar sus productos a las codificaciones que se solicitan en cada país, lo que dificulta una interoperabilidad universal de esta información. Por ejemplo, la industria farmacéutica utiliza

codificaciones para representar la información de sus productos, pero para estas codificaciones varían dependiendo del lugar del mundo donde desean comercializar sus productos, lo que genera a esta industria dificultades en la identificación y aprobación de sus medicamentos por parte de las autoridades de los países en las que desean ingresar sus productos.

2.3. Terminología MedDRA

Esta Terminología busca la unificación de términos y la simplificación de la extracción y análisis de diferentes datos médicos, como los principios activos, efectos adversos, indicaciones médicas, entre otros. La terminología MedDRA abarca muchos conceptos médicos, de estos se destacan las categorías que han sido clasificadas como "médicos y relacionados a la salud", destacando [10]:

- Signos
- Síntomas
- Enfermedades
- Diagnósticos
- Indicaciones terapéuticas

Las categorías mencionadas están codificadas y relacionadas a sus diferentes términos utilizados.

Dentro de la terminología se tienen las siguientes definiciones [10]:

- **Término:** nombre que será asignado a su respectivo código e información anexa dentro de la terminología.
- **CUI:** o identificador de concepto único, código asignado a cada termino medico en este caso.
- **Nivel de termino:** esta clasificación se le asigna al termino con el fin de otorgarle una mayor relevancia dentro de un conjunto de sinónimos. Dentro de esta clasificación existen abreviaciones relevantes a definir:
    - **PT:** siglas en inglés de p*refered term*. Descriptor otorgado al término más adecuado a lo que hace referencia, ya sea signos, síntomas, enfermedades, diagnósticos o indicaciones terapéuticas. Es único dentro de cada CUI.

- **LLT (o LT):** siglas en inglés de *Lowest level term*. Descriptor otorgado a uno o más términos alusivos a un determinado PT. Este es el nivel más bajo de la terminología.

La primera versión de MedDRA contaba con códigos numéricos y símbolos de anteriores etimologías[1] en campos específicos, por lo que se relacionaba a diferentes codificaciones como el COSTART, WHO-ART, CIE9, y otros, pero a pesar de tener los términos ingresados de las termologías anteriores, no siempre poseen el mismo PT en MedDRA [10].

2.4. Corpus MADE 1.0

El Corpus de MADE 1.0, es una colección desidentificada de extractos de 1.089 notas del historial clínico de pacientes con cáncer del hospital "*University of Massachusetts"*. Este corpus contempla 2 secciones importantes, el registro electrónico de salud de cada paciente, y las anotaciones con etiquetas que identifican información y eventos de interés. Las etiquetas incluidas en la anotación son [11]:

- **Drug:** define a cualquier mención de medicamento, incluyendo los nombres nuevos y genéricos, además de los que se usan frecuentemente abreviados.
- **Indication:** se define como el síntoma que es la razón para la administración del medicamento.
- **Adverse Drug Event (ADE):** definido como un síntoma o signo resultado del uso de el o los medicamentos.
- **Other Signs, Symptoms and Diseases (SSLIF):** se define como cualquier otro señal o síntoma que no está directamente relacionado con ningún medicamento mencionado en la ficha.
- **Drug Frequency:** hace referencia al tiempo entre dosis, entrega la frecuencia con la se prescribe el medicamento o la cantidad sugerida de este.
- **Drug Dose, or Dosage:** definida como la cantidad de medicamento administrado por cada dosis única. Ejemplos: gramos, mililitros, gotas, etc.
- **Drug Duration:** se define como la duración total del medicamento.
- **Drug Route:** el método de administración como por ejemplo vía oral o intravenosa.
- **Severity:** define la extensión de la enfermedad o los síntomas que afectan al paciente.

[1] Etimología: Origen o procedencia de las palabras, que explica su significado y su forma.

Además de incluir relaciones entre los diferentes términos antes mencionados [11].

2.5. Pre-procesamiento de textos en lenguaje natural

El lenguaje natural resulta ciertamente complicado de procesar de manera directa, porque generalmente se sugiere un previo procesamiento de los datos con el fin de poder entregar variables más claras y relacionadas a lo que se busca obtener del texto libre. De esta manera se extienden las diferentes etapas del procesamiento del lenguaje natural como se ve a continuación en la figura 2.1 [12].

Figura 2.1Pasos de procesamiento de lenguaje natural. Fuente: ASAI [12].

- **Análisis morfológico:** Consiste en analizar las oraciones para identificar unidades léxicas, es decir busca palabras, signos de puntación e incluso tipos de palabras como sustantivos o verbos. La versatilidad de las palabras dentro de un determinado contexto puede ocasionar problemas de ambigüedad. En esta fase se le otorga una categoría gramatical a la palabra dentro de la oración a la que pertenece, lo que corresponde al etiquetado. Este último cumple con solucionar el problema de ambigüedad antes descrito.
- **Análisis sintáctico:** En este análisis se llega a establecer las relaciones de dependencia estructural entre las palabras antes etiquetadas, esta relación y orden deben aportar a su significado, esto quiere decir que pretende conservar la ubicación de cada palabra etiquetada y el cómo se vinculan unas con otras, para no perder la finalidad de la oración antes diseccionada.
- **Análisis semántico:** Se proyecta más allá de la palabra etiquetada, con el fin de poder direccionar lo más posible al significado de palabra dentro de la oración, ya que, este análisis se hace necesario cuando esta posee demasiados significados.
- **Análisis Pragmático:** Este análisis necesita ir más allá de la estructura del lenguaje, ya que involucra el contexto en el cual está inmerso el texto etiquetado para entregarle su significado.

No siempre se aplican todos los niveles para llegar a un procesamiento de lenguaje natural completo, ya que no es necesario abarcar análisis extensos cuando no son necesarios, frente a esto se hará una breve descripción del primero y más utilizado de sus análisis, el morfológico, y como se puede desarrollar dentro del texto a procesar. Dentro de sus diferentes maneras de enfocar este análisis se encuentra [12]:

- **Segmentación:** fragmenta el texto en secciones que se pueden tomar de manera independiente una de otra
- **Tokenización:** divide la oración a procesar en palabras, descartando espacios y puntos. Las unidades de texto resultantes se conocen como tokens.
- **Etiquetado gramatical:** Se le asigna una etiqueta o categoría gramatical a cada uno de los tokens de una oración en función del rol que cumplan en esta.
- **Reconocimiento de entidades nombradas:** Como su nombre lo dice se reconocen unidades predefinidas y se extraen del texto, como son los nombres propios, locaciones, fechas, etc.

Estos ayudan a entregar un texto segmentado de mayor facilidad de uso en algún proyecto o para continuar con algún análisis más elaborado antes descrito.

Posterior a esta fase de pre-procesamiento inicial de datos, dentro de este trabajo necesitamos encontrar alguna manera viable de poder relacionar el texto de lenguaje natural con la codificación antes expuesta, frente a esta necesidad la Distancia de Edición o Distancia de Levenshtein se vuelve uno de los caminos a tomar [13].

La Distancia de Levenshtein es el mínimo conjunto de operaciones necesarias para que las comparaciones de dos cadenas de textos sean necesarias para transformar la una en la otra, mediante la eliminación, inserción, sustitución o transportación de un carácter. Este concepto aparece en los años 60 con Levenshtein luego siendo modificado por Needleman y Wunsch en 1970 con el objetivo de otorgar un mayor o menor peso al error entre ambos caracteres dependiendo de qué tan frecuente sea, dando mayor asertividad en el modelo. Este presenta algunas complicaciones asociadas a la cantidad de caracteres al ser un modelo absoluto y no relativo. Esto último fue solucionado generando técnicas de normalización sencillas que satisfacen la desigualdad triangular, que consisten en normalizar dividiendo por la longitud de la cadena más larga o por la suma de la longitud de ambas cadenas [13].

2.5.1 Symspellpy

Librería basada en la Distancia de Levenshtein enfocado en la corrección ortográfica y búsqueda difusa destacada dentro de Python que permite encontrar mediante un código de eliminación simétrica, donde solo se necesitan eliminaciones para poder dar con la comparación correcta, desde el diccionario utilizado, la palabra que se busca o que con la cual se debe corregir. Destaca por su velocidad, ya que, mediante al solo requerir eliminación puede procesar más rápido los candidatos y cálculos previos al análisis. Un ejemplo señalado en su página oficial indica que "una palabra media de 5 letras tiene unos 3 millones de posibles errores ortográficos dentro de una distancia de edición máxima de 3, pero esta librería solo necesita generar 25 borrados para cubrirlos todos", dando a conocer el poder de optimización de este sistema [14].

Se tiene especial énfasis en *Symspellpy* no como corrector de ortografía, sino como comparador de caracteres basado en un diccionario, el cual es personalizable y capaz de ser creado por quien vaya a utilizar esta librería. Frente a esta versatilidad es una gran herramienta para la identificación de texto en lenguaje natural asociándolo a una base de datos previamente codificada, o con lo que será utilizada en este trabajo, una terminología médica [15].

2.6. Software a Utilizar

2.6.1 Anaconda (Spyder)

Debido al análisis realizado por TIOBE con relación al número de resultados en distintos motores de búsqueda, Python, lenguaje de programación de código abierto, es el primero estando por sobre C y Java, lenguajes muy populares y que gozaban de una alta popularidad estando mayoritariamente en el top 3 de lenguajes más utilizados [16]. Debido a esto el lenguaje de programación que se utilizará será Python 3, pero esto debe ir acompañado de un software en donde se pueda desarrollar y compilar el código, aquí es donde aparece Anaconda. Anaconda es un software que utiliza Python (o R) con variadas librerías y paquetes de código abierto, permitiendo desarrollar códigos con facilidad. Se encuentra disponible tanto para Windows, Mac OS y Linux [17].

Posee varios ambientes de trabajo, pero el que se utilizará será Spyder, este *subsoftware* permite desarrollar el código deseado sin mayores complicaciones. Este posee diferentes secciones visibles en la figura 2.2:

1. Una terminal de Python de fácil uso donde se visualiza los resultados en la terminal y posibles errores del código, contando además con un historial de comandos.
2. Posee un explorador de variables muy útil para la visualización de estas y revisarlas una a una sin necesidad de estar obligado a imprimirlas en la terminal, además de un explorador de archivos.
3. Un editor de texto para modificar los archivos .py en este caso.
4. Las diferentes opciones con las cuales se puede compilar el código que se está editando.

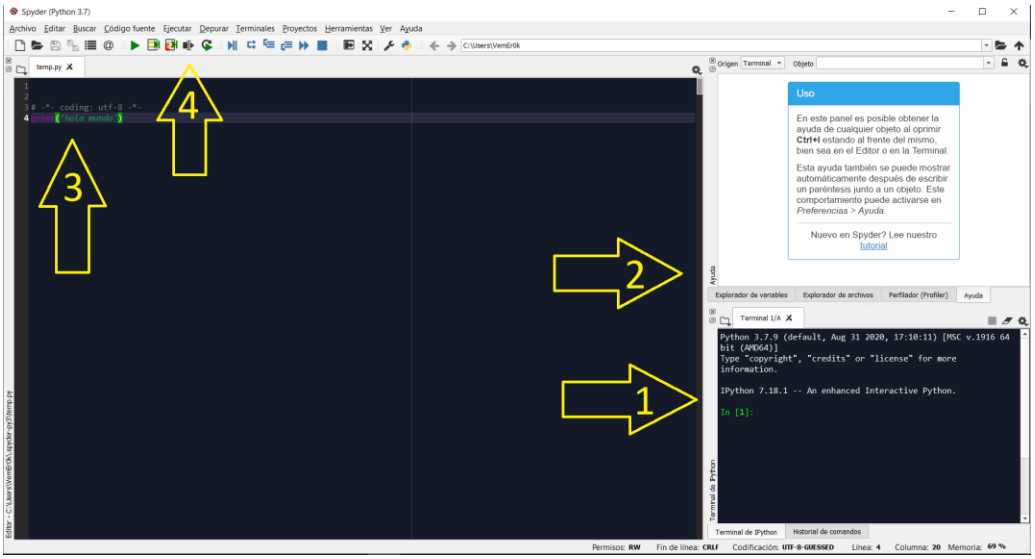

Figura 2.2 Interfaz de Spyder.

Además, está relacionado con "Anaconda Prompt" el cual te permite actualizar, instalar y eliminar librerías de tu computador.

2.6.2 Bibliotecas relacionadas

Se usarán diferentes librerías para poder desarrollar el código necesario para este proyecto:

A. Pandas

Librería de código abierto que apunta al uso sencillo de bases de datos incluyendo herramientas de análisis, reconoce un sinfín de archivos permitiendo la lectura de estos e identificándolos en Python como un *Dataframe* con el cual se puede trabajar [18].

B. Numpy

Esta librería permite el uso de múltiples objetos, trabajo de matrices, archivos y diferentes operaciones matemáticas, lógicas, transformadas, algebra lineal, entre otras, además de la

creación de archivos desde listas, matrices o tablas creadas dentro de Python, permitiendo así guardar resultados de manera sencilla [19].

C. Lxml

Utilizada para la lectura, manipulación y extracción de archivos en xml, originaria de las bibliotecas libxml2 y libxslt, toma la velocidad y la funcionalidad de estas librerías, y los une con la simplicidad de Python [20].

D. Matplotlib

Herramienta que permite disponer de múltiples gráficos y manipulación de estos, teniendo así gráficos personalizables y capaces de exponer con mayor claridad los datos, imágenes o cualquier tipo de información que pueda ser visualizada, además de contar con varios gráficos estadísticos una mayor profundidad de análisis a los datos manipulados con esta librería [21].

2.7. Trabajos Previos

Los siguientes trabajos han realizado codificaciones a modo de lograr aportar profundidad a la información médica o buscan entregar resultados relevantes en relación a los diferentes sistemas de codificación en cuanto a su asertividad y/o dificultades que enfrentan al codificar.

Se han realizado análisis de textos en otros idiomas y han sido comparados con el idioma inglés, predominante en la estructuración de datos. En 2014 se entrenó una inteligencia artificial con el fin de poder apreciar que tanto varia la efectividad de esta herramienta en el idioma sueco en comparación con el idioma inglés. Se percataron que dentro de los resultados la clasificación de la inteligencia artificial fue levemente menor en el idioma sueco, en comparación con los resultados obtenidos en el idioma inglés. Esto lleva a pensar que los enfoques utilizados para el inglés también son aplicados en el texto clínico sueco, pero una pequeña parte de los errores en el modelo son menos recurrentes en las fichas en inglés, lo que abre la posibilidad a mejoras mediante la adaptación del modelo al sistema clínico sueco [22]. Esto nos permite identificar que avances en un idioma determinado puede ser extrapolado a otros idiomas sin tener mayores diferencias que obliguen a cambiar rotundamente los modelos utilizados.

Otro estudio demuestra la importancia de algoritmos de codificación. En Andalucía, España, se utilizó un codificador automático para los diagnósticos clínicos. Se definieron 80 enfermedades crónicas a las cuales hacerle seguimiento con este sistema. El sistema automático contaba con 3

niveles: datos, aplicaciones y presentación, que se componía de un conjunto considerable de software, lenguajes de programación y librerías diferentes. Este elaborado sistema trabajó con datos de 12.5 millones de personas que fueron usuarias entre los años 2001 y 2017, con 435.5 millones de diagnósticos. El codificador automático fue capaz de generar el 88,7% de los diagnósticos antes expuestos, mostrando la robustez de estos algoritmos y la complejidad que pueden llegar a tener en su funcionalidad [23].

Dentro de "*Overview of automatic clinical coding: annotations, guidelines, and solutions for non-English clinical cases at CodiEsp track of CLEF eHealth 2020*", diferentes equipos de trabajo se vieron enfrentados con el objetivo de llegar a los mejores resultados posibles dentro de lo que es la utilización de inteligencia artificial para la codificación de información médica. 22 equipos entregaron diferentes resultados, donde solo 8 lograron una precisión superior al 50% en fases diagnosticas del código implementado, lo que equivale a un 36,3% de los equipos. Esto nos permite ver que la inteligencia artificial puede ser un camino algo complicado a tratar en ámbitos de codificación de información médica [24].

Teniendo cerca de 500 millones de hablantes nativos, también parece la necesidad de tener codificación clínica en el idioma español, como se presenta en "*Transformers for Clinical Coding in Spanish*" el 2021. En este trabajo se utilizaron 3 modelos de aprendizaje por transferencia, multilingual BERT, BETO y XLM-RoBERTa, que soportaban el idioma español, se entrenaron con corpus de oncología para que estos modelos se adaptaran a las particularidades de los textos clínicos en español. Los modelos resultantes se pusieron en 3 tareas de codificación distinta, siguiendo una estrategia de clasificación de frases multietiqueta. En los 3 modelos versionados al español superaron a sus versiones originales en esas tareas [25].

2.8. Discusión

Como antes fue expuesto las fuentes de datos a utilizar en este trabajo serán la terminología MedDRA el corpus etiquetado MADE 1.0. Estas fuentes de datos fueron seleccionadas por su disponibilidad y fiabilidad de la información que hay presentes en estas, destacando que ambas están vinculadas al *"National Center for Biotechnology Information"* (NCBI).

Por parte de la terminología MedDRA, contamos con una codificación para cada síntoma y sus diferentes características asociadas a cada uno de estos, permitiendo ser vinculado a una lista de sinónimos médicos, además de una jerarquía estructural asociada a cada término.

En el caso del corpus MADE 1.0 presenta un conjunto de archivos en texto libre, pero con su respectiva etiquetación de efectos adversos, síntomas, medicamentos, entre otros, permitiendo un análisis más directo a la información relevante para este trabajo.

En cuanto al software, Python destaca por su simplicidad y accesibilidad a diferentes bibliotecas de fácil uso, además de contar con las bibliotecas descritas anteriormente que aportan sus características y funciones dentro de este lenguaje de programación. *Symspellpy* fue una elección segura, librería que contempla muchas mejoras con respecto a otras, tanto en velocidad como en consumo de hardware en el procesamiento, eso sumando a que contempla ese diccionario personalizable que lo hace la librería idónea para el desarrollo de este algoritmo.

Capítulo 3. Materiales y Métodos

3.1. Introducción

En el presente capítulo se describe el corpus MADE y se comenta el preprocesamiento necesario para su uso. Además, se realiza la exploración de la terminología MedDRA, terminando con la descripción de la metodología a utilizar y los algoritmos desarrollados.

3.2. Materiales

3.2.1 Exploración de datos MADE

Dentro de los datos de MADE 1.0 se encuentran 2 carpetas principales, una de estas contiene el corpus con las fichas clínicas y otra las etiquetas asociadas a ellas. Para poder obtener la necesaria a trabajar en esta memoria de título, se procesó y extrajo la totalidad de información etiquetada utilizando Python, así todos los archivos clasificados se almacenaron dentro de una lista para su posterior procesamiento.

Del análisis exploratorio realizado el corpus de fichas clínicas, se pudo observar una inconsistencia de formatos. Si bien la mayoría contiene el formato presente en la figura 3.1, existen registros que no cumplen con este formato. Los registros que no cumplen con estos formatos son propios de cirugías, como se muestran en la figura 3.2, pero aun así hay información útil la cual es extraída en las anotaciones etiquetas en el corpus.

```
[** Hospital **]

Patient: [** Name **], [** Name **]
Acct.#: [** Medical_Record_Number **]
MR#: [** Medical_Record_Number **]
Date of Birth: [** Date **]
Admit: [** Date **]
Date of Service:
Loc: [** Location **]
Dict By: [** Name **] [** Name **] [** Name **] MD
Dict Date: [** Date **]
Trans: [** Date **] 13:54 PM

Clinic Note

MEDICAL DIAGNOSES:
1.  Lymphoplasmacytoid lymphoma involving bone marrow and spleen, diagnosed originally in [** Date **] associated with progressively increasing IgG kappa paraprotein.
2.  Disseminated herpes zoster requiring hospitalization in [** Date **], now resolved.
3.  Compression fractures of the spine secondary to the lymphoma.  He is status post 3 kyphoplasties in [** Date **].  With the first procedure, he developed an acute hemoglobin decrease
for which he was hospitalized and found to have hemorrhagic pericardial effusion, which was drained and found to be nonmalignant.  He received 5 units of red blood cells at that time.  The
etiology of the effusion was never clarified.
4.  Prolonged hospitalization in [** Date **].  He was initially admitted for Salmonella asepsis presenting as excruciating back pain.  He then developed necrotizing fasciitis of the right
gastrocnemius muscle requiring debridement.  He sustained a cardiorespiratory arrest and had a prolonged ICU stay requiring intubation as well as acute hemodialysis.  He developed an
ulceration of the right medial malleolus, which has now healed.  He was hospitalized several more times for recurrent cellulitis of the right leg with the most recent recurrence in [** Date
**].
5.  Bilateral lower extremity DVT with right leg DVT occurring in [** Date **], at which time an IVC filter was placed and the left leg DVT in [** Date **].  He is on chronic Coumadin.
6.  Hypertension.
7.  Hypothyroidism.
8.  Chronic renal insufficiency.
9.  Pancytopenia secondary to his disease as well as chemotherapy.
10.  Peripheral neuropathy secondary to both medications and vasoconstrictors, which were used while he was hypotensive in the ICU in [** Date **].
11.  Recurrent epistaxis, which required hospitalization in [** Date **].  This problem has been quiescent recently.
```

Figura 3.1 Corpus promedio. Fuente: MADE 1.0 [11].

OPERATIVE REPORT

DATE: [** Date **]

PATIENT NAME: [** Name **] [** Name **] MR/EN #: [** Medical_Record_Number **]

SURGEON: [** Name **] [** Name **], M.D.,

PREOPERATIVE DIAGNOSIS: Vertebral Body Fracture at L2 and T9 with Intractable Pain

POSTOPERATIVE DIAGNOSIS: Same

COMMORBIDITIES: Lymphoma, Hypercalcemia, MRSA celullitis, CRI, Psoriasis, HTN, Intractable pain

ESTIMATED BLOOD LOSS: Minimal

ANESTHESIA: MAC

DESCRIPTION OF PROCEDURE: Vertebral body augmentation therapy via Kyphoplasty at L2 and T9
After fully informed written consent was obtained, the patient was placed in the prone position. Monitoring of pulse oximetry, heart rate, and blood pressure was done pre-procedure, during the procedure, and post-procedure. 1 g of Ancef was given pre-procedure. Conscious sedation was administered. The skin was prepped with povidone-iodine three times, and draped in a sterile fashion. The entry site was over the L3 pedicle. With a 25-gauge, 3.5 inch spinal needle, 1 % lidocaine was injected over the entry site subcutaneously and periosteal. A small skin incision was made and an 11-gauge needle was placed through the pedicle into the vertebral body under AP and lateral fluoroscopic guidance. The provided drill in the Kyphon set was utilized. After drilling into the anterior

Figura 3.2 Ficha del corpus orientada a operación Fuente: MADE 1.0 [11].

Ya viendo las anotaciones, se tienen varios archivos de extensión .xml, que todas poseen similitud con la figura 3.3, donde se puede apreciar el id al cual está asociando dentro del corpus, el número de caracteres que posee la palabra clasificada, su clasificación y el offset de esta, dando así la mayor información posible dentro de las opciones que permite este formato.

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<collection>
  <source/>
  <date/>
  <key/>
  <document>
    <id>1_9</id>
    <passage>
      <offset>0</offset>
      <annotation id="591">
        <infon key="type">SSLIF</infon>
        <location length="6" offset="3589"/>
        <text>fevers</text>
      </annotation>
      <annotation id="592">
        <infon key="type">SSLIF</infon>
        <location length="12" offset="3597"/>
        <text>night sweats</text>
      </annotation>
      <annotation id="593">
        <infon key="type">SSLIF</infon>
        <location length="6" offset="3611"/>
        <text>chills</text>
      </annotation>
      <annotation id="594">
        <infon key="type">SSLIF</infon>
        <location length="10" offset="3665"/>
        <text>chest pain</text>
      </annotation>
      <annotation id="595">
        <infon key="type">SSLIF</infon>
        <location length="5" offset="3677"/>
        <text>cough</text>
      </annotation>
      <annotation id="596">
        <infon key="type">SSLIF</infon>
        <location length="7" offset="3684"/>
        <text>dyspnea</text>
      </annotation>
```


Figura 3.3 Ejemplo de datos etiquetados del Corpus. Fuente: MADE 1.0 [11].

Dentro del número de caracteres que compone a cada información presente en MADE, se pudo obtener el gráfico de cajas de la figura 3.4, en el cual se destaca que la cantidad de caracteres por término etiquetado se mantiene inferior a 20, además la mediana general está entre 5 y 15 caracteres. De los grupos de interés, ADE posee un grupo consistente mostrando cuartiles compactos con una mediana próxima a 15 caracteres, y por el lado de SSLIF existe una extensión un poco más amplia del tercer cuartil, pero con una mediana igual a la de ADE, destacándose la gran cantidad de valores atípicos dentro de los datos.

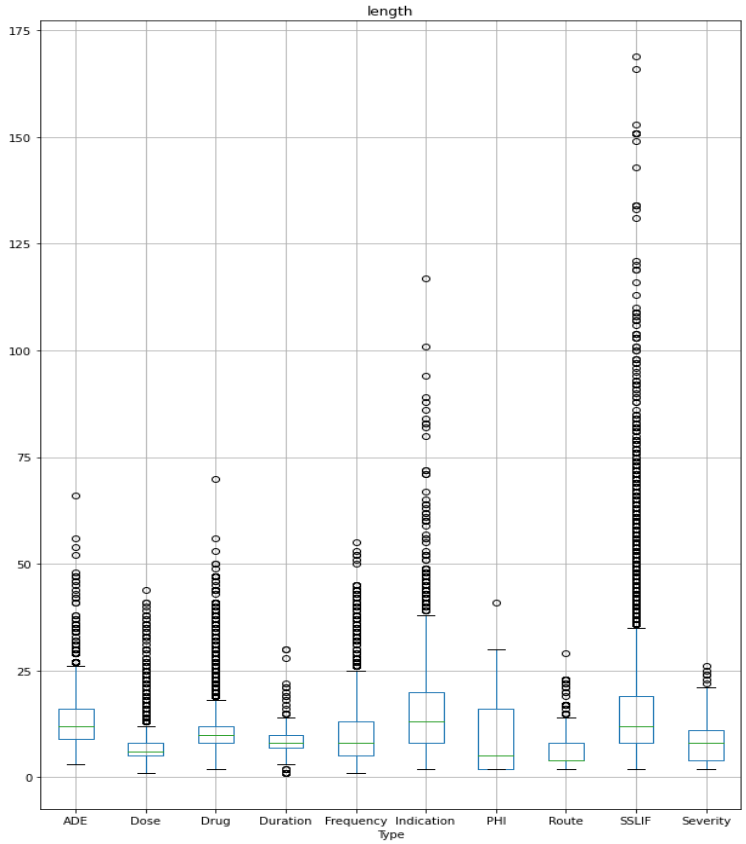

Figura 3.4 Gráfico de cajas vinculado a caracteres por dato etiquetado.

En cuanto a las palabras presentes en cada dato etiquetado, también se separó por categoría, encontrándose con la figura 3.5, gráfico de caja que muestra una compactación clara de varias categorías en 1 o 2 palabras. Los grupos de interés ambos están entre 1 a 3 palabras como máximo, **destacándose que nuevamente SSLIF posee más datos atípicos.**

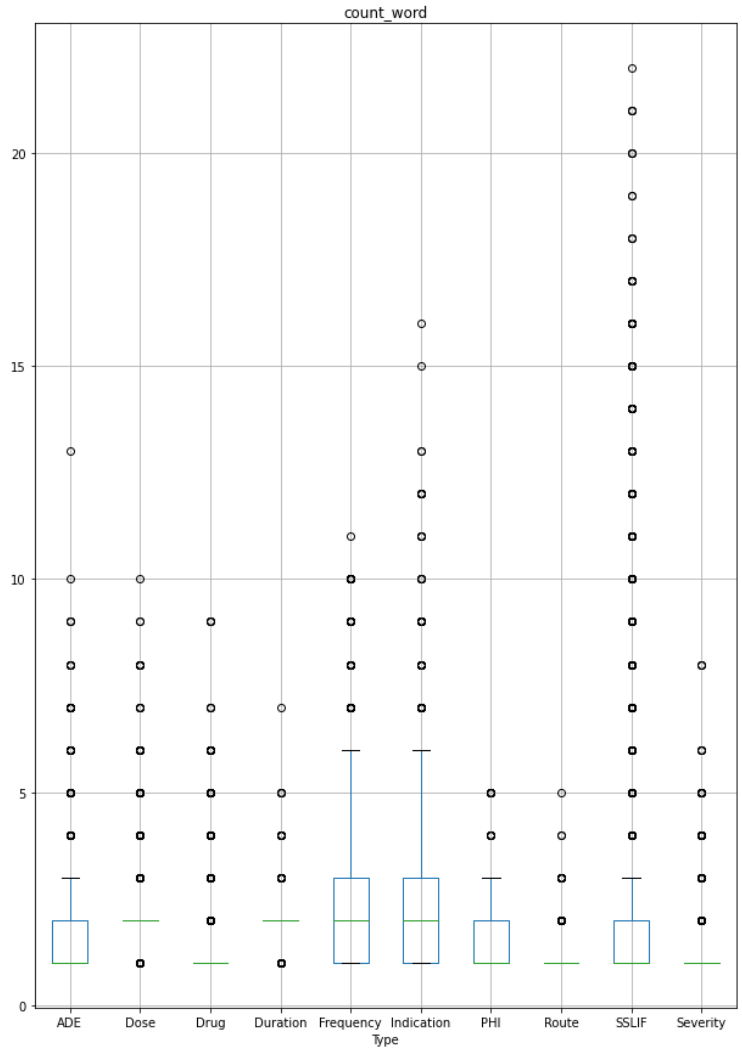

Figura 3.5 Gráfico de caja vinculado a las palabras por dato etiquetado.

Para el tratamiento de los datos se almacena los datos etiquetados en un archivo ".tsv", para luego poder ser utilizados posteriormente, se mantiene la mayor información posible extraída, los cuales son:

- Número de caracteres en las palabras
- El ID dentro del etiquetado
- Clasificación a la que pertenece
- Texto que contiene la información etiquetada
- Número de palabras en cada dato etiquetado

3.2.2 Exploración de MedDRA

La información presente en MedDRA contempla todos los términos que pueden encontrar en relación a efectos adversos, enfermedades o síntomas. Hay presentes un total de 95.905 datos con su CUI, el tipo, su link asociado y su nombre del efecto adverso, esto permite entregar una asociación completa a la información presente dentro de la terminología. Este archivo de extensión “tsv”, es de fácil acceso desde Python y se puede lograr manipular de manera rápida y precisa con las librerías necesarias.

Dentro de los datos presentes encontramos términos inferiores a 5 palabras dentro de la mayoría de los datos, fuera de ello se encuentran excepciones de hasta 22 palabras como se aprecia en la figura 3.6, pero que, a pesar de ello, cuentan con su respectiva codificación e información.

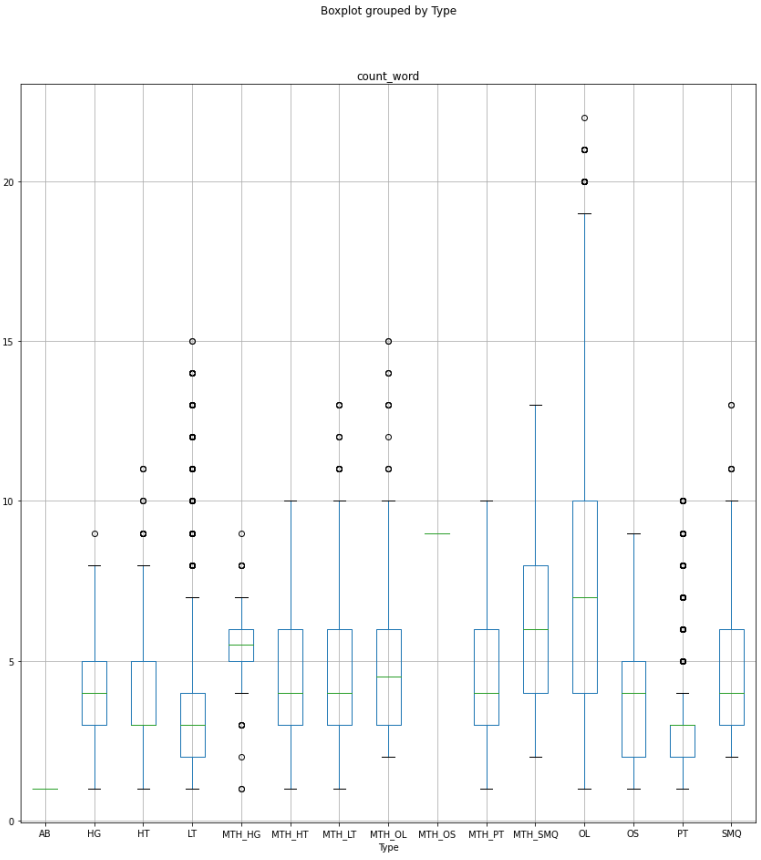

Figura 3.6 Gráfico de caja vinculado a MedDRA

3.3. Metodología

A. Pre-procesamiento de MADE

Para poder acceder a la información de MADE 1.0 se realiza la extracción de los datos etiquetados en Python. Se utiliza la biblioteca xmlx con el objetivo de poder obtener toda la información necesaria del corpus etiquetado. Se encuentra un total de 67781 datos etiquetados, los cuales se graficaron en relación a las palabras presentes y los caracteres de cada uno versus su respectiva clasificación. De estos datos los grupos tanto ADE como SSLIF son los de interés, ya que estos indican los efectos secundarios y las enfermedades sin vinculo aparente a medicamentos, los cuales nos interesa recodificar más adelante.

Cabe destacar que fuera del primer algoritmo, en esta sección se realizará una normalización de la información, quitando caracteres atípicos, saltos de línea, espacios, mayúsculas y puntaciones.

B. Preparación de diccionario

El diccionario que necesita la biblioteca *symspellpy* se crea en base a la terminología de MedDRA, a esta última se le realiza una lectura en Python, del cual se extrae la columna de términos y es guardada como un nuevo archivo, “diccionario.txt”.

Cabe destacar que fuera del primer algoritmo, en esta sección se realizará una normalización de la información, quitando caracteres atípicos, saltos de línea, espacios, mayúsculas y puntaciones.

C. Identificación de términos

Este proceso se lleva a cabo con la biblioteca *symspellpy* configurada de tal manera que utiliza el diccionario antes creado para encontrar términos similares en MADE, mediante el uso de la distancia de Levenshtein. En este punto se configura la distancia máxima de Levenshtein permitida según lo deseado. Para este trabajo se estimó el uso de todos sus rangos permitidos “0”, “1”, “2” y “None”. Ya con la sección antes configurada entra en el proceso de identificar términos de MedDRA en el corpus MADE pre-procesado. La información que encontró un término similar en MedDRA es guardada en un archivo aparte de los que no se encontraban en MedDRA.

D. Recodificación de MADE

Ya con todos los términos identificados, se compara la lista de MADE que encontraron su término en MedDRA en el proceso anterior y se les agrega toda la información presente en la

terminología MedDRA, Otorgándole mayor profundidad informativa a los términos de MADE etiquetados.

E. Separación de resultados

Llegados a este momento los términos de MADE etiquetado a los que se les asignó la información presente en la terminología MADE entran en un proceso de identificación de errores en la recodificación. Esta identificación se realiza mediante el uso de comparadores directos entre el término de MedDRA y el de MADE que se asignaron como similares. Luego comienza un análisis manual dentro de estos "errores" despejando aún más términos que si están bien clasificados, pero que no poseen una similitud exacta (términos que apuntan a la misma condición médica o efecto adverso y que son agregados por la distancia de Levenshtein permitida en cada uno de los casos). Para este nivel de la metodología tendríamos 3 archivos relevantes generados dentro de nuestro algoritmo:

- **Archivo de no clasificados:** Formado por aquellos que no pudieron encontrar un término dentro de MedDRA.
- **Archivo de Clasificados:** Compuesto por los datos que si encontraron un término dentro de MedDRA.
- **Archivo de Errores:** Contiene los términos que fueron mal recodificados con la información de MedDRA.

F. Análisis de resultados, errores y no clasificados

Se finaliza el experimento con la extracción de resultados, conclusiones de cada algoritmo y la presentación de posibles mejoras para el siguiente algoritmo.

3.3.2 Evaluación del modelo.

Para poder tener una referencia de la cual partir y comparar los resultados que más adelante entregaran los algoritmos modificados, se tendrá un algoritmo de control o referencia. Este algoritmo funcionará si ningún tipo de normalización ni tratamiento realizado al texto libre. Esto nos permitirá generar una base de resultados que, si más adelante un algoritmo modificado obtiene mejores o peores resultados, se apuntara directamente a los cambios realizados en el algoritmo como provocador de este cambio en los resultados. Con esto se puede identificar con mayor precisión los

cambios necesarios. Este algoritmo se hará funcionar con los 4 niveles permitidos de distancia de Levenshtein máxima (0,1,2 y None).

Como en todo trabajo en el cual se utiliza texto libre, se debe tener presente la normalización. El siguiente algoritmo contará con ello. Se le realizará una normalización tanto al Corpus etiquetado de MADE como a la terminología de MedDRA, la cual consiste en eliminar saltos de línea, espacios, guiones, comas, barras inversas y cambiando mayúsculas por minusculas. Además, se comparará con los resultados anteriores y se analizarán los archivos de nuevos resultados en búsqueda de necesidades generadas por esta normalización. Este algoritmo se hará funcionar con los 4 niveles permitidos de distancia de Levenshtein máxima (0,1,2 y None). Tanto con los resultados obtenidos en el algoritmo de control y en este algoritmo, se determina cual o cuales son las distancias máximas recomendadas para utilizar en una recodificación de corpus médico.

En el algoritmo final se presentará mejoras relacionadas a la distancia máxima de Levenshtein, otras enfocadas en solucionar los problemas que surjan con la normalización, si es el caso. Con ello se buscará obtener los resultados más altos hasta ahora y poder determinar tanto los valores máximos de Levenshtein recomendados para la tarea, como las razones detrás de los resultados obtenidos al analizar en profundidad los archivos generados con este algoritmo.

Los porcentajes presentes en los resultados se extraen de la siguiente manera:

Porcentaje total (PT)

$$PT = \frac{Ac}{NT} * 100$$

Donde,

PT : porcentaje de términos acertados, en relación al total de términos etiquetados en MADE.

Ac : Total de términos acertados por el algoritmo.

NT : Total de términos etiquetados en MADE.

Esto nos indica que, si el PT es 100%, el algoritmo logró recodificar todos los términos etiquetados en MADE 1.0

Porcentaje de acierto v/s total recodificados (PA)

$$PA = \frac{Ac}{TC} * 100$$

Donde,

PA : Porcentaje de términos acertados, en comparación del total de términos recodificados por el algoritmo (errores y aciertos).

Ac : Total de términos acertados por el algoritmo.

TC : Total de términos recodificados por el algoritmo (aciertos y errores).

Esto nos indica que, si el PA es 100%, todos los términos que el algoritmo recodificó están acertados.

Porcentaje de acierto únicos v/s total recodificados únicos (PDU)

$$PDU = \frac{AU}{TCU} * 100$$

Donde,

PDU: Porcentaje de términos acertados únicos, en comparación del total de términos únicos recodificados por el algoritmo (errores y aciertos).

AU : Total de términos únicos acertados por el algoritmo.

TCU : Total de términos únicos recodificados por el algoritmo (aciertos y errores).

Esto nos indica que, si el PDU es 100%, todos los términos únicos que el algoritmo recodificó están acertados.

3.4. Algoritmos desarrollados

3.4.1 Algoritmo de Referencia

El algoritmo de referencia se desarrollará de la siguiente manera:

1. Extracción de datos etiquetados del corpus MADE 1.0.
2. Guardado de información extraída en MADE data.csv.
3. Apertura del archivo MedDRA.tsv.
4. Se utiliza este último archivo para crear el archivo diccionario.txt.
5. Se llama al archivo diccionario.txt con la librería symspellpy para utilizarlo como base para encontrar similitudes con los datos a procesar.
6. Se abre MADE data.csv y se separan los datos del tipo SSLIF y ADE para su clasificación.
7. Se procesan los datos antes separados y se les asigna una similitud presente dentro del diccionario si es que encuentra alguna.
8. Estos resultados quedan guardados dentro de un archivo "resultado_comparacion.tsv". Mediante la apertura de este último, se vincula la sugerencia encontrada y su respectivo símil dentro de la terminología de MedDRA, llegando así a la codificación asociada a esa "sugerencia".

9. Se filtran los datos recodificados con comparación directa entre la sugerencia y su dato perteneciente a MADE 1.0, para obtener los que serán considerados errores de recodificación.

A continuación, se tendrá un diagrama de flujo en la figura 3.7 referente al programa principal, para una mayor comprensión del proceso antes descrito.

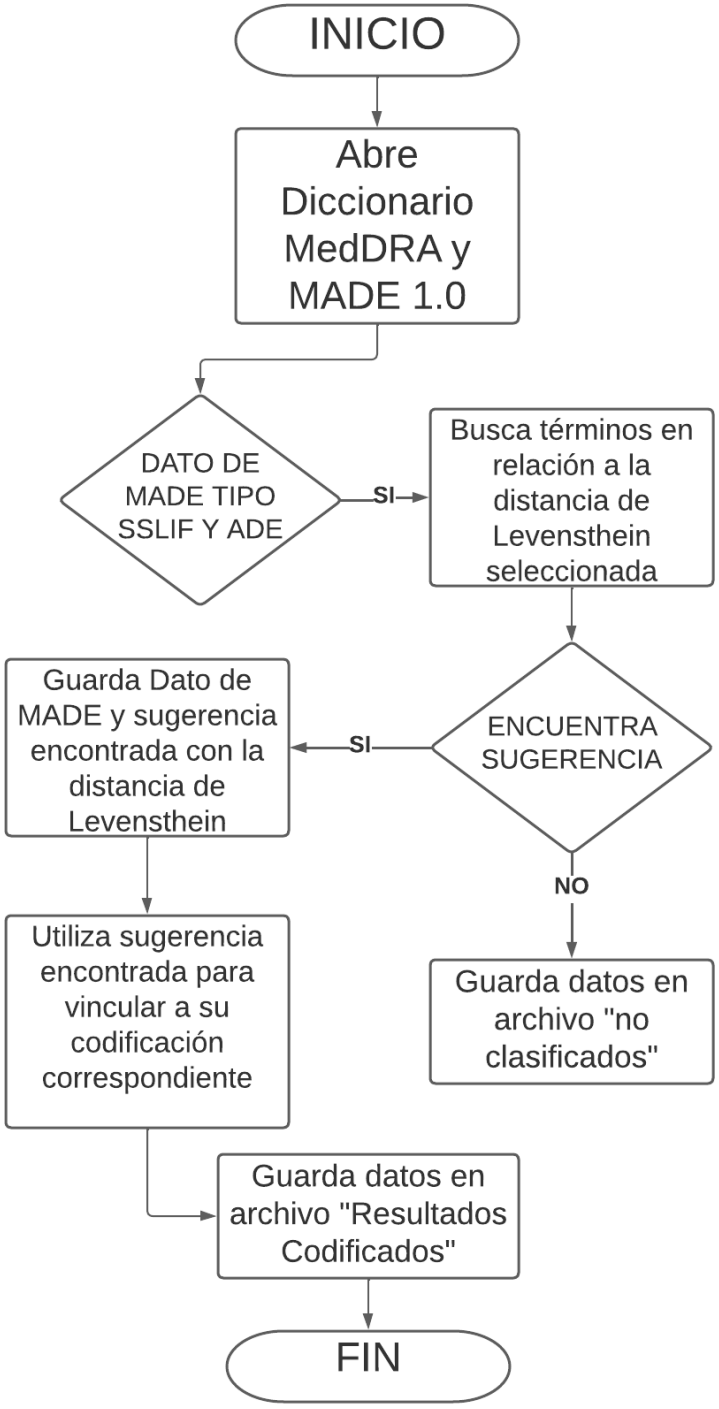

Figura 3.7 Diagrama de Flujo Algoritmo de Referencia.

3.4.2 Algoritmo Normalizado

En este algoritmo se desarrollará la normalización y se estudiarán sus efectos dentro de la recodificación. Se normalizará tanto el corpus etiquetado de MADE 1.0, como la terminología de MedDRA. Además de hacer este cambio, el algoritmo fue modificado en la sección de recodificación final, acorde a las necesidades presentes por este nuevo diccionario normalizado. Se realizaron pruebas en las mismas variaciones de distancia de Levenshtein máxima que con el primer código y se analizaron sus errores.

El nuevo diagrama del código queda descrito a continuación en la figura 3.8:

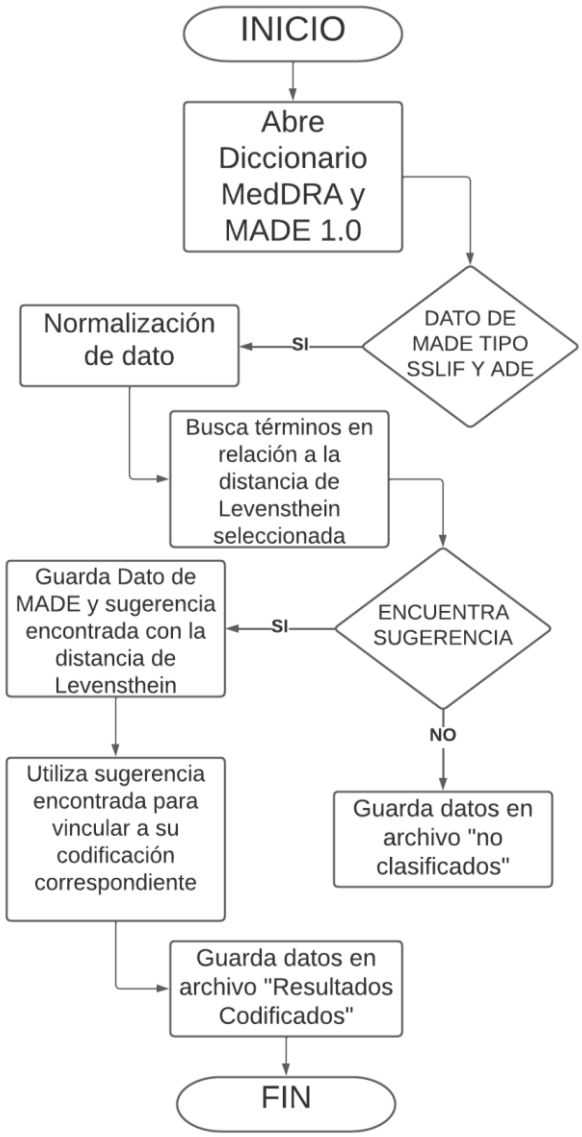

Figura 3.8 Diagrama de Flujo Algoritmo Normalizado.

3.4.3 Algoritmo Final

En este algoritmo se agregan nuevas consideraciones, que son:

1. **Implementación de un extensor de abreviaciones médicas:** para complementar la codificación de MedDRA.
2. **Diferentes fases de revisión:** que apuntarán a encontrar el resultado más indicado dentro de la búsqueda, aumentado la distancia de Levenshtein si este no encuentra el resultado en la distancia anterior. La máxima distancia permitida es 1.
3. **Trato especial a palabras cortas y abreviaciones en el código:** ya que el considerar la Distancia de Levenshtein igual a 1, el error era muy alto para las palabras cortas, interfiriendo en su correcta clasificación, por lo que se trataran solo con distancia de Levenshtein igual a 0.

A continuación, el nuevo diagrama del código queda descrito a continuación en la figura 3.9:

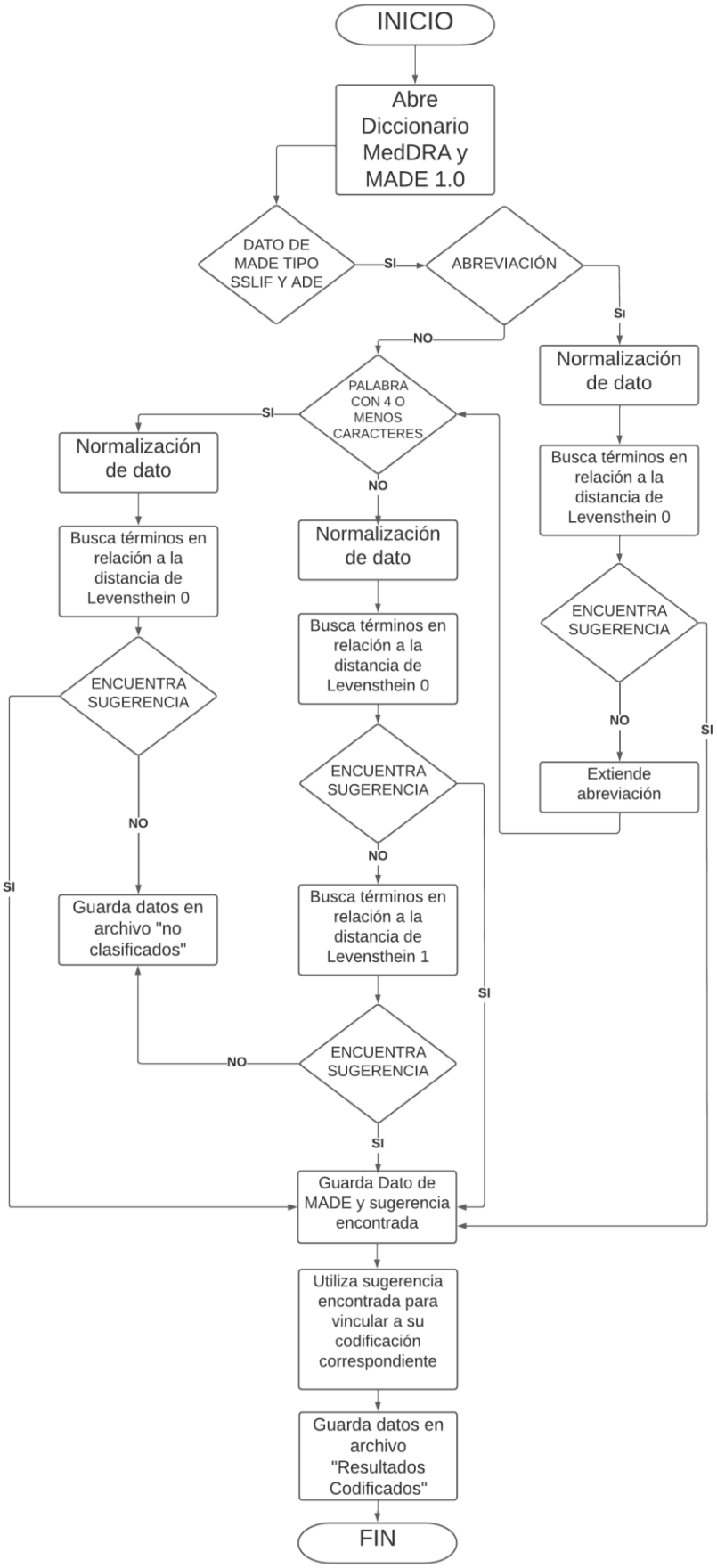

Figura 3.9 Diagrama de Flujo Algoritmo Final.

Capítulo 4. Resultados

4.1. Introducción

En el presente capítulo se describen los resultados obtenidos en este trabajo, de acuerdo con el orden descrito previamente en la metodología.

4.2. Resultados algoritmo de Referencia

El algoritmo anterior se probó en diferentes parámetros de distancia de Levenshtein máxima para llegar a los distintos resultados presentes a continuación:

- **Distancia de Levenshtein máxima igual a 0:**

  De un total de **35.565** datos a recodificar dentro de la base de datos MADE 1.0, el algoritmo bajo estas condiciones logro recodificar **1.892** datos con **0** datos erróneos.

  Por lo que este algoritmo logró recodificar 1.892 registros correspondiente a un **5,32%** del total de etiquetas (35.565 anotaciones originales). De estas **1.892** recodificaciones, el **100%** fueron realizadas correctamente.

  Tabla 4.1 Resultados Algoritmo de Referencia con Levenshtein igual a 0.

	DATOS TOTALES	DATOS UNICOS
Datos clasificados	1892	267
Erróneos	0	0
Total bien clasificados	1892	267
Porcentajes clasificados v/s correctos	100%	100%

- **Distancia de Levenshtein máxima igual a 1:**

  De un total de **35.565** datos a recodificar dentro de la base de datos MADE 1.0, el algoritmo bajo estas condiciones logro recodificar **19.443** datos, con **17.551** datos no exactamente iguales, pero si se omiten los espacios y mayúsculas de las diferencias entre el dato MADE 1.0 y su sugerencia, los errores se reducen a **767** datos.

  Esos **767** errores van vinculados a malas clasificaciones que permite la distancia máxima aplicada o las abreviaciones. Pero hay 313 datos que están bien clasificados, ya que existen situaciones como “edema” y “Oedema” que, si están bien

asociados, pero no son exactamente iguales para ser descartados, esto quiere decir que el total final de datos erróneos es de **454**.

Por lo que este algoritmo logró recodificar **19.443** registros correspondiente a un **53,3%** del total de etiquetas (35.565 anotaciones originales). De estas **19.443** recodificaciones, el **97.66%** fueron realizadas correctamente.

Tabla 4.2 Resultados Algoritmo de Referencia con Levenshtein igual a 1.

	DATOS TOTALES	DATOS UNICOS
Datos clasificados	19443	1065
Erróneos	434	60
Total bien clasificados	19009	1005
Porcentajes clasificados v/s correctos	97,768%	94,366%

- **Distancia de Levenshtein máxima igual a 2:**

De un total de **35.565** datos a recodificar dentro de la base de datos MADE 1.0, el algoritmo bajo estas condiciones logro recodificar **23.397** datos, con **21.505** datos no exactamente iguales, pero si se omiten los espacios y mayúsculas de las diferencias entre el dato MADE 1.0 y su sugerencia, los errores se reducen a **4.665** datos.

Esos 4.665 errores van vinculados a malas clasificaciones que permite la distancia máxima aplicada o las abreviaciones. Pero hay 313 datos que están bien clasificados, ya que existen situaciones como “edema” y “Oedema” que, si están bien asociados, pero no son exactamente iguales para ser descartados, esto quiere decir que el total final de datos erróneos es de **4.352**.

Por lo que este algoritmo logró recodificar **23.397** registros correspondiente a un **53,55%** del total de etiquetas (35.565 anotaciones originales). De estas **23.397** recodificaciones, el **81.39%** fueron realizadas correctamente.

Tabla 4.3 Resultados Algoritmo de Referencia con Levenshtein igual a 2.

	DATOS TOTALES	DATOS UNICOS
Datos clasificados	23397	1384
Erróneos	4352	370
Total bien clasificados	19045	1014
Porcentajes clasificados v/s correctos	81,399%	73,266%

- **Distancia de Levenshtein máxima igual a None:**

  De un total de **35.565** datos a recodificar dentro de la base de datos MADE 1.0, el algoritmo bajo estas condiciones logro recodificar **23.397** datos, con 21505 datos no exactamente iguales, pero si se omiten los espacios y mayúsculas de las diferencias entre el dato MADE 1.0 y su sugerencia, los errores se reducen a **4.665** datos.

  Esos 4.665 errores van vinculados a malas clasificaciones que permite la distancia máxima aplicada o las abreviaciones. Pero hay 313 datos que están bien clasificados, ya que existen situaciones como "edema" y "Oedema" que, si están bien asociados, pero no son exactamente iguales para ser descartados, esto quiere decir que el total final de datos erróneos es de **4.352.**

  Por lo que este algoritmo logró recodificar **23.397** registros correspondiente a un **53,55%** del total de etiquetas (35.565 anotaciones originales). De estas **23.397** recodificaciones, el **81.39%** fueron realizadas correctamente.

Tabla 4.4 Resultados Algoritmo de Referencia con Levenshtein igual a None.

	DATOS TOTALES	DATOS UNICOS
Datos clasificados	23397	1384
Erróneos	4352	370
Total bien clasificados	19045	1014
Porcentajes clasificados v/s correctos	81,399%	73,266%

4.2.1 Análisis de resultados.

Como se pudo apreciar en los diferentes resultados, todos tuvieron una menor cantidad de datos recodificados en relación al total de datos extraídos del corpus MADE 1.0. A pesar de lo expuesto anteriormente en relación a la idea de estandarización de la terminología MedDRA, no llega a cubrir las diferentes necesidades de codificación que exige MADE 1.0, por lo que se puede prever la necesidad de inclusión de nuevos códigos o sinónimos a MedDRA.

La opción de hacer un análisis con la menor normalización posible, buscaba encontrar los porcentajes y cantidades mínimas de clasificación para futuros cambios dentro del software. Esto quiere decir que, si en las modificaciones siguientes se obtienen menores porcentajes de acierto o menor número de información recodificada, los cambios hechos agregan más errores que

soluciones. Por el contrario, si lo porcentajes o totales de información recodificada aumentan, esto será gracias a los cambios realizados en el algoritmo.

4.3. Resultados de Algoritmo Normalizado

Los Resultados para este algoritmo modificado fueron sacados de igual manera que el algoritmo inicial, para realizar una comparación más completa posible:

- **Distancia de Levenshtein máxima igual a 0:**

  De un total de **35.565** datos a recodificar dentro de la base de datos MADE 1.0, el algoritmo bajo estas condiciones logro recodificar **16.672** datos con **0** datos erróneos.

  Por lo que este algoritmo logró recodificar **16.672** registros correspondiente a un **46,88%** del total de etiquetas (**35.565** anotaciones originales). De estas **16.672** recodificaciones, el **100%** fueron realizadas correctamente.

Tabla 4.5 Resultados Algoritmo Normalizado con Levenshtein igual a 0.

	DATOS TOTALES	DATOS UNICOS
Datos clasificados	16672	913
Erróneos	0	0
Total bien clasificados	16672	913
Porcentajes clasificados v/s correctos	100%	100%

- **Distancia de Levenshtein máxima igual a 1:**

  De un total de **35.565** datos a recodificar dentro de la base de datos MADE 1.0, el algoritmo bajo estas condiciones logro recodificar **19.443** datos, con **1.761** datos erróneos.

  Nuevamente de esos 1.761 errores van vinculados a malas clasificaciones que permite la distancia máxima aplicada o las abreviaciones normalizadas. Pero hay 313 datos que están bien clasificados, ya que existen situaciones como “edema” y “Oedema” que, si están bien asociados, pero no son exactamente iguales para ser descartados, esto quiere decir que el total final de datos erróneos es de **1.448.**

Por lo que este algoritmo logró recodificar **19.443** registros correspondiente a un **50,597%** del total de etiquetas (35.565 anotaciones originales). De estas **19.443** recodificaciones, el **92,55%** fueron realizadas correctamente.

Tabla 4.6 Resultados Algoritmo Normalizado con Levenshtein igual a 1.

	DATOS TOTALES	DATOS UNICOS
Datos clasificados	19443	1168
Erróneos	1448	181
Total bien clasificados	17995	987
Porcentajes clasificados v/s correctos	92,553%	84,503%

- **Distancia de Levenshtein máxima igual a 2:**

De un total de **35.565** datos a recodificar dentro de la base de datos MADE 1.0, el algoritmo bajo estas condiciones logro recodificar **32.834** datos, con **14.542** datos erróneos.

Nuevamente de esos 14.542 errores van vinculados a malas clasificaciones que permite la distancia máxima aplicada o las abreviaciones normalizadas. Pero hay 313 datos que están bien clasificados, ya que existen situaciones como “edema” y “Oedema” que, si están bien asociados, pero no son exactamente iguales para ser descartados, esto quiere decir que el total final de datos erróneos es de **14.229**.

Por lo que este algoritmo logró recodificar **32.834** registros correspondiente a un **52,31%** del total de etiquetas (35.565 anotaciones originales). De estas **32.834** recodificaciones, el **56,66%** fueron realizadas correctamente.

Tabla 4.7 Resultados Algoritmo Normalizado con Levenshtein igual a 2.

	DATOS TOTALES	DATOS UNICOS
Datos clasificados	32834	2383
Erróneos	14229	567
Total bien clasificados	18605	1816
Porcentajes clasificados v/s correctos	56,664%	76,206%

- **Distancia de Levenshtein máxima igual a None:**

De un total de 35.565 datos a recodificar dentro de la base de datos MADE 1.0, el algoritmo bajo estas condiciones logro recodificar **32.834** datos, con **14.542** datos erróneos.

Nuevamente de esos 14.542 errores van vinculados a malas clasificaciones que permite la distancia máxima aplicada o las abreviaciones normalizadas. Pero hay 313 datos que están bien clasificados, ya que existen situaciones como “edema” y “Oedema” que, si están bien asociados, pero no son exactamente iguales para ser descartados, esto quiere decir que el total final de datos erróneos es de **14.229**.

Por lo que este algoritmo logró recodificar **32.834** registros correspondiente a un **52,31%** del total de etiquetas (35.565 anotaciones originales). De estas **32.834** recodificaciones, el **56,66%** fueron realizadas correctamente.

Tabla 4.8 Resultados Algoritmo Normalizado con Levenshtein igual a None.

	DATOS TOTALES	DATOS UNICOS
Datos clasificados	32834	2383
Erróneos	14229	567
Total bien clasificados	18605	1816
Porcentajes clasificados v/s correctos	56,664%	76,206%

4.3.1 Análisis algoritmo Normalizado

En cuanto a los resultados del algoritmo modificado y el inicial, se tiene que todos los porcentajes son menores en el algoritmo modificado, tanto en los datos totales bien recodificados como en los que fueron recodificados, a excepción de aquel que presenta una distancia máxima de Levenshtein igual a 0, ya que en este se mantuvo el acierto de un 100% y aumento los datos recodificados de **5.32%** en el código inicial a un **46,88%**. En los demás casos todos eran menores que el algoritmo inicial, por lo que el utilizar la distancia de Levenshtein máxima igual a 0 y luego continuar con una distancia de Levenshtein de mayor libertad, podría llevar a mejores resultados.

Entrando un poco más en detalle con las abreviaciones, ocurre algo complicado, ya que la recodificación resulta alterada donde son tan pocas las letras que lo componen, el algoritmo las considera palabras cortas, no como abreviaciones. Además, la terminología de MedDRA cuenta con muy pocas abreviaciones. Esto llevó a la necesidad de poder discriminar entre palabras cortas y abreviaciones en algún momento del algoritmo. Con respecto a las pocas abreviaciones presentes en la terminología MedDRA, se optó por extender las abreviaciones en caso de no ser encontradas dentro de una primera revisión en el algoritmo, llevando a esa abreviación a una segunda búsqueda dentro de la terminología MedDRA, pero ahora extendida. Para terminar el uso de una distancia de Levenshtein permisiva dentro de palabras con muy pocos caracteres, como es el caso de las

abreviaciones o palabras cortas, tiende mucho a hacer relaciones erróneas, por lo que, en estos tipos de términos, la distancia de Levenshtein debe ser estricta (igual a 0).

4.4. Resultados Algoritmo Final

Estos cambios permitieron llegar a los siguientes resultados:

Tabla 4.9 Resultados Algoritmo Final (solo hasta Levenshtein igual a 1).

	DATOS TOTALES	DATOS UNICOS
Datos clasificados	20898	1232
Erróneos	537	156
Total bien clasificados	20361	1076
Porcentajes clasificados v/s correctos	97,430%	87,338%

Como se puede apreciar en la tabla 4.9, se pudo lograr una mejora en cuanto a los porcentajes de aciertos, pero dentro de todo no se pudo obtener porcentajes superiores a **87,338%** dentro de los datos únicos. En cuanto al porcentaje de aciertos v/s el total de datos es de **57,25%,** obteniendo un aumento leve con respecto a algoritmo sin las 3 consideraciones agregadas antes explicadas.

Se realizaron pruebas permitiendo una distancia de Levenshtein máxima igual a 2 y se obtuvieron los siguientes resultados:

Tabla 4.10 Resultados Algoritmo Final (hasta Levenshtein igual a 2).

	DATOS TOTALES	DATOS UNICOS
Datos clasificados	23992	1832
Erróneos	4907	486
Total bien clasificados	19085	1346
Porcentajes clasificados v/s correctos	79,547%	73,472%

Como era de esperarse al dejar como tope la distancia de Levenshtein igual a 2, agrega resultados positivos, pero también agrega erróneos en cantidades un poco mayores provocando que los porcentajes sean peores que al limitar la distancia de Levenshtein a 1. En cuanto al porcentaje de aciertos v/s el total de datos es de **53,662%,** lo que es claramente menor al valor con distancia de Levenshtein igual a 1, por lo que ni por ese sentido se debe tener en cuenta la utilización de la distancia de Levenshtein igual a 2.

4.5. Discusión

A pesar de que el código acierta en grandes cantidades con respecto a los clasificados, el porcentaje de clasificados es cercano al 60%, esto permite entrever la necesidad de algunas mejoras en el algoritmo y las bases de datos utilizadas.

En la base de MADE encontramos:

1. Términos muy extensos o que especifican demasiado, alejándose de una posible clasificación.
2. Términos cortos que hacen referencia a condiciones médicas.
3. Términos muy generales que no encuentran clasificación por su poca especificación.
4. Excepciones que difieren en algunos caracteres o letras, son pocos, pero son los suficientes como para no encontrar su clasificación.

A continuación, se presentan diferentes ejemplos en relación a los tipos de errores antes descritos:

Tabla 4.11 Tipos de Errores en Datos no Clasificados.

Tipo de Error	Error
Tipo 1	cannot stand at a countertop for any long periods.
	ulcer on the medial malleolus of the right ankle.
	pain was localized to the right sacroiliac joint area.
	cellulitis of his right lower extremity that involves the surgical wound.
	Lymphoplasmacytic lymphoma involving the bone marrow and spleen.
Tipo 2	Limp
	Heat
	Fell
	skin is dry
	Inactive
Tipo 3	Lesion
	Gallop
	Murmur
	Tamponade
	Poorly
Tipo 4	dysfunctional bleeding
	spinal stenosis
	pain in his legs
	Dribbling
	Straining

Por lo que al evitar estos errores en otras bases de datos o modificar el algoritmo lo suficiente como para que estos puntos antes explicados no sean un problema, ayudaría a mejorar consistentemente los resultados.

Tabla 4.12 Resultados Algoritmo de Referencia.

	clasificados v/s correctos (%)	clasificados únicos v/s correctos únicos (%)	Total de Datos Bien clasificados
Levenshtein 0	100	100	1892
Levenshtein 1	97,768	94,366	19009
Levenshtein 2	81,399	73,266	19045
Levenshtein None	81,399	73,266	19045

Tabla 4.13 Resultados Algoritmo Normalizado.

	clasificados v/s correctos (%)	clasificados únicos v/s correctos únicos (%)	Total de Datos Bien clasificados
Levenshtein 0	100	100	16672
Levenshtein 1	92,553	84,503	17995
Levenshtein 2	56,664	76,206	18605
Levenshtein None	56,664	76,206	18605

Tabla 4.14 Resultados Algoritmo Final.

	clasificados v/s correctos (%)	clasificados únicos v/s correctos únicos (%)	Total de Datos Bien clasificados
Hasta Levenshtein 1	97,43	87,338	20361
Hasta Levenshtein 2	79,547	73,472	19085

Como podemos apreciar en las tablas 4.12 y 4.13, los resultados variaron positivamente el total de datos bien clasificados dentro de los rangos de Levenshtein máximo igual a 0 con la normalización. En los otros casos hubo cierta tendencia a disminuir en los casos de Levenshtein máximo igual a 0. Además, se destaca que la distancia de Levenshtein igual a 2 y None, poseen los mismos resultados.

Ya pasando al algoritmo final, en la tabla 4.14 se logra destacar el funcionamiento del algoritmo hasta la distancia de Levenshtein 1, además de la mejora de los resultados dentro de los porcentajes de datos clasificados v/s correctos. Esto último acompañado de un aumento leve en comparación a los algoritmos anteriores en el total de datos bien clasificados.

El algoritmo realizado logra clasificaciones rápidas dentro de lo buscado, pero con cierto nivel de precisión que puede ser mejorado, las complicaciones propias del lenguaje ponen una barrera consistente dentro de lo que es la automatización de codificaciones para poder abarcar un mayor número de aciertos. Los resultados en si son prometedores, ya que, a pesar de las complicaciones encontradas en las bases de datos, los aciertos en relación con los errores de clasificación, posee un porcentaje alto, lo que nos permite ver que a medida que el algoritmo se vaya mejorando, la positividad de clasificación esperada será similar o superior.

Capítulo 5. Conclusiones

5.1. Discusión

En el primer algoritmo, ya se comenzaron a dar los primeros indicios de cuál era la distancia de Levenshtein ideal a utilizar. Los resultados de la primera parte fueron categóricos para determinar que la mejor distancia máxima de Levenshtein es igual a 1. Con un total de datos alto y un porcentaje de acierto de **97,768%** y **94,366%** en datos totales y datos únicos respectivamente.

En el algoritmo modificado se redujo un poco los porcentajes en la distancia máxima de Levenshtein igual a 1, pero de esta sección modificada se destaca el aumento considerable de datos totales y datos únicos clasificados con distancia máxima de Levenshtein igual a 0.

Llegando al algoritmo final se quiso aprovechar el aumento de la sección modificada y vinculada a mejoras en el tratamiento de palabras cortas y abreviaciones, se logró mejorar en el total de datos totales. El porcentaje de acierto dentro de los datos totales obteniendo un **97,430%**, lo que nos ayuda a entender que las mejoras realizadas fueron un aporte para el algoritmo sin perder el porcentaje de acierto conseguido en el primer algoritmo.

5.2. Conclusiones

Se describió el corpus de texto libre, en donde está la información relevante sobre fichas clínicas. El corpus MADE 1.0 incluye información clasificada de pacientes anotada por expertos. Los datos están relacionados a sintomatología y efectos adversos a medicamentos. Además, se presentó en detalle el corpus, se analizó sus características y se desglosaron sus anotaciones según su clasificación.

Se analizó en profundidad la terminología MedDRA, se revisó en detalle el formato de codificación, las categorías que hay presente y que tan diversa es la terminología, además de presentar gráficamente su información clasificada.

Se desarrolló un algoritmo recodificador, basado en la biblioteca *Symspellpy*, que permitía la identificación de sintomatología, enfermedades y efectos adversos en el etiquetado de MADE 1.0, vinculándolo a la terminología de MedDRA, mediante la utilización de la distancia de Levenshtein.

Este algoritmo recodificador se presentó en 3 diferentes versiones, la primera es la más cruda de todas, aplicada de manera directa al corpus. La segunda ya con mejoras apuntando a la normalización, tanto del corpus como de la terminología. La última versión, tomaba lo hecho en la segunda versión, pero se mejoró con tratamientos especiales a palabras cortas y abreviaciones, además de integrar 2 niveles de búsqueda al utilizar la distancia máxima de Levenshtein (igual a “0” e igual a “1”).

Se recopilaron los resultados de las 3 versiones del algoritmo antes mencionada. Se logró analizar en detalle cada uno de los resultados arrojados por las diferentes versiones y como el algoritmo iba mejorando. Con esto se pudo documentar los alcances de este método de codificación en texto libre etiquetado. Además, en el algoritmo de normalización se logró captar el problema de que la gran mayoría de las abreviaciones las consideraba palabras cortas, lo que empeoraba resultados en todos los niveles máximos de Levenshtein utilizados, excepto en la distancia máxima de Levenshtein igual a 0. Para finalizar se obtuvo de los resultados que el combinar distancias máximas de Levenshtein 0 y 1 para la recodificación es lo más óptimo.

5.3. Trabajo Futuro

Existen muchos caminos en los que este trabajo puede ser continuado, primeramente, tenemos que el código puede ser mejorado, ya que, la presencia de factores que impiden que la distancia de Levenshtein sea útil a gran escala por sí sola, nos abre a pensar que el sumar algún otro concepto clasificador o sintetizador de texto libre, pueda ayudar a complementar los resultados con una mejora en la precisión o el alcance del algoritmo. El uso de inteligencia artificial podría ser una herramienta llamativa de cara a lo antes planteado. Con la gran capacidad de adaptabilidad a los problemas, la inteligencia artificial puede permitir ampliar la terminología de MedDRA o llegar a complementar las abreviaciones utilizadas en la expansión de estas, entre otras utilidades.

Ya orientándose más a la información manejada en este trabajo, seria relevante el poder vincular el resto de información presente en MADE 1.0, como los medicamentos etiquetados en el corpus, y entrecruzarlo con las entidades valiosas que ya fueron codificadas, así otorgándole mayor profundidad a la información.

El campo de la información médica y análisis sistemático de esta, es muy importante en otros países del mundo, por su capacidad de prevenir complicaciones médicas y complementa los cuidados médicos con una visión mucho más amplia sobre la farmacovigilancia. Esta información crucial puede ir desarrollándose mediante el análisis grandes poblaciones, lo que abre las puestas a

la extensión de algoritmos automatizados capaces de trabajar con cantidades masivas de información médica y con capacidades de entrecruzamiento de información encontrando tendencias poblacionales relevantes.

La barrera idiomática es algo que aún está presente y no puede omitirse. Este trabajo está basado en el idioma inglés (tanto el corpus como la terminología), por lo que, si este algoritmo quiere ser aplicado en Chile, por ejemplo, sería necesario contar con un corpus y una terminología en el idioma español. Esto presenta la necesidad de generar Corpus y terminologías seguras, confiables y lo más completas posibles, para poder desempeñar una codificación óptima.

Bibliografía

[1] H. Dalianis. "*Clinical Text Mining Secondary Use of Electronic Patient Records*". OAPEN Home. https://library.oapen.org/bitstream/handle/20.500.12657/27905/1002094.pdf?sequ (accedido el 25 de julio de 2022).

[2] "*MedDRA (Medical Dictionary for Regulatory Activities) - ProQuest*". ProQuest | Better research, better learning, better insights. https://www.proquest.com/openview/3b1a8c4832a3daf8d3bebe49f684708e/1?pq-origsite=gscholar&cbl=1096441 (accedido el 25 de julio de 2022).

[3] E. G. Brown, L. Wood y S. Wood. "*The Medical Dictionary for Regulatory Activities (MedDRA) - Drug Safety*". SpringerLink. https://link.springer.com/article/10.2165/00002018-199920020-00002 (accedido el 25 de julio de 2022).

[4] "*MedDRA Hierarchy*". MedDRA. https://www.meddra.org/how-to-use/basics/hierarchy (accedido el 25 de julio de 2022).

[5] C. Tonéatti, Y. Saïdi y V. Meiffrédy. "*Experience using MedDRA for global events coding in HIV clinical trials*". ScienceDirect. https://www.sciencedirect.com/science/article/pii/S1551714405001424 (accedido el 25 de julio de 2022).

[6] J. Rosenstock, V. Fonseca, J. B. McGill, M. Riddle, J. P. Hallé y J. P. Hallé I. Hramiak P. Johnston & M. Davis. "*Similar risk of malignancy with insulin glargine and neutral protamine Hagedorn (NPH) insulin in patients with type 2 diabetes: findings from a 5 year randomised, open-label study - Diabetologia*". SpringerLink. https://link.springer.com/article/10.1007/s00125-009-1452-2 (accedido el 25 de julio de 2022).

[7] R. Madrid Muñoz, C. Cano C y R. Cortés Rojas. "*Impacto de la extensión del postnatal en la adherencia a la lactancia materna. Estudio de Cohorte*". SciELO - Scientific electronic library online. https://www.scielo.cl/scielo.php?pid=S0370-41062018000400484&script=sci_arttext (accedido el 25 de julio de 2022).

[8] C. Méndez, M. Carrasco, B. Mora y M. Araya. "*Caracterización de la enfermedad celiaca en niños atendidos en hospitales públicos chilenos*". SciELO - Scientific electronic library online. https://www.scielo.cl/pdf/rcp/2018nahead/0370-4106-rcp-01001.pdf (accedido el 25 de julio de 2022).

[9] J. C. Gomez Rosado, M. Sanchez Ramirez, J. Valdes Hernandez, L. C. Capitan Morales, M. I. del Nozal Nalda y F. Oliva Mompean. "*Importancia de la calidad del informe de alta en la gestión de una unidad clínica quirúrgica*". ScienceDirect. https://www.sciencedirect.com/science/article/abs/pii/S0009739X12003910 (accedido el 25 de julio de 2022).

[10] "*Guía Introductoria para la Versión 25.0 de MedDRA*". Meddra. https://admin.meddra.org/sites/default/files/guidance/file/intguide_25_0_Spanish.pdf (accedido el 25 de julio de 2022).

[11] A. B. Chapman, K. S. Peterson, P. R. Alba, S. L. DuVall y O. V. Patterson. "*Detecting Adverse Drug Events with Rapidly Trained Classification Models*". PubMed Central (PMC). https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6373386/ (accedido el 25 de julio de 2022).

[12] L. Talamé, A. Cardoso y M. Amor. "*Comparación de herramientas de procesamiento de textos en español extraídos de una red social para Python*". 48 JAIIO. http://170.210.201.137/pdfs/asai/ASAI-06.pdf (accedido el 25 de julio de 2022).

[13] I. Amón y C. Jiménez. "*Funciones de Similitud sobre Cadenas de Texto: Una Comparación Basada en la Naturaleza de los Datos*". Repositorio Universidad Nacional. https://repositorio.unal.edu.co/bitstream/handle/unal/69915/71644758.20104.pdf (accedido el 25 de julio de 2022).

[14] wolfgarbe. "*GitHub - wolfgarbe/SymSpell: SymSpell: 1 million times faster spelling correction & fuzzy search through Symmetric Delete spelling correction algorithm*". GitHub. https://github.com/wolfgarbe/SymSpell (accedido el 25 de julio de 2022).

[15] D W. Garbe. "*Dictionary — symspellpy 6.7.6 documentation*". symspellpy: a SymSpell Python port — symspellpy 6.7.6

documentation. https://symspellpy.readthedocs.io/en/latest/examples/dictionary.html (accedido el 25 de julio de 2022).

[16] "*TIOBE Index - TIOBE*". TIOBE. https://www.tiobe.com/tiobe-index/ (accedido el 25 de julio de 2022).

[17] "*Anaconda | Anaconda Distribution*". Anaconda. https://www.anaconda.com/products/individual (accedido el 15 de marzo de 2022).

[18] "*Pandas documentation — Pandas 1.4.3 documentation*". pandas - Python Data Analysis Library. https://pandas.pydata.org/docs/ (accedido el 25 de julio de 2022).

[19] "*NumPy documentation — NumPy v1.23 Manual*". NumPy. https://numpy.org/doc/stable/ (accedido el 25 de julio de 2022).

[20] "*lxml - Processing XML and HTML with Python*". lxml - Processing XML and HTML with Python. https://lxml.de (accedido el 25 de julio de 2022).

[21] "*Matplotlib — Visualization with Python*". Matplotlib — Visualization with Python. https://matplotlib.org (accedido el 25 de julio de 2022).

[22] M. Skeppstedt, M. Kvist, G. H. Nilsson y H. Dalianis. "*Automatic recognition of disorders, findings, pharmaceuticals and body structures from clinical text: An annotation and machine learning study*". Research Gate. https://www.researchgate.net/publication/260031316_Automatic_recognition_of_disorders_findings_pharmaceuticals_and_body_structures_from_clinical_text_An_annotation_and_machine_learning_study (accedido el 25 de julio de 2022).

[23] D. Muñoyerro Muñiz, J. A. Goicoechea Salazar, F. J. García León, A. Laguna Téllez, D. Larrocha Mata y M. Cardero Rivas. "*Conexión de registros sanitarios: base poblacional de salud de Andalucía Conexión de registros sanitarios: base poblacional de salud de Andalucía*". SciELO - Salud Pública. https://www.scielosp.org/article/gs/2020.v34n2/105-113/es/ (accedido el 25 de julio de 2022).

[24] A. Miranda Escalada, A. Gonzalez Agirre, J. Armengol Estapé y M. Krallinger. "*Overview of automatic clinical coding: annotations, guidelines, and solutions for non-English clinical cases at CodiEsp track of CLEF eHealth 2020*". ceur-ws. http://ceur-ws.org/Vol-2696/paper_263.pdf (accedido el 26 de julio de 2022).

[25] G. López García, J. M. Jerez, N. Ribelles, E. Alba y F. J. Veredas. "*Transformers for Clinical Coding in Spanish*". IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/9430499 (accedido el 25 de julio de 2022).

UNIVERSIDAD DE CONCEPCION – FACULTAD DE INGENIERIA
RESUMEN DE MEMORIA DE TITULO

Departamento	:	Departamento de Ingeniería Eléctrica.
Carrera	:	Ingeniería Civil Biomédica.
Nombre del memorista	:	Bastián Mercado Ocares.
Título de la memoria	:	Codificación Automática de Efectos Adversos presentes en Texto Libre.
Fecha de la presentación oral	:	15 de Septiembre de 2022.

Profesor(es) Guía	:	Rosa Figueroa Iturrieta.
Profesor(es) Revisor(es)	:	Esteban Pino Quiroga.
Concepto	:	
Calificación	:	

Resumen (máximo 200 palabras)

La recodificación de términos presentes en MADE 1.0 se realiza utilizando la distancia de Levenshtein y con la implementación de 3 algoritmos de emparejamiento vinculados a la terminología de MedDRA, utilizando tanto términos preferidos como los términos asociados a éste. Este algoritmo recodificador se presentó en 3 diferentes versiones, donde la última versión se mejoró hasta lograr cerca de un 60% de codificación de los efectos adversos presentes en MADE 1.0, dentro de estos un 97,43% fue clasificado de manera correcta. Con esto se evidencia la potencia que posee la distancia de Levenshtein como codificador, acompañado de una adecuada terminología. Se trataron todos los objetivos planteados a lo largo de este trabajo, llegando a aborda tanto la terminología MedDRA como el corpus MADE 1.0, detallando sus características, como ambas afectaron en los resultados finales y como pueden ser mejoradas para futuros usos. Por otra parte, a pesar de las complicaciones antes mencionadas, el algoritmo final posee un porcentaje alto de términos bien clasificados en relación al total de términos que logro identificar en MADE 1.0, lo que nos permite ver que, a medida que el algoritmo se vaya mejorando, la positividad de clasificación esperada será similar o superior.