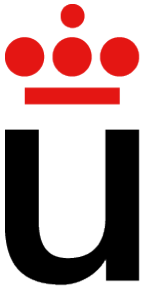

Universidad
Rey Juan Carlos

GRADO EN INGENIERÍA BIOMEDICA

Curso Académico 2022/2023

Trabajo Fin de Grado

IDENTIFICACIÓN Y DESAMBIGUACIÓN DE SIGLAS EN TEXTOS MÉDICOS MEDIANTE EL USO DE LA ARQUITECTURA TRANSFORMER

Autor : Elena Álvarez López

Tutor : David Roldán Álvarez

Trabajo Fin de Grado

Identificación y desambiguación de siglas en textos médicos mediante el uso de la arquitectura Transformer.

Autor : Elena Álvarez López

Tutor : David Roldán Álvarez

La defensa del presente Proyecto Fin de Carrera se realizó el día de de 202X, siendo calificada por el siguiente tribunal:

Presidente:

Secretario:

Vocal:

y habiendo obtenido la siguiente calificación:

Calificación:

Fuenlabrada, a de de 202X

Dedicado a

los que me han apoyado siempre,

los que están presentes y los que se han ido,

los que han confiado.

Gracias.

II

Agradecimientos

Este proyecto ha sido fruto del trabajo, la lucha, la constancia y la ambición. También, del apoyo recibido, ya que sin vosotros no hubiese sido posible.

Gracias a David, por apoyarme, ayudarme, guiarme y dirigir este proyecto de la mejor manera que existe. Por su constancia, trabajo y horas invertidas conmigo.

Gracias al equipo de ASHO, por darme esta increíble oportunidad, por enseñarme nuevos caminos y permitirime conocer más profundamente este campo.

Gracias a todos los profesores que han contribuido, de manera indirecta, al transmitirme sus conocimientos en diferentes ámbitos y permitirme llegar hasta aquí.

Gracias a mis compañeros de grado, especialmente a Andrea, Sara, Leticia, María y Miriam, que ya son familia, y como bien sabéis, sin vosotras esto no hubiese sido posible.

Gracias papá y mamá, por dejarme volar, experimentar, y llegar donde quiero. Por darme alas y apoyarme siempre en todo, por ser mi luz, mi camino y mi motivación.

Gracias a mis hermanos, por aguantar lo inaguantable, por ser la calma en la tempestad, y por su apoyo incondicional.

Gracias Isabel, Paola, Carla, Candela, María, Sara y Ángela, mis amigas de siempre, por ser las de siempre y seguir estando, por escucharme y valorarme, por admirar mi evolución, y por animarme a seguir creciendo.

Gracias José, por aportarme tu conocimiento, por animarme a seguir siempre, a no tirar la toalla, por ayudarme como el que más, y estar siempre.

Y gracias a ti, lector, por invertir tu tiempo en este trocito de mí.

Resumen

La desambiguación de acrónimos médicos es un desafío clave en el Procesamiento del Lenguaje Natural (PLN) en el dominio de la medicina. Los textos clínicos y de investigación presentan abundantes acrónimos médicos que pueden tener múltiples significados en función del contexto, lo cual dificulta la compresión precisa y la extracción de información relevante.

El objetivo de este Trabajo Fin de Grado es utilizar la arquitectura Transformer específicamente diseñada para abordar la desambiguación de acrónimos médicos en la lengua española. La arquitectura Transformer es un modelo de aprendizaje profundo basado en atención que ha demostrado excelentes resultados en tareas de PLN debido a su capacidad para generar relaciones de dependencia a lo largo de secuencias presentes en el texto. A diferencia del resto de modelos basados en Redes Neuronales Recurrentes (RNN), esta arquitectura no presenta limitaciones de longitud en las secuencias, por lo que tiene capacidad para llevar a cabo un procesamiento más eficiente utilizando mecanismos de atención, lo cual resulta especialmente beneficioso, ya que los acrónimos médicos están acompañados de un contexto más extenso que resulta fundamental para determinar su significado.

El modelo se entrenará utilizando un conjunto determinado de datos etiquetados que contenga acrónimos médicos junto con sus correspondientes expansiones, de manera que, el modelo aprenderá a mapear secuencias de texto que contienen acrónimos, a las expansiones correctas correspondientes en función del contexto. Posteriormente, se evaluará con un conjunto de prueba independiente.

Con este estudio, se espera contribuir al avance de la desambiguación de acrónimos en el contexto clínico y brindar herramientas efectivas para mejorar la compresión de estos. Además, se pretende mostrar las limitaciones encontradas en el desarrollo del proyecto y sentar las bases para futuras investigaciones en el desarrollo de enfoques más sofisticados y mejorados para la desambiguación de acrónimos médicos utilizando técnicas de aprendizaje profundo.

Summary

Disambiguation of medical acronyms is a key challenge in natural language processing in the domain of medicine. Clinical and research texts are abundant with medical acronyms that can have multiple meanings depending on the context, making it difficult to accurately understand and extract relevant information.

The objective of this Final Degree Project is to use the Transformer architecture specifically designed to address the disambiguation of medical acronyms in the Spanish language. The Transformer architecture is an attention-based deep learning model that has shown excellent results in Natural Language Processing (NLP) tasks due to its ability to generate dependency relationships across sequences present in text. Unlike the rest of the models based on Recurrent Neural Networks (RNN), this architecture does not present limitations on the length of the sequences, so it has the capacity to carry out more efficient processing using attention mechanisms, which is especially beneficial. since medical acronyms are accompanied by a more extensive context that is essential to determine their meaning.

The model will be trained using a given set of labeled data containing medical acronyms along with their corresponding expansions, so the model will learn to map text sequences containing acronyms to the correct corresponding expansions based on context. Subsequently, it will be evaluated with an independent test set.

With this study, it is expected to contribute to the advancement of the disambiguation of acronyms in the clinical context and provide effective tools to improve their understanding. In addition, it is intended to show the limitations found in the development of the project and lay the foundations for future research in the development of more sophisticated and improved approaches for the disambiguation of medical acronyms using deep learning techniques.

Índice general

Índice de figuras

Acrónimos

BERT Bidirectional Encoder Representations from Transformers

CSV Comma Separated Values

IA Inteligencia Artificial

MLM Masked Language Modeling

ML Machine Learning

NSP Next Sentence Prediction

PLN Procesamiento del Lenguaje Natural

RNN Redes Neuronales Recurrentes

SVM Support Vector Machine

TFG Trabajo Fin de Grado

TSV Tab-Separated Values

Capítulo 1

Introducción

1.1. Motivación

La incesante cantidad de datos clínicos junto con la mejora e incremento de las herramientas que permiten su análisis ha dado lugar a que puedan implementarse nuevos mecanismos que utilicen estos datos para mejorar los sistemas sanitarios. En los últimos años, el gran avance de la Inteligencia Artificial (IA) ha propiciado el estudio de como esta nueva tecnología puede aplicarse en el sector de la salud. De hecho, la IA ya se está utilizando en la medicina en múltiples áreas, entre las que se destacan: el diagnóstico y tratamiento de enfermedades, análisis de imágenes médicas y la investigación clínica. La función que desempeña la IA en investigación clínica es analítica. Permite reconocer patrones y relaciones ocultas en grandes conjuntos de datos clínicos, con el objetivo de mejorar la calidad de los datos clínicos.

Uno de los principales retos en este escenario es que la información clínica digital aparece desestructurada con frecuencia, y está escrita en lenguaje natural por los profesionales del sector, lo que impide el procesamiento automático de los datos. Es por este motivo, por el cual se necesita realizar un preprocesado del lenguaje para poder obtener información relevante contenida en los informes clínicos. Particularmente, en la documentación clínica, existe una gran cantidad de siglas que los médicos utilizan tanto para diagnósticos como procedimientos. No obstante, el uso de estas siglas conlleva aparejados problemas a la hora de la codificación clínica debido a la ambigüedad, que consiste en que un mismo acrónimo puede tener distintos significados dependiendo del contexto, como, por ejemplo: IR -Insuficiencia Renal, Insuficiencia Respiratoria. . . .

El objetivo de este proyecto es realizar un desambiguador de siglas, de manera que los datos clínicos proporcionados por los profesionales del sector puedan ser procesados con dicho modelo, y realizar una desambiguación sobre los acrónimos. Esto va a permitir realizar estudios de manera más efectiva, o la aplicación de otras herramientas de IA.

1.2. Machine Learning, Inteligencia Artificial y Procesado del Lenguaje Natural

El concepto de IA hace referencia al proceso en el que la inteligencia humana es replicada por máquinas, proporcionando a estas la habilidad de generar respuestas "humanas" y capacidad resolutiva de problemas mediante procesos de aprendizaje. Es decir, es una rama de la informática cuyo objetivo es crear sistemas y programas que pueden realizar tareas que requieran inteligencia humana, como la percepción, razonamiento, aprendizaje y toma de decisiones.

Un subcampo de la IA es el Machine Learning (ML) que define la capacidad de las máquinas de recibir datos y aprender por sí mismas mediante el ajuste de algoritmos, gracias al procesamiento de información. También, permite crear sistemas con capacidad de aprendizaje automático e independiente de intervención humana. Por lo tanto, el objetivo conjunto del ML y la IA, es trabajar para conseguir una toma de decisiones inteligentes, basadas en aprendizaje y experiencia, pero llevada a cabo por máquinas y aplicables a cualquier campo que requiera procesamiento y análisis de grandes cantidades de datos para obtener información y tomar decisiones más informadas [1].

Dado que la mayor cantidad de datos generados por humanos están escritos en lenguaje natural, la aplicación conjunta de ML junto IA, resulta de especial interés. En combinación con el Procesamiento del Lenguaje Natural (PLN), comúnmente NLP de sus siglas inglesas (Natural Language Processing), que es una rama de la IA que busca entender y procesar datos del lenguaje natural (humano), es posible implementar programas o algoritmos capaces de analizar grandes cantidades de datos en dicho lenguaje. Entre las tareas que realiza el PLN, están la traducción automática, el análisis de sentimientos, la generación de textos... [2]

En resumen, la aplicación de la IA, el ML y el PLN en el procesamiento de datos clínicos tiene el potencial de mejorar significativamente la atención médica y la eficiencia del sistema de salud. Su aplicación conjunta puede analizar grandes conjuntos de datos clínicos para de-

tectar patrones para el diagnóstico temprano de enfermedades, extraer información de las notas clínicas, las historias clínicas electrónicas y otros documentos médicos para apoyar la toma de decisiones clínicas.

1.3. Aplicación de la Inteligencia Artificial en medicina

La medicina es un campo que permite la aplicación y desarrollo de técnicas de PLN e IA. Concretamente, la IA ha tenido un impacto muy significativo en medicina y ha aportado numerosas contribuciones. Entre los ámbitos de contribución más destacados encontramos el diagnóstico médico, pronóstico y predición de resultados, descubrimiento de fármacos, cirugía asistida por robot o monitoreo y atención al paciente.

La IA ha demostrado ser efectiva en el diagnóstico de enfermedades mediante algoritmos que permiten la detección de patrones y anomalías en imágenes médicas con una precisión muy elevada, derivando en un mejora en la detección temprana de enfermedades y un diagnóstico más preciso. Un caso concreto puede verse en el artículo publicado en la revista de oftalmología [3], donde se demuestra la capacidad de detección temprana de signos en enfermedades oculares como la reinopatía diabética. También, como caso general, Hosein et al. demuestran la mejora de la precisión de la detección temprana de diversas enfermedades con radiología diagnóstica en su articulo [4].

La IA ha contribuido a la predicción y pronóstico de resultados ante diferentes tratamientos para una enfermedad. El análisis de grandes cantidades de datos clínicos, genómicos y moleculares con un algoritmo de IA permite la identificiación de patrones y factores de riesgo que predicen la respuesta ante un determinado tratamiento, así como la supervivencia de los pacientes. En el artículo publicado por Nazari et al. 2020 [5], los investigadores utilizaron datos clínicos y moleculares de pacientes con leucemia mieloidea aguda para desarrollar modelos de IA que predicen la respuesta al tratamiento.

Por otro lado, los textos clínicos son una fuente de información relevante para la práctica médica. Estos textos están caracterizados por: gran cantidad (historiales clínicos y artículos de medicina); constante crecimiento; alta fiabilidad, ya que deben estar escritos con rigurosidad lingüística; estructuración, son textos escritos en lenguaje natural pero guardan una estructura en la forma de incluir el contenido, facilitando su análisis; utilidad, debido a que los hospitales

presentan una predisposición muy alta para digitalizar sus historiales clínicos...

La precisión, es una de las características más relevantes del lenguaje médico; no obstante, existen muchos acrónimos, abreviaciones, terminología o siglas que dificultan su interpretabilidad, y que, además, han de ser reconocidas por el sistema. Es por este motivo, por el cual la desambiguación de acrónimos y siglas es fundamental para poder realizar una buena interpretación de estos.

La IA junto con PLN permite analizar y comprender estos textos para obtener información relevante, patrones, así como detección de diferentes estructuras de texto que resulten de interés. En este TFG, se propone utilizar las herramientas necesarias para obtener los diferentes acrónimos incluidos en la documentación clínica, y así facilitar su comprensión.

1.4. Estado del arte

Un desambiguador de siglas médicas es un sistema que utiliza técnicas de PLN y de IA para identificar y resolver ambigüedades en el uso de las siglas médicas en los textos clínicos. Los métodos utilizados por estos sistemas pueden ser de varios tipos.

En primer lugar, uno de los métodos más común de desambiguación son los **métodos basados en reglas**, que utilizan patrones de reglas predefinidos para identificar y definir siglas y acrónimos en el texto. Son muy simples y eficaces siempre y cuando los acrónimos del texto sean comunes y repetitivos. Por otro lado, los **métodos basados en diccionarios** utilizan un diccionario de siglas y acrónimos para identificar y definir términos en el texto. Son métodos muy precisos, pero están limitados a la cantidad y calidad del diccionario utilizado. Por último, están los **métodos basados en aprendizaje automático**, los cuales utilizan algoritmos de aprendizaje automático para analizar y clasificar los términos en el texto. Estos métodos tienen la capacidad de identificar siglas y acrónimos que no están presentes en los diccionarios tradicionales [6].

Para poder realizar la descomposición de un texto en lenguaje natural, se debe tener en cuenta tres pasos fundamentales: segmentación del texto, selección de información relevante y localización de los datos de interés (en el caso de un corrector de siglas, localizar las siglas presentes en el texto).

El proceso de PLN comienza con la recopilación y preparación de datos no estructurados de diversas fuentes. El preprocesamiento consiste en utilizar técnicas como la creación de tokens,

derivación, lematización y eliminación de palabras de parada, para preparar los datos para sus aplicaciones. La creación de tokens consiste en dividir una oración en unidades individuales de palabras o frases, y será la implementada en este proyecto. Por otro lado, la derivación y lematización consiste en simplificar palabras en función de su raíz, mientras que la eliminación de palabras de parada consiste en eliminar palabras que no aportan un significado a la oración [7].

Una vez generados los datos preprocesados, se requiere entrenar un modelo de PLN para que realice aplicaciones específicas basadas en la información textual proporcionada, que en este caso, consistirá en obtener el significado de un acrónimo o sigla.

La desambiguación de siglas médicas, a pesar de haber sido un tema de interés desde hace décadas, no siempre ha estado automatizada. De hecho, ha sido en los últimos años, cuando este campo ha experimentado un crecimiento exponencial y evolucionado a medida que la tecnología y los enfoques de procesamiento de lenguaje natural se han desarrollado.

La desambiguación de siglas requiere una tarea precedente que consiste en la detección de estas siglas en el texto, para posteriormente, llevar a cabo su desambiguación. A lo largo del tiempo, han ido surgiendo diferentes métodos para la detección de acrónimos en un texto, no obstante, todos presentan ventajas y limitaciones.

Los primeros métodos de detección de siglas se basaban en buscar coincidencias de los elementos del texto con un diccionario de siglas. El principal inconveniente de este método, y por lo que no puede ser implementado para abordar la tarea propuesta en este TFG, es la diversidad de siglas médicas existentes en la actualidad, así como su ambigüedad. Como ya se ha explicado anteriormente, en el ámbito médico, el contexto adquiere un papel muy relevante, ya que una misma sigla puede tener significados muy diferentes, en función del contexto en el que se encuentre.

Más adelante, y con el desarrollo de la IA, se introducen métodos de aprendizaje para la detección de siglas en un texto. Estos métodos pueden ser tanto supervisados, como no supervisados. La diferencia principal entre estos es que, en las técnicas de aprendizaje supervisado, se requiere un entrenamiento del conjunto de datos etiquetados, de manera que, en este caso, se está entrenando los datos marcando la posición en la que existe un acrónimo, para que el algoritmo lo aprenda, y luego pueda detectarlo. Sin embargo, en las técnicas de aprendizaje no supervisado, no se requiere un aprendizaje previo con un conjunto de entrenamiento, si no que

el algoritmo busca similitudes o patrones entre los datos. [8]

Retomando el problema de la multitud de siglas y acrónimos existentes, la aplicación de técnicas que requieran encontrar patrones entre estos puede conllevar un proceso largo y costoso, por lo que resulta menos frecuente. Resulta más eficiente poder realizar un entrenamiento con el conjunto de datos etiquetado, algunas de las técnicas de aprendizaje supervisado que se han aplicado para la detección de acrónimos pueden ser: Naive Bayes, arboles de decisión o SVM, de sus siglas en ingles Support Vector Machine.

En la actualidad, el modelo de Naive Bayes tiene más aplicación para la desambiguación de acrónimos, como se mostrará a continuación. Sin embargo, tanto los árboles de decisión como el método SVM, ya han sido utilizados para la detección de estos. Por ejemplo, en el trabajo de Ronzano y Furlong (2017) [9] sobre identificación de abreviación en la literatura biomédica, incluyen árboles de decisión para la identificación de abrevaciones medicas. Además, fueron Nadeau y Turney (2005) [10] en su artículo sobre aprendizaje supervisado en identificación de acrónimos, los promotores en incluir estas técnicas para el campo de detección de acrónimos.

Por otro lado, en el estudio de la evolución de la desambiguación de siglas médicas, podemos distinguir varias etapas. En las primeras etapas de la informática médica, la desambiguación de siglas médicas se basaba en reglas manuales. Los expertos en el dominio médico creaban conjuntos de reglas para determinar el significado correcto de las siglas en diferentes contextos clínicos.

Décadas más tarde, gracias a el desarrollo de las técnicas de minería de datos, los investigadores comenzaron a buscar patrones en los datos clínicos para ayudar a la desambiguación de siglas médicas. Estos patrones a menudo incluían información contextual, como el contexto en el que se usó la sigla y la presencia de otros términos relacionados.

Ya en las últimas décadas, con la aparición del aprendizaje automático, se propusieron alternativas de uso de este en el campo de la desambiguación de siglas médicas. Los enfoques de aprendizaje automático utilizan modelos estadísticos, principalmente supervisados, para aprender automáticamente patrones a partir de grandes conjuntos de datos etiquetados y luego aplicar esos patrones para desambiguar nuevas instancias de datos. Algunos casos de desambiguación de acrónimos ya desarrollados son, los llevados a cabo por Cuadros et al. (2018) [11] en su artículo de desambiguación de abreviaciones biomédicas con métodos de ML, donde se utiliza el modelo de aprendizaje supervisado de Naive Bayes, o el proyecto realizado por García

García, M. E. [12], entre otros. También, podemos destacar el trabajo de Pomares-Quimbaya et al. (2020) [13] dónde se describen modelos de aprendizaje supervisado para llevar a cabo la desambiguación.

Cómo técnica novedosa, en los últimos años, se han aplicado técnicas de redes neuronales a la desambiguación de siglas médicas. Estos enfoques utilizan redes neuronales artificiales para aprender patrones complejos en los datos, lo que les permite tener un mejor rendimiento en la desambiguación de siglas médicas que los enfoques de aprendizaje automático anteriores. Las redes neuronales son técnicas de Deep Learning o aprendizaje profundo, caracterizadas por generar conexiones entre diferentes neuronas y nodos, donde se manda información y se sacan conclusiones y predicciones determinadas a partir de los datos introducidos inicialmente. Existen diferentes tipos de redes neuronales, pero cuando trabajamos con datos secuenciales como frases de texto, generamos relaciones recurrentes dentro de cada capa, obteniendo las llamadas Redes Neuronales Recurrentes (RRN). En el trabajo de Kai et al. (2019), publicado en la revista Engineering Science and Technology Review [14], presenta el éxito de realizar una desambiguación de acrónimos médicos basándose en RNN y el nivel de carácter, es decir, carácter por carácter, y no por palabras, como los modelos tradicionales.

Por otro lado, en el artículo de Serguei Pakhomov et al. publicado en el National Center for Biotechnology Information [15], se demuestra como los métodos de desambiguación basados en el espacio vectorial son más efectivos que los métodos de clasificación estándar. En los modelos basados en el espacio vectorial, el contexto elegido se muestra como vectores cuyas componentes representan la existencia de los términos que presentan, es decir, cada componente representa un término cuyo valor indica la presencia o ausencia de un vector en un documento determinado.

Por último, Yonghui Wu et al. (2015) realizaron un estudio, publicado en School of Biomedical Informatics [16], dónde examina el uso de Word Embeddings para desambiguación clínica. Los modelos de Word Embeddigns consisten en representaciones de palabras influenciadas por la representación del resto de vectores en el conjunto de datos en el que se encuentran, es decir, dos palabras contextualmente similares, ocuparán dos vectores próximos en el espacio.

En la actualidad, el foco en la desambiguación de acrónimos médicos se encuentra en el desarrollo de técnicas basadas en redes neuronales recurrentes como las mencionadas anteriormente. En el artículo publicada por Cho et al. (2014) [17] se propone el uso de un modelo

codificación-decodificación (más comúnmente, encoder-decoder), a partir de dos RNN. La función principal del codificador es leer cada vector de entrada y acumular la información; mientras que el decodificador se encarga de predecir la salida a partir del estado final del codificador. No obstante, la limitación principal de este método se encuentra en la longitud de la secuencia, es decir, cuando se trabaja con secuencias cortas, es fácil almacenar toda la información de esta, pero cuando se encuentran secuencias de gran longitud, es posible que no se recupere toda la información de esta, debido a que estos métodos tienen memoria a corto plazo.

Esta limitación puede ser tratada haciendo uso de los mecanismos de atención, introducidos por Vaswani et al. (2017) en su artículo "Attention Is All You Need" [18] con una nueva arquitectura a la que denominaron Transformer, que permite a la red enfocarse en partes específicas de una secuencia de entrada, y por tanto procesar secuencias de gran longitud.

En general, todos los estudios actuales, sugieren que los enfoques basados en aprendizaje automático, en particular, los enfoques basados en RNN tienen un mejor rendimiento en la desambiguación de siglas médicas que los enfoques basados en reglas o conocimiento. Sin embargo, todavía existen desafíos importantes en la desambiguación de siglas médicas, como la falta de datos de entrenamiento etiquetados y la variabilidad en el uso de las siglas en diferentes contextos clínicos.

1.5. Estructura de la memoria

En esta sección se resume el contenido de cada uno de los capítulos que forman esta memoria, siguiendo la siguiente estructura.

- En el capítulo 1, **Introducción**, se incluye la introducción al TFG, dando contexto al marco en el que se encuadra este TFG y presentando trabajos realizados en este área.

- En el capítulo 2, **Objetivos**, se expone la finalidad de este TFG y los hitos que se pretenden alcanzar. Se incluye también la metodología y el plan de trabajo considerados en el desarrollo.

- En el capítulo 3, **Herramientas**, se detallan las diferentes herramientas y tecnologías utilizadas. Así como los lenguajes en los que se ha programado el proyecto.

- En el capítulo 4, **Diseño e Implementación**, se detalla la implementación de cada una de las partes desde el proyecto, así como la forma de puesta en marcha.

- En el capítulo 5, **Conclusiones**, se incluye una reflexión sobre las conclusiones y los conocimientos adquiridos en el desarrollo de este TFG.

Capítulo 2

Objetivos

Con este TFG, se espera contribuir al avance de la desambiguación de acrónimos en el contexto clínico y brindar herramientas efectivas para mejorar la compresión de los textos clínicos.

2.1. Objetivos

El objetivo de este TFG es desarrollar un sistema de desambigüación de acrónimos médicos que permita identificar el significado correcto de un acrónimo en un contexto médico determinado. Este sistema tiene como finalidad la mejora de la comunicación entre los profesionales de la salud para garantizar la seguridad del paciente, evitando posibles confusiones que puedan derivar en errores médicos.

2.1.1. Objetivo general

Los objetivos generales de este trabajo son fundamentalmente dos:

1. Desarrollo de un sistema de desambiguación de acrónimos médicos que permita la obtención del significado correcto de un acrónimo médico en un determinado contexto.

2. Evaluación de la eficacia y precisión de dicho sistema con diferentes métodos y pruebas en comparación con sistemas similares.

2.1.2. Objetivos específicos

En el caso de este desambiguador de acrónimos médicos, algunos objetivos específicos que se han llevado a cabo para alcanzar los objetivos generales son:

1. Análisis y elección de diferentes fuentes de datos médicos sobre las que se aplicará el sistema de desambiguación.

2. Estudio y evaluación de algoritmos que podrían ser implementados para el desarrollo del proyecto.

3. Selección de algoritmos, técnicas, modelos, lenguajes y metodología que se seguirá para el desarrollo del sistema de desambiguación.

4. Diseño e implementación del modelo de desambiguación de acrónimos médicos.

5. Entrenamiento del modelo con diferentes datos para mejorar precisión y eficacia.

6. Realización de pruebas de evaluación en diferentes ámbitos y contextos médicos para llevar a cabo posibles mejoras sobre el modelo realizado.

7. Comparación de los resultados obtenidos con otros sistemas de desambiguación de acrónimos médicos ya implementados y determinar así su efectividad.

2.2. Plan de trabajo

El plan de trabajo temporal que se ha seguido en el desarrollo de este TFG para lograr objetivos específicos, y así poder alcanzar los objetivos generales ha sido:

- Revisión bibliográfica de las diferentes técnicas y algoritmos utilizados en sistemas de desambiguación de acrónimos médicos ya implementados.

- Creación de un corpus de datos clínicos una vez seleccionada la fuente de datos para el desarrollo del sistema.

- Análisis y procesamiento de los datos para seleccionar los datos de interés para el desarrollo del sistema.

- Planificación del desarrollo del modelo que se va a utilizar para la implementación del desambiguador.
- Diseño e implementación del modelo, junto con el entrenamiento y evaluación requeridos.
- Comparación de resultados con modelos ya existentes y planificación de posibles mejoras

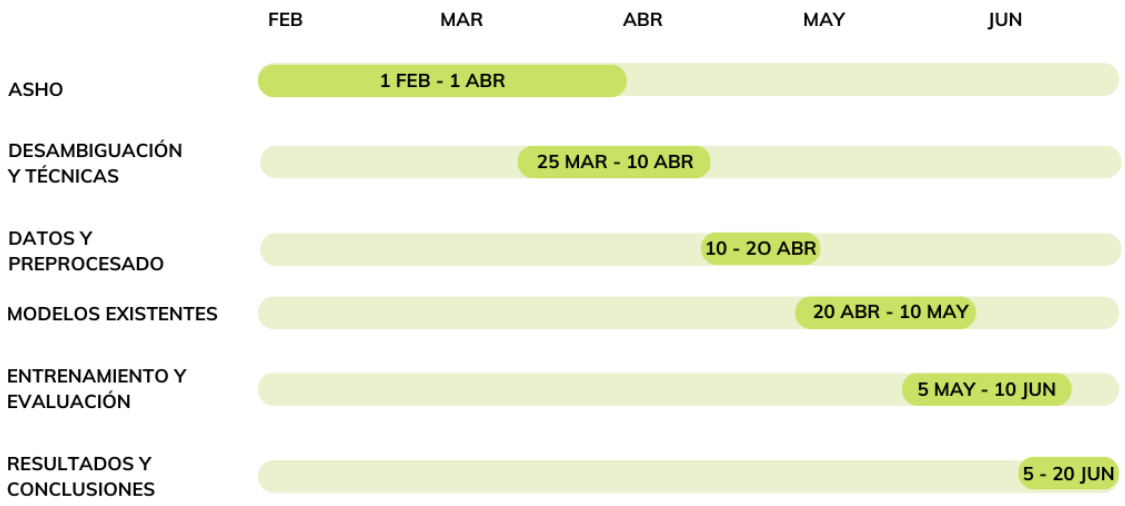

Figura 2.1: Diagrama de Gantt del desarrollo del TFG

Más concretamente, tal y como se muestra en el diagrama de gantt 2.1 se pueden distinguir 10 etapas marcadas en el desarrollo de este proyecto. En primer lugar, la idea nació en colaboración con la empresa ASHO en la que tuve la oportunidad de estar colaborando durante un periodo de tiempo, y dónde entré en contacto con la IA en el ámbito de datos clínicos. Más adelante, me interesé por el conflicto existente con las siglas médicas en textos clínicos y su necesidad de desambiguación, lo que me derivó a investigar técnicas existentes para llevar a cabo este proceso. Fue entonces cuando me decidí a intentar generar un nuevo proyecto que realizase esta tarea, y me encargué de obtener, analizar y preprocesar grandes cantidades de datos clínicos para probar los diferentes modelos investigados. Finalmente, me decidí por entrenar estos datos y evaluarlos sobre el modelo más eficiente, así como compararlo con el resto, y proponer líneas futuras para este TFG.

Capítulo 3

Herramientas

En el contexto de la investigación sobre desambiguación de acrónimos médicos, es fundamental la utilización de herramientas especializadas para poder abordar la complejidad y ambigüedad de este campo. En esta sección, se indican las herramientas más destacadas que han sido implementadas, examinando sus características, funcionamiento, ventajas y desventajas para abordar el problema expuesto en este TFG.

3.1. Transformer

Un Transformer es una arquitectura de red neuronal que se ha convertido en una de las principales herramientas para el procesamiento de texto en el campo del PLN. La arquitectura Transformer está caracterizada por su memoria a muy largo plazo gracias al mecanismo de atención y el procesamiento de la secuencia completa en paralelo. Fué introducido por primera vez en el artículo "Attention is All You Need" de Vaswani et al. en el año 2017 [18], y desde entonces, ha resultado de mucho interés en diferentes ámbitos como el modelado del lenguaje, traducción automática o generación de texto.

Esta arquitectura está basada en el uso de la técnica "atención", que permite a la red enfocarse en partes específicas de una secuencia de entrada. Además, el mecanismo de acción de un Transformer consiste en, a partir de una secuencia de entrada, se convierte a representación numérica en una capa de embedding, a la cual se añadirá la posición y entrará a una etapa de codificación que permite obtener la información relevante de la secuencia, que conectará con el decodificador, el cual recibe la salida del codificador junto con la salida del propio decodifica-

dor en el paso anterior, para generar una secuencia de salida. Este proceso, se puede observar graficamente en la figura 3.1

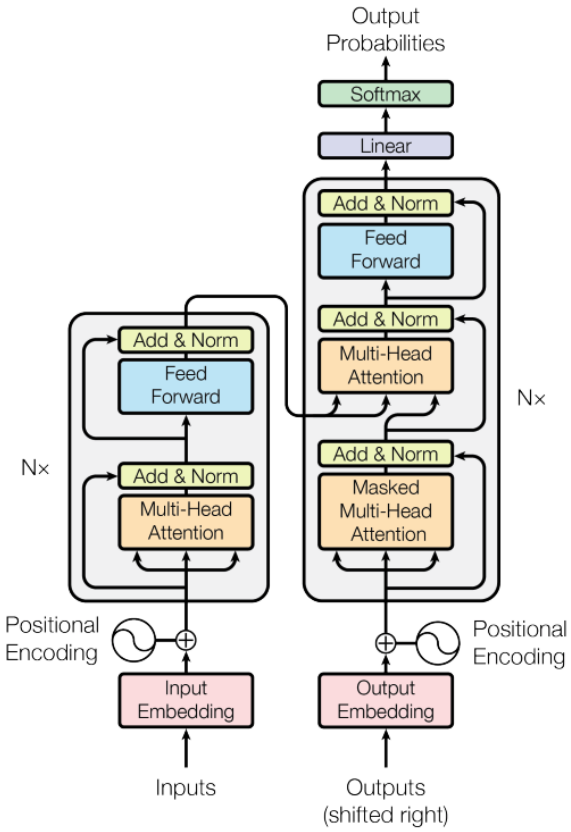

Figura 3.1: Arquitectura Transformer referida en Vaswani et al.2017

El encoder (codificador) del Transformer está compuesto por 6 capas. Cada una de estas capas, a su vez está formada por una capa de multi-head attention para los mecanismos de atención y otra de feed forward, que es una red implementada hacia delante. Estas capas están conectadas mediante conexiones residuales y una capa de normalización, como se puede apreciar en la figura 3.2. Los elementos de secuencia que entran al codificador son embeddings.

Por otro lado, el decoder (decodificador) representado en la figura 3.3 también está compuesto por 6 capas. Cada una de estas capas están implementadas y conectadas del mismo modo que el codificador, con una capa de multi-head attention para los mecanismos de atención y otra de feed forward, a diferencia del codificador, en el decodificador se añade una tercera capa de multi-head attention que recibe tanto la información procedente del propio decodificador, como la procedente del codificador.

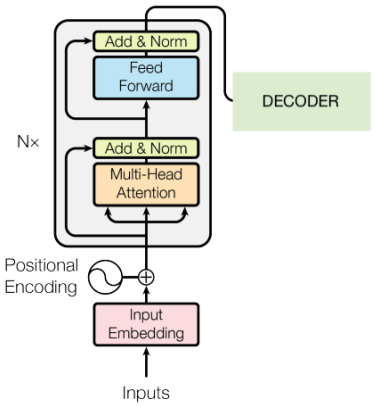

Figura 3.2: Arquitectura encoder

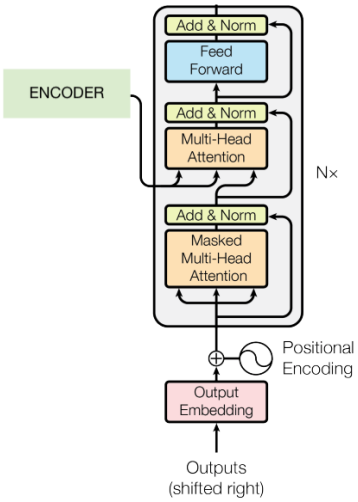

Figura 3.3: Arquitectura decoder

Los elementos fundamentes de un Transformer que permiten conformar el codificador y el decodificador son: el bloque embedding y codificación de posición, bloque atencional, bloque residual, red neuronal, el bloque de codificación y el bloque de decodificación.

Bloque embedding y codificación de posición. En este bloque, se convierte el texto en tokens, que son representaciones numéricas que pueden ser entendidas por la red. La codificación de posición permite añadir la posición de los tokens dentro de una secuencia. Para esta codificación se utilizan funciones senoidales para las posiciones pares (3.1) y cosenoidales para las posiciones impares (3.2).

$$\mathrm{PE}(\mathrm{pos}, 2i) = \sin\left(\frac{\mathrm{pos}}{10000^{\left(\frac{2i}{\mathrm{dmodel}}\right)}}\right) \tag{3.1}$$

$$\mathrm{PE}(\mathrm{pos}, 2i+1) = \cos\left(\frac{\mathrm{pos}}{10000^{\left(\frac{2i}{\mathrm{dmodel}}\right)}}\right) \tag{3.2}$$

De esta manera, se obtendrán los positional encodings, formados por los embeddings y la información posicional, utilizados a la entrada del codificador y decodificador.

Bloque atencional. Se encarga de analizar la totalidad de la secuencia de entrada y encontrar relaciones entre palabras de la secuencia. Permite expresar numéricamente las relaciones que existen en la secuencia a diferentes niveles, para codificar cada una de ellas con la información del contexto y saber qué elementos del texto requieren una mayor atención. Para poder realizar este bloque, se requieren 3 vectores obtenidos a partir de los tokens iniciales: queries, keys y values (son representaciones alternativas de los tokens iniciales). En primer lugar, se toma el vector query de cada token, y se compara con cada uno de los vectores key existentes. Esta comparación es una multiplicación de vectores que devolverá como resultado una puntuación para medir el grado de similitud entre pares de palabras y el grado de atención que requieren (a mayor puntuación, más atención). Una vez obtenida la puntuación resultante de la multiplicación de query y key, se ponderará cada uno de los vetores values, indicando la importancia de cada palabra. Para esto, debemos escalar las puntuaciones obtenidas y se multiplicará dicha matriz resultante de escalar las puntuaciones por la matriz que contiene los vectores value, de manera que, se obtendrán 4 tokens que contendrán la información de contexto más relevante para cada palabra de cada secuencia. Por lo tanto, el bloque atencional, toma los tokens iniciales y codifica los tokens resultantes como los elementos de la secuencia que tienen más relevancia. Se necesitan múltiples bloques atencionales para poder identificar asociaciones entre palabras y grupos de palabras a diferentes niveles. La salida de estos bloques se combina en una última red neuronal que combina toda la información resultante en un único vector por cada bloque de entrada 3.4.

Bloque residual. Recibe tanto la entrada como la salida del bloque atencional para evitar la pérdida de información por la profundidad característica de la red. Se encarga de sumar los datos de entrada y salida, y normalizarlos para que tengan la escala adecuada para el siguiente bloque.

Red neuronal. Procesa en paralelo todos los vectores de la secuencia, tomando la información atencional de las capas anteriores y consolidándola en una única representación.

Bloque de codificación. Constituido por varios elementos: bloque atencional, bloque resi-

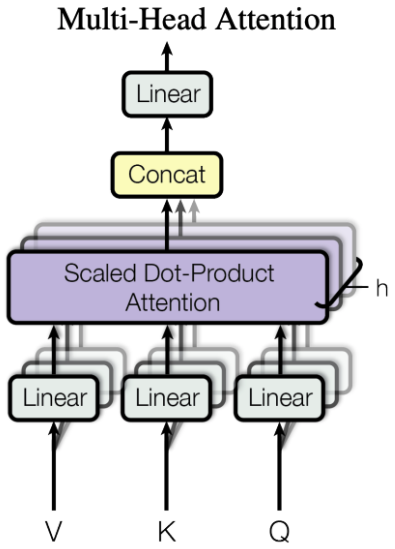

Figura 3.4: Bloque de atención: Multi Head Attention

dual y red neuronal

Bloque de decodificación. Constituido por: bloque atencional, bloque residual y red neuronal

Salida. La etapa final del Transformer consta de una capa lineal seguida de una capa softmax. Estas son necesarias debido a que el decodificador devuelve un vector numérico que requiere de estas capas para su decodificación. La capa lineal es una capa fully-conected (completamente conectada) que proyecta el vector salida del decodificador con un vector de neuronas, cada una con una determinada puntuación. La probabilidad de esta puntuación es asignada con la capa softmax, de manera que se elegirá aquella neurona con mayor probabilidad.

La diferencia principal entre el codificador y el decodificador es que el decodificador recibe la salida del codificador e implementa una capa de unión que los conecta (codificador y decodificador). Además, dado que los Transformer están caracterizados por ser modelos autorregresivos, requieren la implementación de una capa que permita que la salida para cada frase sólo dependa de los elementos de la capa anterior, son modelos causales.

3.1.1. BERT

El PLN consigue descifrar el significado de palabras dentro de un contexto. En esencia, el PLN es una combinación de informática y lingüística. La arquitectura Transformer, así como las

anteriores a esta, presenta el inconveniente de que es bidireccional (no direccional), ya que, sólo entrena hacia delante y lee la secuencia entera de palabras una sola vez (no secuencialmente), por lo tanto, sólo puede predecir las siguientes palabras en función de las anteriores. Para resolver este problema surge BERT (Bidirectional Encoder Representations from Transformers) que utiliza codificadores que aplican un entrenamiento bidireccional.

BERT es un método de preentrenamiento de representaciones de lenguaje, es decir, se entrena un modelo de comprensión del lenguaje con un propósito general, sobre un corpus grande de texto sin fromato que después será útil para realizar las tereas requeridas de PLN.

Entre los modelos actuales preentrenados de BERT podemos distinguir 3 principalmente: BERT-Base, BERT-Large y BERT-Multilingüe. A su vez, estos modelos han sido entrenados para texto en minúsucula completamente, o manteniendo las mayúsculas: BERT-Uncased, BERT-Cased, respectivamente. Las caracterísiticas de estos modelos se encuentran detalladas en el artículo publicado por Devlin et al. en 2017 [19] y estan recogidas en la tabla 3.5.

BERT-Base - Cuenta con 12 capas de codificación, 12 head attention, 786 capas ocultas y 110 millones de parámetros.

BERT-Large - Cuenta con 24 capas de codificación, 16 head attention, 1024 capas ocultas y 340 millones de parámetros.

BERT-Multilingüe - Cueta con 104 idiomas, 12 capas, 12 head-attention, 768 capas ocultas y 110 millones de parámetros.

MODELO	CAPAS DE CODIFICACIÓN	CABEZAS DE ATENCIÓN	CAPAS OCULTAS	PARÁMTROS (MILLONES)
$\text{BERT}_{\text{BASE}}$	12	12	768	110
$\text{BERT}_{\text{LARGE}}$	24	16	1024	340
$\text{BERT}_{\text{MULTILINGÜE}}$	12	12	768	110

Figura 3.5: Modelos preentrenados BERT: BERT-Base, BERT-Large y BERT-Multilingüe

Los modelos BERT más pequeños están destinados a entornos con recursos computacionales restringidos y se pueden ajustar de la misma manera que los modelos BERT originales. Por otro lado, los modelos más grandes, requieren un mayor coste computacional y están destinados a labores más específicas.

Biblioteca BERT

BERT es una biblioteca de código abierto creada en 2018 en Google. Es una técnica nueva para PNL y adopta un enfoque de modelos de entrenamiento completamente diferente al de cualquier otra técnica.

BERT es un acrónimo de Representaciones de codificador bidireccional de Transformer. Eso significa que, a diferencia de la mayoría de las técnicas que analizan oraciones de izquierda a derecha o de derecha a izquierda, BERT va en ambas direcciones usando el codificador Transformer. Esta característica permite al modelo aprender el contexto de una palabra basándose en todo su entorno (izquierda y derecha de la palabra). En su forma simple, Transformer incluye dos mecanismos separados: un codificador que lee el texto de entrada y un decodificador que produce una predicción para la tarea. Dado que el objetivo de BERT es generar un modelo lingüístico, sólo es necesario el mecanismo codificador.

Su objetivo es generar un modelo de lenguaje. Por lo tanto, proporciona una forma de preentrenar con mayor precisión que los modelos con menos datos. El enfoque bidireccional que utiliza significa que obtiene más contexto para una palabra que si solo estuviera entrenando en una dirección. Con este contexto adicional, se puede determinar que esta arquitectura está basada en dos técnicas principalmente: Masked-Language Modeling (MLM) y Next Sentence Prediction (NSP).

Masked-Language Modeling

En el codificador del Transofmer, se lee la secuencia entera, por tanto, la predicción de la siguiente palabra podría estar afectada por la que se ha leído previamente. BERT utiliza MLM como estrategia de entrenamiento, que consiste en enmascarar al azar el 15 % de las palabras en una oración con un símbolo o token [MASK] y luego tratar de predecirlas basándose en las palabras que rodean a la palabra enmascarada. Esto es completamente diferente de cualquier otro modelo de lenguaje existente porque mira las palabras antes y después de una palabra enmascarada, al mismo tiempo. El modelo entonces intentará predecir el valor original de estas palabras enmascaradas basándose en el contexto extraído por las palabras que la rodean. Gran parte de la precisión que tiene BERT se puede atribuir a esto.

Next Sentence Prediction (NSP)

Para poder entender la relación entre frases de palabras, se sigue un proceso basado en la insercción de tokens [CLS] que marcan el inicio de la frase y token [SEP] que marca fin. A cada token se le añade un embedding que indica a que frase pertenece y un vector de posición que permite conocer su posición exacta en la frase.

Para poder implementar este método de entrenamiento y con el objetivo de calcular la probabilidad de que sea la segunda secuencia la siguiente frase respecto la anterior, deberá haber **incrustaciones de tokens (token embeddings)** para marcar el principio y el final de las oraciones. Además, **incrustaciones de segmentos (segment embeddings)** para poder distinguir diferentes oraciones, y por último, necesitará **incrustaciones posicionales (positional embeddings)** para indicar la posición de las palabras en una oración. Un ejemplo de esta representación puede apreciarse en la figura 3.6.

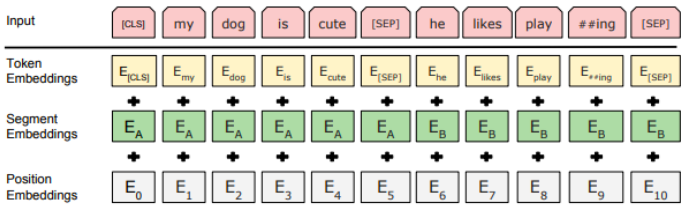

Figura 3.6: BERT embeddings input

BETO

Dado que BERT ha sido implementado en lengua inglesa principalmente, para poder utilizarlo en PLN en español, nace BETO. BETO es un modelo de BERT entrenado con un corpus en español mediante la técnica de enmascaramiento descrita anteriormente.

Del mismo modo que para los modelos preentrenados de BERT, se distinguen principalemnte dos modelos preentrenados de BETO: BETO-Cased y BETO-Uncased.

3.2. Python

Python es un lenguaje de programación interpretado y multiparadigma, es decir, soporta más de un modelo de desarrollo entre los que destacamos: imperativo, funcional y orientado a

objetos. Además, es de tipado dinámico, multiplataforma y multipropósito.

A continuación, se va a explicar cada una de las características de este lenguaje.

Interpretado. Los lenguajes de programación pueden clasificarse en interpretados o compilados según la forma en la que se traducen. Los lenguajes interpretados son aquellos en los que el código es traducido por un intérprete de forma instantánea; mientras que los compilados requieren una traducción completa del código en un proceso de compilación. La ventaja que presentan los lenguajes interpretados es que no requieren procesos de compilación y por tanto se ahorra una gran cantidad de tiempo, y además el código fuente puede ser ejecutado en cualquier software siempre que este incluya el intérprete.

Multiparadigma. Un lenguaje multiparadigma se refiere a aquel que soporta diferentes modelos de desarrollo. En el caso de Python, se distinguen tres modelos diferentes: imperativo, funcional y orientado a objetos.

Imperativo. Los lenguajes de programación se clasifican en imperativo o declarativo, siendo los imperativos aquellos que describen el estado del programa y permiten su modificación con condiciones o código; mientras que los declarativos, sólo declaran condiciones que describen un problema y detallan su solución. Es decir, los imperativos permiten llevar a cabo la solución a un determinado problema, mientras que los declarativos solo pueden describirlo.

Funcional. Es un lenguaje que permite incluir funciones, de manera que, se permite la variación del programa mediante la mutación de variables. La principal ventaja de esta característica es que, operar con funciones recursivas ahorrará gran cantidad de código.

Orientado a objetos. La programación orientada a objetos permite una mejor organización del código, ya que, el objeto es una entidad que tiene un determinado estado, y a su vez las entidades son propiedades características del objeto.

De tipado dinámico. Se dice que un lenguaje es de tipado dinámico, cuando la variable puede tomar diferentes valores de distintos tipos y en diferentes momentos; es decir, las variables son declaradas en función de su contenido. Esto va a presentar como ventaja que una determinada variable puede cambiar el valor y tipo durante la ejecución sin necesidad de volver a ser declarada.

Python ha sido una herramienta muy útil en la elaboración de todo este proyecto por su amplia operabilidad en diferentes ámbitos.

En cuanto al procesado de los datos, ha permitido desempeñar tareas con unas simples líneas

de código como: cambiar el nombre de una gran cantidad de archivos a la vez, convertir un archivo en otro tipo de archivo, eliminar palabras duplicadas de un archivo de texto, llevar a cabo operaciones matemáticas básicas, descargar contenido, efectuar análisis básicos de registros, encontrar errores en varios archivos

Por otro lado, también permite la obtención de información básica de los datos como, por ejemplo: tareas de limpieza de datos, es decir, corregir y eliminar datos incorrectosy extraer y seleccionar varias características de los datos

Además, Python es un lenguaje que contiene múltiples bibliotecas. Una biblioteca es una colección de códigos usados con frecuencia que los desarrolladores pueden incluir en sus programas de Python para evitar tener que escribir el código desde cero. De forma predeterminada, Python incluye la biblioteca estándar, que contiene una gran cantidad de funciones reutilizables; además, más de 137 000 bibliotecas disponibles para diversas aplicaciones, incluidos el desarrollo web, la ciencia de datos y el ML.

3.2.1. Numpy

Entre las bibliotecas de Python más populares, y que han sido utilizadas para el desarrollo de este TFG, encontramos la biblioteca Numpy, que permite crear y administrar matrices, manipular formas lógicas y efectuar operaciones de álgebra lineal. Entre las características más ventajosas de Numpy destacan la capacidad de trabajar con datos multidimensionales, la diversidad de funciones matemáticas que permite implementar y la facilidad de integración con otras librerías.

3.2.2. Keras

La librería Keras, es otra de las bibliotecas más populares de Python, conocida como la biblioteca de red neuronal profunda, que cuenta con un excelente soporte para el procesamiento de datos. Proporciona una interfaz simple y fácil de usar para la creación y entrenamiento de modelos de redes neuronales, por lo que se utiliza frecuentemente en el desarrollo de modelos de aprendizaje profundo. La característica más relevante de esta librería y el motivo principal de implementación en este TFG, es su modularidad y flexibilidad, es decir, permite construir y combinar fácilmente diferentes capas y modelos para crear arquitecturas de redes neuronales

personalizadas. Además, cuenta con una amplia gama de capas y funciones ya implementadas para el desarrollo de la red como, por ejemplo, capas de convolución, capas recurrentes y funciones de activación, optimización, pérdida y evaluación.

3.2.3. NLTK

NLTK, de sus siglas, Natural Language Toolkit, es una biblioteca de Python para el PLN. Proporciona herramientas para tareas como tokenización, el preprocesamiento, la desambiguación, etiquetado y análisis de sentimientos. Gracias a estas herramientas, hemos podido implementar de manera mas eficiente alguna de las tareas de este proyecto, como el preprocesamiento del texto y su tokenización.

3.2.4. Re

La librería re de Python, de su acrónimo, "Expresiones Regulares", permite buscar patrones en el texto. Las expresiones regulares son patrones de coincidencia de texto descritos con una sintaxis formal. A su vez, los patrones se interpretan como un conjunto de instrucciones, que luego se ejecutan con una cadena como entrada para producir un subconjunto de coincidencia o una versión modificada del original.

Esta biblioteca resulta especialmente útil para este TFG debido a que gran parte de los acrónimos, por lo general son expresiones regulares formadas por una sucesión de letras en mayúscula, de manera que, buscando como patrón aquellas palabras formadas por una sucesión de dos o más letras en mayúscula, se puede implementar esta librería como método para identificar las posiciones de los acrónimos presentes en un texto.

3.2.5. TensorFlow

TensorFlow es una biblioteca de código abierto para la computación numérica y Machine Leraning a gran escala. El funcionamiento básico de TensorFlow consiste en un grafo computacional, donde los nodos representan operaciones matemáticas y cada conexión o arista entre nodos representa los datos multidimensionales (tensores) que fluyen entre esas operaciones. Esto permite la construcción de modelos de aprendizaje automático de manera eficiente y escalable.

La principal ventaja que se ha aprovechado de TensorFlow para desarrollar este TFG es la capacidad de construcción, entrenamiento e implementación de modelos. TensorFlow proporciona herramientas para exportar e importar modelos entrenados y utilizarlos en diferentes ámbitos y con diferentes datos.

3.2.6. Pandas

Pandas es una biblioteca de análisis de datos de código abierto escrita en Python. Proporciona estructuras de datos y herramientas de manipulación de datos de alto rendimiento y fáciles de usar, por lo que es una herramienta esencial en el análisis y manipulación de datos en Python.

Esta librería está caracterizada principalmente por su capacidad de definir estructuras de datos basadas en arrays de NumPy, leer y escribir ficheros en diferentes formatos, o acceder a datos mediante índices o nombre de filas y columnas; además, trabaja con estructuras de datos diferentes, series y DataFrame.

Uno de los motivos por los que se ha implementado esta librería, son las funciones que incluye para la manipulación de datos, que permiten la selección, limpieza y filtrado de datos, operaciones lógicas y matemáticas con columnas, agrupación y cálculo de agregaciones como sumas, promedios, máximos y mínimos. Otro de los motivos, es su capacidad de análisis exploratorio de datos. Pandas ofrece herramientas que permiten realizar estadísticas descriptivas de los datos, y así obtener la media, mediana, desviación típica...

Además, Pandas tiene una alta capacidad de integración con otras bibliotecas como NumPy y TensorFlow, ya mencionadas previamente.

3.2.7. PyTorch

PyTorch es una biblioteca de código abierto que al igual que Tensorflow está especializada en diferenciación automática para proyectos de aprendizaje automático, y cálculos de tensor y acelereación de GPU, es decir, principalmente para funciones de ML a gran escala.

Al igual que TensorFlow, PyTorch destaca por su capacidad para crear, entrenar y cargar modelos, pero a diferencia del anterior, utiliza un gráfico computacional dinámico, por lo que el gráfico se modifica durante la ejecución, permitiendo un mayor nivel de flexibilidad y facilidad de depuración.

3.3. Entorno de edicción

Un entorno de edición es un programa o conjunto de programas que engloban todas las tareas necesarias para el desarrollo de un programa o aplicación. Entre las funciones que tiene que cumplir un entorno de programación encontramos: edicción de programa, compilación, ejecución y depuración.

3.3.1. Jupyter Notebook

El primer entorno utilizado para el desarrollo de este TFG fue Jupyter Notebook, un entorno de desarrollo interactivo y de código abierto que permite la creación y compartición de documentos que contienen código, visualizaciones, texto explicativo y otros elementos.

El motivo de implementación en este entorno fue principalmente por su flexibilidad y capacidad para integrar código y visualizaciones. Además, en los archivos de Jupyter, denominados notebooks, se puede escribir y ejecutar código en celdas individuales, lo cual resulta muy eficiente para poder depurar grandes programas.

Por otro lado, la principal limitación que se encontró fue la memoria. Las limitaciones de memoria afectaron al rendimiento y ejecución del código, ya que, al trabajar con grandes conjuntos de datos, la carga requiere grandes recursos que superan en muchos casos la memoria disponible, derivando a la interrupción de la ejecución. Además, cada celda de código ocupa una determinada memoria en el entorno de ejecución, de manera que, al ejecutar varias celdas que procesen grandes cantidades de datos, es común el agotamiento de la memoria disponible. Dado que, en este TFG se han procesado corpus con muchos datos, y entrenado diferentes modelos, este entorno no satisfacía las necesidades, ya que impedía la ejecución completa del programa.

3.3.2. Google Colaboratory

Como alternativa a Jupyter Notebook, se planteó la posibilidad de ejecutar este código en la nube, con el objetivo de solventar los problemas de memoria ya mencionados. Google Colaboartory es un entorno de desarrollo en la nube que proporciona acceso a recursos informáticos y permite ejecutar notebooks de Jupyter; motivo principal de la selección de este método. Con este método, sí que se consiguió llevar a cabo gran parte del preprocesado de los datos; no obs-

tante, las principales limitaciones que se encontraron fueron: el espacio disponible en Google Drive, la nube en la que se almacenan todos los archivos de Google, y el tiempo de ejecución.

En primer lugar, los archivos que contenían los datos para entrenar el modelo pesaban bastante, lo cual dificulta la posibilidad de almacenarlos en la nube de Google. Además, Google Colaboratory cuenta con un tiempo de ejecución máximo por sesión, generalmente de 12 horas, de manera que, después de ese tiempo, la sesión se reinicia y se pierden todos los datos almacenados en la memoria. Sin embargo, pese a estas limitaciones, este entorno si permitió reslizar el entrenamiento del modelo, y ejecucción completa del programa.

3.4. Fuentes de datos

3.4.1. CSV

Un archivo CSV, de su abreviatura Comma Separated Values (valores separados por comas), es una herramienta de texto plano utilizada para almacenar y organizar datos. El formato de archivo se representa mediante una tabla divida por comas generalmente, aunque también, pueden utilizarse como separadores otros caracteres como, punto y coma o tabulaciones. Los archivos de texto diseñados en formato CSV organizan su información en filas y columnas, de manera que, cada columna corresponde a un encabezado que define la categoría en la que se encuentra un dato específico; mientras que las filas, son cada línea de la tabla y conforman un bloque de información.

La principal ventaja que presenta el formato CSV es su interoperabilidad y su carácter multiplataforma, debido a su alta compatibilidad, estos archivos pueden ser utilizados entre múltiples plataformas. Además, son bastante flexibles ya que carecen de restricciones de estructura y permiten incluir cualquier tipo de texto. También son muy ligeros, ocupando un tamaño de almacenamiento generalmente menor que el resto de formatos. Por estas características, los archivos CSV aparecen con frecuencia como herramienta en el ámbito de la ciencia de datos.

3.4.2. TSV

El formato TSV (Tab-Separated Values) es un tipo de formato de archivo utilizado para almacenar datos tabulares de manera estructurada. Al igual que el formato CSV, el TSV se

utiliza para representar datos en forma de tabla, donde cada línea del archivo representa una fila y los valores de cada columna están separados por un carácter delimitador.

En el formato TSV los valores están separados por tabulaciones. Esta estructura facilita la lectura y el procesamiento de los datos utilizando programas y herramientas que admiten la lectura de archivos TSV, siendo ampliamente utilizado en diversas aplicaciones que involucran el intercambio y procesamiento de datos tabulares, como hojas de cálculo, bases de datos y sistemas de gestión de datos.

Capítulo 4

Diseño e implementación

En esta sección se detallará la metodología propuesta, la arquitectura del modelo, las estrategias de preprocesamiento de datos, así como los conjuntos de datos utilizados y las métricas de evaluación empleadas. A través de este enfoque, se espera mejorar la precisión y la eficiencia en la desambiguación de siglas médicas, contribuyendo así al avance y la mejora de la calidad de la investigación con datos clínicos.

4.1. Pruebas de concepto

Para el desarrollo de este TFG se han realizado diferentes pruebas hasta llegar a la solución más apropiada. Dado que el objetivo de este TFG es conseguir detectar los acrónimos presentes en un texto y desambiguarlos en función del contexto, se utilizaron diferentes métodos que permitían obtener resultados que se acercaban al objetivo; no obstante, no eran completamente eficientes ya que mostraban algunas limitaciones que se expondrán a continuación.

La primera prueba se basó en la combinación de un modelo de BERT explicado previamente en la sección de herramientas, aplicando la propiedad de MLM, un tipo de modelo de lenguaje utilizado para predecir palabras o tokens en una oración en función del contexto.

El proceso llevado a cabo consistió en entrenar este modelo con un gran corpus de datos clínicos que contienen siglas médicas, de manera que pudiese aprender las relaciones y patrones del lenguaje. En este entrenamiento, el método MLM enmascara aleatoriamente un número de tokens para que en la predicción pueda llevarse a cabo la desambiguación. Un ejemplo de este método aplicado a los datos se muestra en la siguiente frase:

- *[CLS] se solicita interconsulta [MASK] clinica de dolor quienes inician [MASK] [MASK] [MASK] analgesico con acetaminofen + [MASK] [MASK] [MASK] [MASK] + 30 mg via oral [MASK] [MASK] 6 h [MASK], dosis de morfina [MASK] caso de dolor severo (2 mg por [MASK] [MASK]) y pregabalina (150 [MASK] / 12 h) [SEP] [CLS] [MASK] inicia tratamiento antiviral con aciclovir 10mg / kg [MASK] 8 [MASK] i [SEP]*

- *destaca : hto : [MASK], [MASK] % [MASK] hb : 7, 2 g / [MASK] ; leucocitos : 235 [SEP] [CLS] - glucemia de 93 mg / dl con hemoglobina [MASK] [MASK] [MASK] [MASK] [MASK] normal ; creatinina : 1, 15 mg [MASK] dl [MASK] colesterol total : 114 mg / dl ; ferritina 133 mcg / l [MASK] sideremia 17 mcg / [MASK] ; bilirrubina 0, 29 mg [MASK] dl ; alanina - [MASK] [MASK] [MASK] [MASK] [MASK] 8 u / [MASK] [MASK] aspartato - aminotransferasa'*

Una vez entrenado el modelo y con las palabras enmascaradas, este permitirá realizar predicciones sobre aquellas palabras que se hayan enmascarado. No obstante, en la implementación de este método, se encontró una limitación principal que consistía en que, a la hora de inferir, el modelo priorizaba los acrónimos en lugar de utilizar la forma extendida de dicho acrónimo.

4.2. Arquitectura

Tras el estudio de las diferentes arquitecturas y modelos, y considerando los resultados altamente prometedores que proporcionan aquellos modelos que siguen una arquitectura tipo Tansformer en el campo de la literatura biomédica en inglés para permitir identificar y desambiguar los acrónimos presentes en textos clínicos, se ha decidido implementar esta tarea con esta misma arquitectura para textos en español. El éxito principal de la arquitectura Transformer reside en que evita la necesidad de un entrenamiento muy laborioso, ya que, parte de modelos entrenados con lengua española, y el proceso posterior está más enfocado al entrenamiento de una capa que realice una tarea determinada, en este caso, desambiguación de acrónimos, y así permitir que los modelos aprendan a reconocer acrónimos en un texto e identifiquen la definición que los corresponde.

4.2.1. Modelo

El funcionamiento general que se pretende alcanzar con este modelo es, dada una frase con un acrónimo, se sustituirá el acrónimo por sus diferentes formas extendidas, de manera que el modelo calculará cuál de sus formas extendidas es la correcta en ese contexto. Para un determinado acrónimo, se devolverá la probabilidad de que una definición sea la apropiada en el contexto de la frase, de manera que, en caso de que dicha probabilidad sea alta, se etiquetará este proceso como 1 (definición correcta en el contexto de la frase), y en caso contrario, como 0; los casos ambiguos serán representados con la etiqueta 0.5.

Para alcanzar este resultado, se ha implementado una arquitectura Transformer con un modelo BERT. Los modelos BERT ofrecidos por Google y expuesto por Devlin et al. en 2017 en el artículo [19], y explicados en el punto 3.1.1, han sido entrenados en inglés. Dado que el desarrollo de este TFG está propueto para el idioma español, se necesita un modelo preentrenado en dicho lenguaje para obtener unos resultados optimos. Tras el análisis de varios modelos y las ventajas y limitaciones que presentaban (4.4.2), este proyecto se centrará en los dos modelos preentrenados de BERT en español descritos por Cañete et al. 2020 en su artíuclo: BETO-Cased y BETO-Uncased [20].

Dado que resulta de interés mantener las mayúsculas y minúsculas presentes en el texto para poder alcanzar el objetivo de este trabajo, y conseguir identificar las siglas y acrónimos presentes en el texto, nos centraremos concretamente en la aplicación del modelo BETO-Cased. La arquitectura del modelo para la implementanción de este TFG se muestra en la figura 4.1.

4.3. Datos

Para el desarollo de este TFG se han utilizado datos de diferentes fuentes. A su vez, estos han sido preprocesados para adquirir la forma requerida para poder aplicarse al modelo.

Se puede distinguir entre 3 archivos diferentes: el diccionario de acrónimos que contiene la versión del acrónimo en su forma corta, es decir, el acrónimo, y su definición, el corpus de datos clínicos que contiene las frases utilizadas para entrenar el modelo con la estructura requerida, y un corpus de frases clínicas con acrónimos para realizar las predicciones.

```
BertForSequenceClassification(
  (bert): BertModel(
    (embeddings): BertEmbeddings(
      (word_embeddings): Embedding(31002, 768, padding_idx=1)
      (position_embeddings): Embedding(512, 768)
      (token_type_embeddings): Embedding(2, 768)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (encoder): BertEncoder(
      (layer): ModuleList(
        (0-11): 12 x BertLayer(
          (attention): BertAttention(
            (self): BertSelfAttention(
              (query): Linear(in_features=768, out_features=768, bias=True)
              (key): Linear(in_features=768, out_features=768, bias=True)
              (value): Linear(in_features=768, out_features=768, bias=True)
              (dropout): Dropout(p=0.1, inplace=False)
            )
            (output): BertSelfOutput(
              (dense): Linear(in_features=768, out_features=768, bias=True)
              (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
              (dropout): Dropout(p=0.1, inplace=False)
            )
          )
          (intermediate): BertIntermediate(
            (dense): Linear(in_features=768, out_features=3072, bias=True)
            (intermediate_act_fn): GELUActivation()
          )
          (output): BertOutput(
            (dense): Linear(in_features=3072, out_features=768, bias=True)
            (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
        )
      )
    )
    (pooler): BertPooler(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (activation): Tanh()
    )
  )
  (dropout): Dropout(p=0.1, inplace=False)
  (classifier): Linear(in_features=768, out_features=2, bias=True)
```

Figura 4.1: Arquitectura del modelo implementado

4.3.1. Diccionario de acrónimos

Para el desarrollo del archivo CSV que contiene las siglas, acrónimos y abreviaciones junto con su definición en la literatura médica, se utilizó el documento publicado por el Ministerio de Sanidad y Consumo escrito por Yetano, J., y Alberola, V. [21], junto con la información del diccionario de siglas médicas[1]. La obtención de los datos se ha llevado a cabo implementado técnicas de Web Scraping, que han permitido obtener grandes cantidades de datos de estos sitios web.

Este documento incluye las siglas médicas acompañadas de las definciones que puede tomar cada una. La creación del archivo consistió en transformalo a formato CSV, incluyendo una columna con la versión corta de la expresión o acrónimo, seguido de una segunda columna que incluye la definición para este, tal y como se muestra en la imagen adjunta 4.2.

[1] http://diccionario.sedom.es/

Abbreviation	Definition
DMNID	Diabetes mellitus no insulinodependiente
IVRS	Infección de las vías respiratorias superiores
LOE	Lesión con ocupación de espacio
NLP	Nefrolitotomía percutánea
PSAP	Presión sistólica de la arteria pulmonar
SCA	Síndrome cerebral agudo
SCA	Síndrome clínico aislado
SCA	Síndrome compartimental abdominal
SCA	Síndrome confusional agudo
SCA	Síndrome coronario agudo
SCA	Spinocerebellar ataxia (Ataxia espinocerebelosa)
VPPB	Vértigo paroxístico posicional benigno

Figura 4.2: Ejemplo del CSV del diccionario con las siglas médicas

4.3.2. Corpus de datos para entrenar el modelo

Para poder realizar un buen entrenamiento del modelo, se ha necesitado conseguir un archivo que contuviese multitud de frases clínicas con acrónimos. Para realizar esta tarea, se obtuvo un conjunto de textos clínicos planos cuya procendencia se encuentra en informes médicos anónimos en español proporcionados por la tarea BARR2 de IberEval2018 [22] en archivos txt, combinados con un archivo tsv que contiene las posiciones de los acrónimos, así como su definición, forma extendida y corta. Al combinar los textos planos con las anotaciones, se conseguía un dataset con toda la información relevante. El formato de texto que contenía este dataset era el siguiente:

Mujer de 42 años en el momento de someterse a trasplante hepático. Entre sus antecedentes personales previos al trasplante destacan apendicectomía en el año 1978, cirrosis hepática de probable origen alcohólico diagnosticada en 1989, HDA secundaria a varices esofágicas grado II en 1996 junto a hipertensión portal y ascitis. En Febrero de 1998 fue diagnosticada de hepatocarcinoma. La paciente recibió trasplante hepático en octubre de 1998 que cursó sin incidencias y fue dada de alta en tratamiento con tacrolimus. A los 18 meses del trasplante la paciente refirió por vez primera debilidad de miembros inferiores y pérdida de sensibilidad de los mismos evolucionando en el plazo de 2 meses a una paraplejia completa que afecta a vejiga urinaria.

La combinación del tsv con estos textos, daba como resultado un conjunto de frases relevantes, agrupadas en un archivo CSV que resume toda la información contenida en los textos planos para realizar una tarea de desambiguación de acrónimos.

El formato de este CSV consta de 6 columnas: **filename**, que indica el nombre del archivo txt del que proviene la frase, **abbreviation**, que indica la abreviación o acrónimo que contiene

dicha frase, **definition**, con la definición extendida para ese acrónimo, **start**, indica el token con la posición de incio del acrónimo, **end**, indica el token con la posición final del acrónimo y **sentence_long**, que muestra la frase que contiene el acrónimo.

Todo esto se muestra en la tabla 4.3 a continuación.

filename	abbreviation	definition	start	end	sentence_long
S0210-48062004000...	ml	mililitro	159	161	Ante la ausencia de r...
S0210-48062004000...	l	litro	156	157	Se realiza una gasom...
S0210-48062004000...	mEq	miliequivalente	152	155	Se realiza una gasom...
S0210-48062004000...	pCO2	presión parcial de co2	125	129	Se realiza una gasom...
S0210-48062004000...	HLA	human leucocyte anti...	104	107	Posteriormente el pac...
S0210-48062004000...	mmHg	milímetro de mercurio	136	140	Se realiza una gasom...
S0210-48062004000...	L	leucocito	194	195	Durante su permanen...
S0210-48062004000...	L	leucocito	167	168	Durante su permanen...
S0210-48062004000...	mg	miligramo	90	92	Ante la ausencia de r...
S0210-56912008000...	ECOT	ecocardiograma trans...	3	7	El ecocardiograma tra...
S0210-56912008000...	TACHT	tomografía computari...	3	8	La tomografía comput...
S0210-56912008000...	mm	milímetro	57	59	La TACHT al ingreso ...
S0210-56912008000...	angio-RMN	angioresonancia mag...	104	113	El ECOT a la semana...
S0210-56912008000...	TAS	tensión arterial sistólica	78	81	Se inició tratamiento ...
S0004-06142005001...	Kg	kilogramo	104	106	El paciente recibe trat...
S0004-06142005001...	mg	miligramo	101	103	El paciente recibe trat...
S0004-06142005001...	IgA	inmunoglobulina a	116	119	En programa de diális...
S0004-06142005001...	LCR	líquido cefalorraquídeo	375	378	La resonancia magné...
S0004-06142005001...	EEII	extremidades inferiores	111	115	A la exploración física...
S0004-06142005001...	gr	gramo	79	81	El paciente recibe trat...
S0004-06142005001...	Kg	kilogramo	52	54	El paciente recibe trat...
S0004-06142005001...	mg	miligramo	49	51	El paciente recibe trat...
S0004-06142005001...	ELISA	análisis de inmunoab...	281	286	La resonancia magné...
S0004-06142005001...	TSP	tropical spastic parap...	65	68	El paciente fue diagn...
S0004-06142005001...	PCR	reacción en cadena d...	330	333	La resonancia magné...
S0004-06142005001...	LCR	líquido cefalorraquídeo	126	129	La resonancia magné...

Figura 4.3: Ejemplo del CSV con el preprocesado de los textos

4.3.3. Datos para predicción

Para realizar la inferencia se utilizaron textos clínicos de la tarea BARR2 de IberEval2018 [22] al igual que los escogidos para la parte de entrenamiento, pero con una procedencia diferente, es decir, se generaron archivos independientes. Estos no tienen un formato predefinido. Son textos planos, sin anotar, en los que no se conoce información sobre la localización de los acrónimos. Pueden presentar cualquier tipo de estructura ya que, antes de entrar al modelo, serán procesados tal y como se contará en la sección 4.4.2.

4.4. Desarrollo

En esta sección se detalla el código escrito durante este TFG, así como los diferentes procesos implementados para pre-procesar el dataset, entrenar el modelo y realizar la inferencia. Todas las herramientas utilizadas han sido previamente descritas en el capítulo 3.

La parte computacional de este TFG puede dividirse en varias etapas, entre las que diferenciamos:

- Una primera etapa de preprocesado de los datos encaminada a la generación de los datos necesarios para el entrenamiento del modelo.
- Una segunda etapa de implementación, carga y aplicación del modelo y tokenizador a nuestros datos para realizar el entrenamiento de este.
- Una tercera etapa enfocada en realizar predicciones con el modelo previamente entrenado.
- Una última etapa de post procesado, cuyo objetivo era mostrar las predicciones realizadas por el modelo sobre las frases de entrada.

Todas estas etapas fueron implementadas siguiendo un enfoque modular y definiendo funciones específicas para cada tarea con el objetivo de facilitar la compresión y escalabilidad del proceso, de la forma mostrada en la figura 4.4.

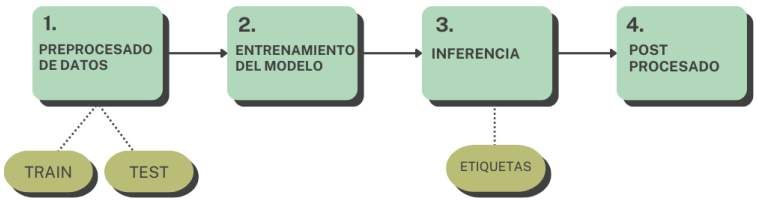

Figura 4.4: Diagrama de flujo del desarrollo computacional

En esta sección, se proporcionará una explicación detallada del código desarrollado, así como las técnicas utilizadas en cada etapa. Se definirán y explicarán las diferentes funciones, así como las conexiones existentes entre ellas para llevar a cabo el proceso de desambiguación completo.

4.4.1. Entrenamiento del modelo

Para la elaboración de este proyecto, se ha utilizado un modelo con arquitectura tipo Transformer, explicada en la sección 3.1, y más concretamente BETO-cased, por el requerimiento de distinción de las letras en mayúscula, modelo desarrollado e implementado por Cañete et al. 2020 en su artículo [20].

La versión del modelo preentreando utilizada es BERT para clasificación de secuencias, incluida en su repositorio de github [23]: 'dccuchile/bert-base-spanish-wwm-cased'. Además, también se incluye el tokenizador propio de este modelo, añadiendo como tokens especiales <start>y <end>.

En primer lugar, cargamos las secuencias de entrada que serán preprocesadas tal y como se expondrá a continuación. Además, son introducidas al tokenizador, de manera que se devolverá una secuencia númerica con un número asociado a cada palabra.

Para entrenar este tipo de modelos, es necesario que todas las secuencias tengan la misma longitud. Para ello, se establece una longitud máxima, truncando aquellas sequencias que sobrepasen dicha longitud y completando aquellas secuencias que no lleguen a dicha longitud con 0.

Para poder realizar el entrenamiento del modelo, se debe separar los datos de entrada en entrenamiento y validación, para lo cual aplicamos una función de la librería Scikit-Learn, que nos permite realizar esta separación de manera aleatoria. Una vez obtenida esta partición, introducimos el tensor de los datos de entrenamiento en el modelo y realizamos el entrenamiento.

En el procesado de la frase se pueden distinguir dos etapas: una primera etapa en la que, a partir de la información del CSV generado para entrenamiento, y conociendo la posición de los acrónimos, se introducen en la frase los tokens <start>y <end>, antes y despues del acrónimo presente en la frase, respectivamente. Y una segunda etapa en la cual, a la frase generada en el paso anterior, se le añade al inicio, el token [CLS], seguido de la definición del acrónimo localizado en dicha frase, y del token [SEP].

Visualmente, esto se muestra en el siguiente ejemplo:

- **Frase:** *En programa de diálisis peritoneal desde enero del mismo año por IRC progresiva secundaria a glomerulonefritis tipo inmunoglobulina a con esclerosis focal y segmentaria asociada diagnosticada en 1984*

- **Tokens de encapsulación de acrónimo:** *En programa de diálisis peritoneal desde enero del mismo año por <start>IRC <end>progresiva secundaria a glomerulonefritis tipo inmunoglobulina a con esclerosis focal y segmentaria asociada diagnosticada en 1984*

- **Tokens de inicio y separación:** *[CLS] Insuficiencia Renal Crónica [SEP] En programa de diálisis peritoneal desde enero del mismo año por <start>IRC <end>progresiva secundaria a glomerulonefritis tipo inmunoglobulina a con esclerosis focal y segmentaria asociada diagnosticada en 1984*

El CSV resultante tiene la siguiente estructura 4.5:

texto	start	end	acr	tokens	def
El ecocardiograma transesofágico a la semana mostró mejoría respecto al del ingreso, a los 15 días la angioresonancia magnética (angio-RMN) informó de una normal distribución de la salida de los troncos supraaórticos sin imágenes sugerentes de rotura traumática de la aorta	135	138	RMN	El ecocardiograma transesofágico a la semana mostró mejoría respecto al del ingreso, a los 15 días la angioresonancia magnética (angio-<start> RMN <end>) informó de una normal distribución de la salida de los troncos supraaórticos sin imágenes sugerentes de rotura traumática de la aorta	Resonancia magnética nuclear
En programa de diálisis peritoneal desde enero del mismo año por IRC progresiva secundaria a glomerulonefritis tipo inmunoglobulina a con esclerosis focal y segmentaria asociada diagnosticada en 1984	65	68	IRC	En programa de diálisis peritoneal desde enero del mismo año por <start> IRC <end> progresiva secundaria a glomerulonefritis tipo inmunoglobulina a con esclerosis focal y segmentaria asociada diagnosticada en 1984	Insuficiencia renal crónica
En programa de diálisis peritoneal desde enero del mismo año por IRC progresiva secundaria a glomerulonefritis tipo inmunoglobulina a con esclerosis focal y segmentaria asociada diagnosticada en 1984	65	68	IRC	En programa de diálisis peritoneal desde enero del mismo año por <start> IRC <end> progresiva secundaria a glomerulonefritis tipo inmunoglobulina a con esclerosis focal y segmentaria asociada diagnosticada en 1984	Insuficiencia respiratoria crónica
La resonancia magnética nuclear (RMN) no mostró lesiones a nivel cerebral o espinal, el análisis del líquido cefalorraquídeo (LCR) mostró pleocitosis linfocitaria sin bandas oligoclonales, los potenciales evocados resultaron alterados a nivel tibial y fueron los test serológicos, ELISA, y la reacción en cadena de la polimerasa (PCR) los que fueron positivos para HTLV-I en líquido cefalorraquídeo y sangre en el primer caso y en sangre la PCR	33	36	RMN	La resonancia magnética nuclear (<start> RMN <end>) no mostró lesiones a nivel cerebral o espinal, el análisis del líquido cefalorraquídeo (LCR) mostró pleocitosis linfocitaria sin bandas oligoclonales, los potenciales evocados resultaron alterados a nivel tibial y fueron los test serológicos, ELISA, y la reacción en cadena de la polimerasa (PCR) los que fueron positivos para HTLV-I en líquido cefalorraquídeo y sangre en el primer caso y en sangre la PCR	Resonancia magnética nuclear

Figura 4.5: Ejemplo del CSV que almacena las frases preprocesadas

Una vez obtenidas las frases preprocesadas, son almacenadas en un nuevo archivo CSV que contiene los datos mencionados previamente (texto, start, end, acrónimo, y definición), junto con una nueva columna **tokens**, que almacena la frase con el token encapsulado. De este modo, y con las frases ya preprocesdas, se podía pasar al entrenamiento del modelo.

Las características de este entrenamiento fueron: optimizador AdamW, con 8 epochs, tamaño del batch de 32, longitud máxima de 128 y una tasa de aprendizaje de 1×10^{-5}.

Tras el entrenamiento, realizamos una evaluación de este con los datos de validación, calculando diferentes métricas. Entre las métricas evaluadas se encuentran: el recall, la precisión, el f1-score y el area bajo la curva. Las prestaciones obtenidas por este modelo para nuetro conjunto de datos se resumen en la tabla 4.6 a continuación:

Como se puede apreciar en la tabla, las medidas de evaluación del modelo son razonables. Fue por este motivo, por ensayos previos con otros modelos, como por ejemplo, con el modelo preentrenado de BERT-Multilingüe, descrito en la sección 3.1.1., que ha sido creado incluyendo el español (entre otros 104 idiomas), cuya implementación para este proyecto no resultó conveniente debido a que supone un gasto de gran cantidad de recursos al contar con una dimensión

METRICAS	VALORES
PRECISIÓN	0.9856
RECALL	0.97857
F1 SCORE	0.98207
AREA BAJO LA CURVA	0.9707

Figura 4.6: Prestaciones obtenidas por el modelo

de vocabulario de 110M y sólo utilizar una parte muy pequeña de esta, la correspondiente al español. Además, la precisión sobre la lengua también es menor, y no aportará grandes beneficios. Por otro lado, la aplicación del modelo RoBERTa [24], mejora del modelo BERT que realiza un entrenamiento con un corpus en español de 570GB, tampoco se ha considerado eficiente ya que supone un gasto de recursos muy elevado, y los resultados obtenidos no mejoraron los de el modelo BERT.

Por estos motivos, además de por la confianza depositada en el rendimiento del modelo BERT por sus características, es el motivo por el que se decidió utilizar este para la inferencia.

4.4.2. Inferencia

Para poder desambigüar acrónimos se ha partido de frases sin anotar que los contienen. Además, para inferir la definición correcta a través del modelo diseñado, ha sido necesario transformar estas frases a un formato idéntico al utilizado durante el entrenamiento del modelo.

Para realizar este preprocesado y conseguir frases de la forma: *[CLS] Insuficiencia Renal Crónica [SEP] En programa de diálisis peritoneal desde enero del mismo año por <start>IRC <end>progresiva secundaria a glomerulonefritis tipo inmunoglobulina a con esclerosis focal y segmentaria asociada diagnosticada en 1984*, se ha utilizado una función que recibe como argumento de entrada un texto y localiza el acrónimo presente en este, ya que, a diferencia del entrenamiento, se desconoce su posición. Dicha función devuelve un dataframe que contiene, para el texto introducido, tantas filas como acrónimos y definiciones existentes haya para esos acrónimos, y columnas con información sobre: el texto introducido, la posición de inicio y fin del acrónimo, el acrónimo y su definición, la frase con el acrónimo localizado, y la frase completamente procesada.

Para una explicación más representativa de como se preprocesan estos textos, se va a mostrar

un ejemplo a continuación.

La funcion recibe como argumento de entrada una variable denominada *texto* que contiene un texto aleatorio, como por ejemplo:

- *texto = En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales*

A continuación, haremos una llamada a la función introduciendo la variable como argumento de entrada, de manera que, tras localizar el acrónimo, la salida de esta función será un dataframe como el mostrado en la Figura 4.7:

	texto	start	end	acr	tokens	def	sentences
0	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	203	206	GOT	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, <start> GOT <end>, GPT, LDH, CPK, sodio y potasio normales	Glutámic oxaloacetic transaminase (Transaminasa glutámico oxalacética)	[CLS] Glutámic oxaloacetic transaminase (Transaminasa glutámico oxalacética) [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, <start> GOT <end>, GPT, LDH, CPK, sodio y potasio normales [SEP]
1	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	208	211	GPT	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, <start> GPT <end>, LDH, CPK, sodio y potasio normales	Glutamic pyruvic transaminase (Transaminasa glutámico pirúvica)	[CLS] Glutamic pyruvic transaminase (Transaminasa glutámico pirúvica) [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, <start> GPT <end>, LDH, CPK, sodio y potasio normales [SEP]
2	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	213	216	LDH	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, <start> LDH <end>, CPK, sodio y potasio normales	Lactatodeshidrogenasa	[CLS] Lactatodeshidrogenasa [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, <start> LDH <end>, CPK, sodio y potasio normales [SEP]
3	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	213	216	LDH	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, <start> LDH <end>, CPK, sodio y potasio normales	Linfoma difuso histiocítico	[CLS] Linfoma difuso histiocítico [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, <start> LDH <end>, CPK, sodio y potasio normales [SEP]
4	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	218	221	CPK	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, <start> CPK <end>, sodio y potasio normales	Creatine phosphokinase (Creatinofosfocinasa)	[CLS] Creatine phosphokinase (Creatinofosfocinasa) [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, <start> CPK <end>, sodio y potasio normales [SEP]
5	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	61	62	U	En el examen bioquímico la gamma-glutamil transpeptidasa 139 <start> U <end>/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Unidad	[CLS] Unidad [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 <start> U <end>/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
6	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	61	62	U	En el examen bioquímico la gamma-glutamil transpeptidasa 139 <start> U <end>/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Uracilo	[CLS] Uracilo [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 <start> U <end>/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
7	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	61	62	U	En el examen bioquímico la gamma-glutamil transpeptidasa 139 <start> U <end>/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Uranio	[CLS] Uranio [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 <start> U <end>/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
8	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	61	62	U	En el examen bioquímico la gamma-glutamil transpeptidasa 139 <start> U <end>/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Urología	[CLS] Urología [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 <start> U <end>/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
9	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	61	62	U	En el examen bioquímico la gamma-glutamil transpeptidasa 139 <start> U <end>/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Útero	[CLS] Útero [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 <start> U <end>/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
10	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	63	64	L	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/<start> L <end>, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Lactancia	[CLS] Lactancia [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/<start> L <end>, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
11	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	63	64	L	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/<start> L <end>, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Leucemia	[CLS] Leucemia [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/<start> L <end>, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
12	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	63	64	L	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/<start> L <end>, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Leucocito	[CLS] Leucocito [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/<start> L <end>, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]

13	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	63	64	L	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/<start> L <end>, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Linfocito	[CLS] Linfocito [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/<start> L <end>, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
14	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	63	64	L	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/<start> L <end>, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Lóbulo	[CLS] Lóbulo [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/<start> L <end>, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
15	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	63	64	L	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/<start> L <end>, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Lumbar	[CLS] Lumbar [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/<start> L <end>, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
16	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	63	64	L	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/<start> L <end>, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Símbolo de litro	[CLS] Símbolo de litro [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/<start> L <end>, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
17	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	81	82	U	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 <start> U <end>/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Unidad	[CLS] Unidad [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 <start> U <end>/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
18	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	81	82	U	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 <start> U <end>/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Uracilo	[CLS] Uracilo [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 <start> U <end>/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
19	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	81	82	U	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 <start> U <end>/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Uranio	[CLS] Uranio [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 <start> U <end>/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
20	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	81	82	U	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 <start> U <end>/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Urología	[CLS] Urología [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 <start> U <end>/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
21	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	81	82	U	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 <start> U <end>/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Útero	[CLS] Útero [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 <start> U <end>/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
22	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	83	84	L	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/<start> L <end>, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Lactancia	[CLS] Lactancia [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/<start> L <end>, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
23	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	83	84	L	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/<start> L <end>, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Leucemia	[CLS] Leucemia [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/<start> L <end>, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
24	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	83	84	L	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/<start> L <end>, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Leucocito	[CLS] Leucocito [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/<start> L <end>, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
25	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	83	84	L	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/<start> L <end>, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Linfocito	[CLS] Linfocito [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/<start> L <end>, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
26	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	83	84	L	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/<start> L <end>, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Lóbulo	[CLS] Lóbulo [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/<start> L <end>, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
27	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	83	84	L	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/<start> L <end>, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Lumbar	[CLS] Lumbar [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/<start> L <end>, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]
28	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/L, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	83	84	L	En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/<start> L <end>, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales	Símbolo de litro	[CLS] Símbolo de litro [SEP] En el examen bioquímico la gamma-glutamil transpeptidasa 139 U/L, f alcalina 217 U/<start> L <end>, hierro 19 ug/dL, la ferritina 436 ng/ml, glucosa, creatinina, calcio, fósforo, colesterol, triglicéridos, proteínas, GOT, GPT, LDH, CPK, sodio y potasio normales [SEP]

Figura 4.7: Salida de la función de preprocesamiento del texto

A partir de este dataframe, y con las frases preprocesadas, están preparadas para introducirlas al modelo y realizar la inferencia.

Para obtener las predicciones del modelo, se añade una capa softmax a la salida de este.

Esta capa se utiliza en modelos de clasificiación ya que, toma un vector de entrada, y genera un vector de salida que representa la distribución de probabilidad de las diferentes clases.

De esta manera, recibimos a la salida del modelo para una determinada frase un vector con dos valores siguiendo la forma, [valor1, valor2], siendo el primer elemento el de la posición 0, y el segundo el de la posición 1, y obteniendo tantos vectores como predicciones de acrónimos diferentes estemos realizando.
Para el etiquetado de estos valores, se crea una función de etiquetado de manera que:

- Si la probabilidad del elemento en la posición 1 es mayor a 0.75, se etiqueta como 1, y la definición para el acrónimo se predice acorde con el contexto de la frase.

- Si la probablidad del elemento en la posición 0, es mayor a 0.75, se etiqueta como 0, y la defincón se clasifica como no óptimas en el contexto de la frase en el que se encuentran.

- En cualquier otro caso, el modelo no está lo suficientemente seguro de que la forma extendida seleccionada sea la correcta o no, etiquetandose entonces como 0.5.

Con el vector de etiquetas ya calculado para cada frase, se incorporó una nueva columna al dataframe mencionado en la parte de preprocesado de los datos, que incluía la etiqueta de predicción para cada frase, tal y como se muestra en la figura 4.8

38	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (RCP) en extensión	199	202	RCP	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión	Reanimación / Resucitación cardiopulmonar	[CLS] Reanimación / Resucitación cardiopulmonar [SEP] A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión [SEP]	0.5
39	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (RCP) en extensión	199	202	RCP	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión	Reflejo cutáneo-plantar	[CLS] Reflejo cutáneo-plantar [SEP] A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión [SEP]	1.0
40	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (RCP) en extensión	199	202	RCP	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión	Registro de casos psiquiátricos	[CLS] Registro de casos psiquiátricos [SEP] A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión [SEP]	0.0
41	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (RCP) en extensión	199	202	RCP	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión	Registro de cáncer poblacional	[CLS] Registro de cáncer poblacional [SEP] A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión [SEP]	0.0
42	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (RCP) en extensión	199	202	RCP	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión	Responsabilidad civil profesional	[CLS] Responsabilidad civil profesional [SEP] A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión [SEP]	0.0
43	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (RCP) en extensión	199	202	RCP	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión	Respuesta completa patológica	[CLS] Respuesta completa patológica [SEP] A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión [SEP]	0.5

Figura 4.8: Etiquetado

De este modo, se puede apreciar que, para este caso particular, y para el acrónimo RCP, las definiciones que no coinciden con la correcta se etiquetan con un valor de 0, y ocasionalmente 0.5; mientras que la definición correcta, recibe la etiqueta 1.

4.4.3. Post Procesado

El post procesado de este proyecto consistió en la sustitución de la definición correspondiente al acrónimo identificado como correcto en el contexto de la frase, y etiquetado por tanto con valor de 1. De esta manera, recibiremos como salida final del modelo, a partir del texto de entrada, una dataframe con tantas filas como acrónimos se hayan identificado en el texto y tantas definiciones tenga cada acrónimo, es decir, un texto con 6 acrónimos, y suponiendo 3 definiciones por acrónimo, generaría un dataframe con una salida de 18 filas.

En esta fase de post procesado, se reemplaza en la frase inicial que incluía el acrónimo, la posición que ocupaba este por su definición, y se añade como nueva columna al dataframe ya existente, de manera que, en los casos en los que la etiqueta no sea 1, el valor de la columna sera NaN, y en caso contrario, se encontrará la frase con el acrónimo remplazado por su definición. Utilizando el mismo ejemplo que en el caso anterior, la salida final para un acrónimo particular puede verse en la figura 4.9.

38	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (RCP) en extensión	199	202	RCP	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión	Reanimación / Resucitación cardiopulmonar	[CLS] Reanimación / Resucitación cardiopulmonar [SEP] A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión [SEP]	0.5	NaN
39	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (RCP) en extensión	199	202	RCP	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión	Reflejo cutáneo-plantar	[CLS] Reflejo cutáneo-plantar [SEP] A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión [SEP]	1.0	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (Reflejo cutáneo-plantar) en extensión
40	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (RCP) en extensión	199	202	RCP	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión	Registro de casos psiquiátricos	[CLS] Registro de casos psiquiátricos [SEP] A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión [SEP]	0.0	NaN
41	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (RCP) en extensión	199	202	RCP	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión	Registro de cáncer poblacional	[CLS] Registro de cáncer poblacional [SEP] A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión [SEP]	0.0	NaN
42	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (RCP) en extensión	199	202	RCP	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión	Responsabilidad civil profesional	[CLS] Responsabilidad civil profesional [SEP] A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión [SEP]	0.0	NaN
43	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (RCP) en extensión	199	202	RCP	A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión	Respuesta completa patológica	[CLS] Respuesta completa patológica [SEP] A la exploración física no se observa déficit motor pero sí disminución de la sensibilidad vibratoria en ambas extremidades inferiores, reflejos osteotendinosos (ROT) vivos y reflejo cutaneoplantar (<start> RCP <end>) en extensión [SEP]	0.5	NaN

Figura 4.9: Post procesado

4.5. Resultados y errores

Para relizar un análisis completo de los resultados proporcionados por el proceso completo de desambiguación, se realizaron de manera independiente 3 pruebas de predicciones, obteniendo para cada uno las etiquetas asociadas a la predicción, 4.10.

- En la primera tarea, se utilizaron 30 frases, entre las cuales se obtuvieron 17 acrónimos diferentes. Para este número de frases y acrónimos, se distinguieron un total de 2692 combinaciones posibles, ya que para un mismo acrónimo, puede haber gran variedad de definiciones. En esta primera tarea, se etiquetaron 1653 defincciones correctas en ese contexto, y por tanto etiquetadas con 1, 817 con 0.5 y 222 con 0, por tanto un total de 1039 de predicciones no óptimas en el contexto de la frase en el que se encuentran.

- En una segunda tarea, se utilizaron 51 frases, entre las cuales se obtuvieron 32 acrónimos diferentes. En este caso, las combinaciones se elevaron a 6687, las etiquetas con 1 alcanzaron un valor de 4589, mientras que las etiquetas con 0.5 y 0, 1397 y 701, respectivamente.

- En la ultima tarea realizada, se seleccionaron 100 frases diferentes, entre las cuales se obtuvieron un total de 54 acrónimos diferentes. Para este último caso, las combinaciones resultantes fueron 13321, el número de etiquetados con 1 resultó 10826, mientras que el número de etiquetados con 0.5 y 0, resultó 1401 y 1094, respectivamente.

	Frases	Acrónimos	Textos a inferir	Etiquetado
Tarea 1	30	17	2692	1 - 1653 0.5 - 817 0 - 222
Tarea 2	50	32	6687	1 - 4589 0.5 - 1397 0 - 701
Tarea 3	100	54	13321	1 - 10826 0.5 - 1401 0 - 1094

Figura 4.10: Tabla de resultados de la desambiguación final

Aún teniendo en cuenta que un mismo acrónimo puede aparecer en diferentes frases, y por lo tanto, se cuenta como acrónimo repetido a pesar de que esté en una predicción diferente, el

número de predicciones etiquetadas con 1, y por tanto clasificadas como acorde en ese contexto, resulta bastante elevado en comparación con las predicciones clasificadas como no óptimas en el contexto de la frase en el que se encuentran (etiquetas 0.5 y 0). Esto indica que, en algunos casos, se estarían etiquetando algunas predicciones como óptimas en el contexto sin serlo, tal y como se muestra en la figura 4.11.

En este ejemplo, podemos apreciar como para el caso concreto de la predicción de la abreviación U, se etiquetan como 1 todas las defincionesexistentes para esa determinada abreviación (Unidad, Uracilo, Uranio, Urología...), en lugar de únicamente la correcta.

	texto	start	end	acr	tokens	def	sentences	Prediction	
0	En el análisis bioquímico destacaban una GGT 220 U/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	41	44	GGT	En el análisis bioquímico destacaban una <start> GGT <end> 220 U/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	Gammaglutamiltranspeptidasa	[CLS] Gammaglutamiltranspeptidasa [SEP] En el análisis bioquímico destacaban una <start> GGT <end> 220 U/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml [SEP]	1.0	En el análisis bioquímico destacaban una Gammaglutamiltranspeptidasa 220 U/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml
1	En el análisis bioquímico destacaban una GGT 220 U/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	54	57	GPT	En el análisis bioquímico destacaban una GGT 220 U/L, <start> GPT <end> 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	Glutamic pyruvic transaminase (Transaminasa glutámico pirúvica)	[CLS] Glutamic pyruvic transaminase (Transaminasa glutámico pirúvica) [SEP] En el análisis bioquímico destacaban una GGT 220 U/L, <start> GPT <end> 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml [SEP]	1.0	En el análisis bioquímico destacaban una GGT 220 U/L, Glutamic pyruvic transaminase (Transaminasa glutámico pirúvica) 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml
2	En el análisis bioquímico destacaban una GGT 220 U/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	66	69	GOT	En el análisis bioquímico destacaban una GGT 220 U/L, GPT 45 U/L, <start> GOT <end> 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	Glutámic oxaloacetic transaminase (Transaminasa glutámico oxalacética)	[CLS] Glutámic oxaloacetic transaminase (Transaminasa glutámico oxalacética) [SEP] En el análisis bioquímico destacaban una GGT 220 U/L, GPT 45 U/L, <start> GOT <end> 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml [SEP]	1.0	En el análisis bioquímico destacaban una GGT 220 U/L, GPT 45 U/L, Glutámic oxaloacetic transaminase (Transaminasa glutámico oxalacética) 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml
3	En el análisis bioquímico destacaban una GGT 220 U/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	86	89	ALP	En el análisis bioquímico destacaban una GGT 220 U/L, GPT 45 U/L, GOT 44 unidad /L, y <start> ALP <end> 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	Fosfatasa alcalina	[CLS] Fosfatasa alcalina [SEP] En el análisis bioquímico destacaban una GGT 220 U/L, GPT 45 U/L, GOT 44 unidad /L, y <start> ALP <end> 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml [SEP]	1.0	En el análisis bioquímico destacaban una GGT 220 U/L, GPT 45 U/L, GOT 44 unidad /L, y Fosfatasa alcalina 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml
4	En el análisis bioquímico destacaban una GGT 220 U/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	49	50	U	En el análisis bioquímico destacaban una GGT 220 <start> U <end>/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	Unidad	[CLS] Unidad [SEP] En el análisis bioquímico destacaban una GGT 220 <start> U <end>/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml [SEP]	1.0	En el análisis bioquímico destacaban una GGT 220 Unidad/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml
5	En el análisis bioquímico destacaban una GGT 220 U/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	49	50	U	En el análisis bioquímico destacaban una GGT 220 <start> U <end>/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	Uracilo	[CLS] Uracilo [SEP] En el análisis bioquímico destacaban una GGT 220 <start> U <end>/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml [SEP]	1.0	En el análisis bioquímico destacaban una GGT 220 Uracilo/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml
6	En el análisis bioquímico destacaban una GGT 220 U/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	49	50	U	En el análisis bioquímico destacaban una GGT 220 <start> U <end>/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	Uranio	[CLS] Uranio [SEP] En el análisis bioquímico destacaban una GGT 220 <start> U <end>/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml [SEP]	1.0	En el análisis bioquímico destacaban una GGT 220 Uranio/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml
7	En el análisis bioquímico destacaban una GGT 220 U/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	49	50	U	En el análisis bioquímico destacaban una GGT 220 <start> U <end>/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml	Urología	[CLS] Urología [SEP] En el análisis bioquímico destacaban una GGT 220 <start> U <end>/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml [SEP]	1.0	En el análisis bioquímico destacaban una GGT 220 Urología/L, GPT 45 U/L, GOT 44 unidad /L, y ALP 153 U/L, hierro 20 ug/dL, y ferritina 441,6 ng/ml

Figura 4.11: Ejemplo de resultados de la desambiguación final

Tras estudiar las limitaciones y errores que presenta en la predicción nuestro modelo, se llegó a la conclusión de que:

- En primer lugar, fallaban aquellas predicciones de acrónimos que aparecían por primera vez en test y no se habían considerado en el entrenamiento.
- El hecho de que se planteen múltiples definiciones para un mismo acrónimo, también supone un problema para el modelo en aquellos casos en los que se superan las 5 definiciones diferentes, principalmente por falta de contexto de la definición, etiquetando en estos casos, más de una como correcta.

Capítulo 5

Conclusiones

En esta sección, se presentan las conclusiones obtenidas a partir del desarrollo de este TFG sobre desambiguación de siglas médicas, así como las oportunidades de investigación futura y la aplicación práctica de los conocimientos adquiridos a lo largo del grado.

5.1. Conclusiones finales

En este TFG se ha abordado el desafío de conseguir una desambiguación de siglas y acrónimos médicos en la lengua española con el objetivo de mejorar la compresión e interpretación de textos clínicos. Se ha hecho uso de técnicas y algoritmos basados en el aprendizaje automático, PLN, y la IA, para conseguir un modelo de desambiguación.

Los resultados obtenidos durante la evaluación del modelo fueron alentadores, alcanzando una precisión del 98 %, demostrando así su eficacia para llevar a cabo la desambiguación de una amplia gama de siglas y acrónimos médicos. Sin embargo, es importante destacar las limitaciones que se han encontrado en este trabajo, principalmente relacionadas con los datos y el entrenamiento del modelo. El conjunto de datos que se utilizó para el entrenamiento del modelo, fue bastante exhaustivo en términos de cantidad de ejemplos y diversidad de siglas y acrónimos médicos; no obstante, presentó algunas limitaciones que han afectado al rendimiento del modelo, y por tanto a las predicciones que realiza.

En primer lugar, el conjunto de datos podía ser representativo de las siglas y acrónimos médicos, pero no suficiente, ya que existe una muy amplia gama de estos. Además, aunque se intentaron recopilar todos los ejemplos posibles en los diferentes contextos en los que un

acrónimo o sigla puede existir, es complicado determinar si se han omitido algunas variaciones y contextos específicos. Dada esta falta de representatitividad, se ha visto limitada la capacidad del modelo para generalizar adecuadamente con nuevos ejemplos y contextos.

Por otro lado, la calidad de los datos también ha sido una limitación. Se han encontrado algunos casos de acrónimos y siglas que presentan definiciones reconocidas como diferentes, pero referidas a lo mismo, como por ejemplo, para el acrónimo *L*, se reconocen las definiciones *Litro* y *Unidad de litro*, como diferentes.

5.2. Trabajos futuros

Dado que la principal limitación de este trabajo ha sido la imposibilidad de entrenar un modelo en todos los contextos en los que puede existir un acrónimo determinado, sería positivo realizar experimientos aumentando los datos para el entrenamiento del modelo y así incluir más contexto para los acrónimos ya existentes, así como nuevos acrónimos con sus respectivos contextos. De esta manera, el modelo podría comprender más contextos y acrónimos y mejoraría la eficacia de su predicción.

Además, sería conveniente unificar las definiciones asociadas a un determinado acrónimo, de manera que pudiesemos evitar aquellos errores de predicción generados por la repetición de una misma definición, pero con diferentes términos.

Por último, y dado que este proyecto ha estado focalizado en explorar la arquitectura Transformer en su versión preentrenada BERT, para lengua española, BETO, se propone la posibilidad de utilizar otros modelos ya preentrenados en lengua inglesa, como ELECTRA o ALBERT, entre otros, y probar su aplicación y rendimiento en corpus con datos clínicos en lengua española.

5.3. Aplicación de lo aprendido

En el desarrollo de este TFG he podido aplicar los conocimientos adquiridos a lo largo de mi grado en Ingeniería Biomédica, así como las asignaturas que he cursado. A continuación, destacaré cómo cada una de estas asignaturas ha contribuido a mi capacidad para abordar este desafío.

Fundamentos de la programación: la asignatura de fundamentos de la programación inició mi camino en el ámbito de la informática y programación. Sentó las bases para mi comprensión de los conceptos fundamentales, aprendí a escribir algoritmos, utilizar estructuras de control, manipular variables y entender la lógica detrás de un programa.

Bases de datos: esta asignatura ha sido la más decisiva en el grado. El conocimiento adquirido en el curso de bases de datos fue de gran relevancia en mi trabajo de desambiguación de siglas médicas. Aprendí a diseñar y gestionar bases de datos, así como a escribir consultas SQL para extraer información específica. Esto me permitió desarrollar un repositorio de datos eficiente y estructurado para almacenar las siglas y sus posibles significados.

Inteligencia Artificial: la asignatura de Inteligencia Artificial me proporcionó una comprensión profunda de los algoritmos y técnicas utilizados en el campo de la inteligencia artificial. Aprendí sobre el aprendizaje automático y las redes neuronales, lo cuál me hizo indagar por el tema, y llegar a la arquitectura Transformer, que tanto interés me generó. Aprendí herramientas valiosas para poder realizar una tarea de desambiguación de siglas médicas. Utilicé algoritmos de clasificación y procesamiento del lenguaje natural para mejorar la precisión de la desambiguación.

Por último, las prácticas externas como ingeniero de datos en ASHO: mis prácticas externas como ingeniero de datos en ASHO fueron una experiencia enriquecedora, ya que pude aplicar directamente mis conocimientos adquiridos en el ámbito real. Durante este periodo, conocí nuevos algoritmos y estrucuturas de inteligencia artificial, trabajé en el procesamiento y análisis de grandes volúmenes de datos médicos, lo cual me permitió entender los desafíos prácticos y aplicar técnicas de manejo de datos en el contexto médico. Además, fueron ellos los que me impulsaron a iniciarme en este camino.

En conclusión, las asignaturas de Fundamentos de Programación, Bases de Datos, Inteligencia Artificial y mis prácticas externas como ingeniero de datos en ASHO han sido fundamentales para poder desarrollar este Trabajo de Fin de Grado sobre la desambiguación de siglas médicas. Estas asignaturas me han proporcionado las herramientas y habilidades necesarias para desarrollar algoritmos, trabajar con bases de datos, aplicar técnicas de inteligencia artificial y enfrentar los desafíos prácticos de la industria biomédica.

Bibliografía

[1] Niklas Kühl, Max Schemmer, Marc Goutier, and Gerhard Satzger. Artificial intelligence and machine learning. *Electronic Markets*, pages 1–10, 2022.

[2] Prakash M Nadkarni, Lucila Ohno-Machado, and Wendy W Chapman. Natural language processing: an introduction. *Journal of the American Medical Informatics Association*, 18(5):544–551, 2011.

[3] Abhimanyu S Ahuja and Lawrence S Halperin. Understanding the advent of artificial intelligence in ophthalmology. *Journal of Current Ophthalmology*, 31(2):115, 2019.

[4] Mohammad Hosein Rezazade Mehrizi, Peter van Ooijen, and Milou Homan. Applications of artificial intelligence (ai) in diagnostic radiology: a technography study. *European radiology*, 31:1805–1811, 2021.

[5] Elham Nazari, Amir Hossein Farzin, Mehran Aghemiri, Amir Avan, Mahmood Tara, and Hamed Tabesh. Deep learning for acute myeloid leukemia diagnosis. *Journal of Medicine and Life*, 13(3):382, 2020.

[6] Michael Krauthammer and Goran Nenadic. Term identification in the biomedical literature. *Journal of biomedical informatics*, 37(6):512–526, 2004.

[7] Augusto Cortez Vásquez, Jaime Pariona Quispe, Ana Maria Huayna, et al. Procesamiento de lenguaje natural. *Revista de investigación de Sistemas e Informática*, 6(2):45–54, 2009.

[8] Mahesh Joshi, Serguei Pakhomov, Ted Pedersen, and Christopher G Chute. A comparative study of supervised learning as applied to acronym expansion in clinical reports. In *AMIA annual symposium proceedings*, volume 2006, page 399. American Medical Informatics Association, 2006.

[9] Francesco Ronzano and Laura Inés Furlong. Ibi-upf at barr-2017: Learning to identify abbreviations in biomedical literature system description. In *IberEval@ SEPLN*, pages 255–263, 2017.

[10] David Nadeau and Peter D Turney. A supervised learning approach to acronym identification. In *Advances in Artificial Intelligence: 18th Conference of the Canadian Society for Computational Studies of Intelligence, Canadian AI 2005, Victoria, Canada, May 9-11, 2005. Proceedings 18*, pages 319–329. Springer, 2005.

[11] Montse Cuadros, Naiara Pérez, Iker Montoya, and Aitor García Pablos. Vicomtech at barr2: Detecting biomedical abbreviations with ml methods and dictionary-based heuristics. In *IberEval@ SEPLN*, pages 322–328, 2018.

[12] María Elena García García. Desambiguación de acrónimos en literatura médica española. 2021.

[13] Alexandra Pomares-Quimbaya, Pilar López-Úbeda, Michel Oleynik, and Stefan Schulz. Leveraging pubmed to create a specialty-based sense inventory for spanish acronym resolution. In *Digital Personalized Health and Medicine*, pages 292–296. IOS Press, 2020.

[14] Ren Kai, Li Na, Xiong Wei, and Wang Shi-Wen. Disambiguation of biomedical acronyms based on a bidirectional recurrent neural network of character-level features. *Journal of Engineering Science & Technology Review*, 12(6), 2019.

[15] Serguei Pakhomov, Ted Pedersen, and Christopher G Chute. Abbreviation and acronym disambiguation in clinical discourse. In *AMIA annual symposium proceedings*, volume 2005, page 589. American Medical Informatics Association, 2005.

[16] Yonghui Wu, Jun Xu, Yaoyun Zhang, and Hua Xu. Clinical abbreviation disambiguation using neural word embeddings. In *Proceedings of BioNLP 15*, pages 171–176, 2015.

[17] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

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

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

[20] José Cañete, Sebastián Donoso, Felipe Bravo-Marquez, Andrés Carvallo, and Vladimir Araujo. Albeto and distilbeto: Lightweight spanish language models. *arXiv preprint arXiv:2204.09145*, 2022.

[21] Javier Yetano Laguna and Vicent Alberola Cuñat. *Diccionario de siglas médicas*. SEDOM, 2011.

[22] BARR: A biomedical abbreviation and acronym resolver. `https://temu.bsc.es/BARR2/`. Accedido el 22 de junio de 2023.

[23] José Cañete, Gabriel Chaperon, Rodrigo Fuentes, Jou-Hui Ho, Hojin Kang, and Jorge Pérez. Spanish pre-trained bert model and evaluation data. In *PML4DC at ICLR 2020*, 2020.

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