Universidad de Buenos Aires
Facultad de Ciencias Exactas y Naturales
Departamento de Computación

Word-embeddings contextualizados para detección de entidades nombradas en textos de radiología en español

Tesis de Licenciatura en Ciencias de la Computación

Manuel J. J. Costa

Directora: Viviana Cotik

Buenos Aires, Marzo 2025

WORD-EMBEDDINGS CONTEXTUALIZADOS PARA DETECCIÓN DE ENTIDADES NOMBRADAS EN TEXTOS DE RADIOLOGÍA EN ESPAÑOL

La creciente digitalización de los procesos médicos ha generado una gran cantidad de datos textuales, como informes de estudios clínicos, que permiten mejorar procesos a través de la automatización de la extracción de información de los mismos. Sin embargo, esta última presenta desafíos significativos, especialmente en español, debido a la escasez de recursos en este idioma y al uso de vocabulario especializado. También algunos de estos textos suelen tener abreviaturas, errores ortográficos y de tipeo, lo que agrega una complejidad adicional. Este trabajo busca contribuir al campo del Procesamiento del Lenguaje Natural Biomédico (*BioNLP*, por sus siglas en inglés) mediante el desarrollo de mejores representaciones de textos que optimicen la extracción de información para informes clínicos escritos en español.

En esta tesis se desarrolla un estudio sobre el uso de *word embeddings* y modelos de lenguaje para informes de ecografía escritos en español. Se proponen y evalúan diferentes modelos de *embeddings*, incluyendo técnicas estáticas como FastText y modelos basados en arquitecturas contextuales como *Transformers* y BiLSTM. Los *embeddings* se entrenaron utilizando un corpus anonimizado de más de 80 mil informes de ecografías. Se realizaron dos tipos de evaluaciones sobre los *embeddings*: una extrínseca y una intrínseca. Para la evaluación extrínseca se utilizó la tarea de reconocimiento de entidades nombradas con el conjunto de datos de la competencia SpRadIE. Además, se realiza un estudio de ablación para intentar establecer un vínculo más directo entre el uso de las representaciones y el rendimiento obtenido por los modelos. Para la evaluación intrínseca, se presenta un marco basado en análisis cualitativo, para medir la calidad de los *embeddings* en dominios donde no existen *benchmarks* estandarizados.

Los resultados obtenidos muestran mejoras sobre el estado del arte para la tarea de reconocimiento de entidades nombradas de SpRadIE, destacando el impacto de usar representaciones contextuales adaptadas al sub-dominio específico de la tarea. Observamos que los mejores resultados del reconocimiento de entidades nombradas se obtienen con modelos basados en Transformers, sin embargo las representaciones generadas a partir de BiLSTM parecen capturar información semántica más rica, como evidencian los estudios de ablación y el análisis cualitativo.

Palabras claves: *Embeddings*, reconocimiento de entidades nombradas, BioNLP en español, informes clínicos, ecografías, *transformers*, BiLSTM, FastText

CONTEXTUALIZED WORD-EMBEDDINGS FOR NAME ENTITY RECOGNITION IN SPANISH RADIOLOGICAL TEXTS

The growing digitalization of medical processes has generated a large volume of textual data, such as clinical study reports, which allow for process improvements through the automation of information extraction. However, this task presents significant challenges, especially in Spanish, due to the scarcity of resources in this language and the use of specialized vocabulary. In addition, some of these texts often contain abbreviations, spelling errors, and typos, adding an extra layer of complexity. This work aims to contribute to the field of Biomedical Natural Language Processing (BioNLP) by developing better text representations to enhance information extraction from clinical reports written in Spanish.

This thesis presents a study on the use of word embeddings and language models for ultrasound reports written in Spanish. Different embedding models are proposed and evaluated, including static techniques such as FastText and models based on contextual architectures like Transformers and BiLSTM. The embeddings were trained using an anonymized corpus of more than 80,000 ultrasound reports. Extrinsic and intric evaluation methods were carried out. For the extrinsic evaluation, the chosen task was named entity recognition, using the dataset from the SpRadIE competition. Additionally, an ablation study was conducted to better understand the relationship between the use of these representations and the performance of the models. For the intrinsic evaluation, a framework based on qualitative analysis is presented to assess the quality of the embeddings in domains where standardized benchmarks are not available.

The results show improvements over the state of the art for the named entity recognition task in SpRadIE, highlighting the impact of using contextual representations tailored to the specific subdomain. We observe that the best results are obtained with Transformer-based models; however, the representations generated by BiLSTM appear to capture richer semantic information, as evidenced by the ablation studies and qualitative analysis.

Keywords: Embeddings, named entity recognition, Spanish BioNLP, clinical reports, ultrasound, transformers, BiLSTM, FastText

AGRADECIMIENTOS

A mi papá (QEPD), que desde siempre me inculcó el amor por el aprendizaje.
A Bety (QEPD), que me enseñó que las matemáticas podían ser bellas.
A Lubni, por acompañarme y soportarme en todo este largo proceso.
A Vivi, por guiarme y empujarme para adelante siempre.
A mis amigos y compañeros de cursada, con los que nos apoyamos a lo largo de esta hermosa e intensa carrera.
A mi mamá, por estar ahí cuando la necesité.
A la UBA, que me acompaña desde que tengo doce años y que, de una forma u otra, es una parte de mi.

Índice general

1. INTRODUCCIÓN

En este capítulo presentamos las razones que motivan la realización del presente trabajo, introducimos algunas nociones básicas, cuál es el problema que se quiere estudiar y cuáles son los desafíos que se presentan. Por último se describe la estructura del resto del informe.

1.1. Motivación

En los últimos años ha tenido lugar un crecimiento significativo en la cantidad de textos digitalizados y, por lo tanto, en la cantidad de información valiosa que se puede extraer de ellos. En particular en la medicina hay un impulso importante a la digitalización de los procesos. En Argentina esto se da de la mano de proyectos gubernamentales como la Estrategia de Salud Digital[1].

Durante el desarrollo de la actividad médica se producen, entre otro tipos de información, distintos textos. Podemos mencionar artículos científicos, notas escritas en la historia clínica -como notas de visita, informes de altas médicas (epicrisis)- e informes de estudios como radiografías y ecografías. Entre otros beneficios que se pueden obtener del procesamiento automático de estos textos podemos nombrar:

- La contribución a la construcción de estadísticas útiles para estudios clínicos, que podría impactar en diagnósticos y tratamientos.
- La posibilidad de mejorar la atención a un paciente, por ejemplo, permitiendo que un profesional reciba inmediatamente una alarma en caso de detectarse un resultado que requiera atención urgente.

El formato informal y semi-estructurado (en el mejor de los casos) de estos textos hace que la extracción de información no sea trivial. Los mismos usan un vocabulario específico, que varía con el dialecto de cada país, hospital, especialidad, e incluso a veces de médico a médico. Distintos términos pueden referir al mismo concepto y el mismo término puede tener varios significados. Presentan también abundancia de acrónimos y abreviaturas. En algunos casos (como por ejemplo, en las notas que se escriben durante las consultas) la formalidad no es la principal prioridad sino que es más importante que la escritura sea rápida, lo que conlleva a que ocurran faltas de ortografía o errores de tipeo.

Este punto se puede ilustrar mejor mostrando una oración tomada de un informe (en este caso, de una ecografía):

```
RD Diam Long: 8.5 cm RI Diam Long: 9cm A nivel de FID se observan estructuras ganglionares de forma y ecogenicidad conservada de 1 cm de diametro AP.
```

La mayor parte de la investigación sobre el procesamiento automatizado de textos biomédicos suele ser en inglés[2]. Sin embargo, es conveniente analizar los textos en el

[1] https://www.argentina.gob.ar/salud/digital, consultado en abril de 2024.

[2] Ver Névéol et al.[48] para una comparación detallada de la producción de artículos científicos sobre el tema en diferentes idiomas.

idioma en el que fueron escritos. Dado que el español es uno de los idiomas más hablados del mundo [21] resulta necesario tener herramientas para trabajar con recursos en ese idioma.

Si bien muchos métodos desarrollados en inglés pueden ser utilizados en otros idiomas [56], resulta necesario adaptarlos a las características de cada uno. Una de las principales herramientas que ha sido clave en el avance del *procesamiento del lenguaje natural* son los *word embeddings* (ver Sección 2.3) pero para su correcto funcionamiento es necesario que sean entrenados con el vocabulario del idioma y dominio en que se los quiere aplicar.

Trabajar en español, además, presenta el desafío adicional de conseguir datos. No hay una gran disponibilidad de *corpora*[3] públicos para la comunidad de investigación en procesamiento del lenguaje. Obtener informes médicos para experimentar no es fácil, ya que no son públicos, porque eso afectaría la privacidad de los pacientes. La disponibilidad de *corpus* anotados[4] es aún menor, ya que se requiere de una gran cantidad de horas de trabajo por parte de anotadores expertos. El conjunto de datos utilizado en este trabajo fue publicado en el año 2017 y se describe en la Sección 3.2.

1.2. Definiciones

Se llama lenguaje natural al lenguaje hablado y escrito por seres humanos. El procesamiento del lenguaje natural (NLP por sus siglas en inglés) es un área de estudio multidisciplinaria que se encarga de procesarlo para analizarlo o generarlo [35]. *Para este fin, una de las herramientas que suele emplearse con mayor frecuencia son técnicas de aprendizaje automático (o Machine Learning).*

Dentro del procesamiento del lenguaje natural, se conoce como BioNLP (Biomedical NLP) al área que se encarga específicamente de textos relacionados con la biología y la medicina. Es importante destacar que incluso dentro de este campo hay una gran diversidad en cuanto al tipo de textos que podemos encontrar: artículos científicos, enciclopedias, historias clínicas informes de diversos estudios médicos, e informes de alta médica, entre otros. Además existen una amplia diversidad de especialidades médicas y áreas de estudio a las que pueden pertenecer estos textos. Cada una de estas combinaciones definen diferentes *sub-dominios* dentro del *dominio* biomédico.

En el contexto de NLP, el concepto de dominio se refiere al área temática o campo específico al que pertenece un *corpus* o problema. Cada dominio tiene características propias, como vocabulario especializado, patrones lingüísticos y tipos de información predominantes, que deben ser consideradas. Por ejemplo, el dominio biomédico incluye terminología médica y abreviaturas clínicas, mientras que el dominio financiero abarca términos económicos y métricas de mercado. Los dominios además suelen ser jerárquicos, por lo que se puede decir que un dominio es un sub-dominio de otro cuando el primero engloba un subconjunto de los textos del segundo. Por ejemplo, dentro del dominio biomédico podríamos identificar el sub-dominio clínico (referente a la actividad clínica-hospitalaria), y a su vez dentro de este un sub-dominio como el de la radiología, el cual es aun más acotado y específico (términos relacionados con imágenes diagnósticas y mediciones anatómicas).

[3] Un *corpus* (pl. *corpora*) es una colección de textos o discursos utilizados para un propósito específico, que puede estar anotado o no.

[4] Un *corpus* "anotado" o "etiquetado" es un conjunto estructurado de textos marcados con información adicional que describe aspectos lingüísticos como la morfología, sintaxis, y semántica. Estos metadatos facilitan el análisis computacional del texto, permitiendo aplicar diversos métodos.

La extracción de información es un sub-campo de NLP que consiste en técnicas para obtener información estructurada a partir de fuentes no-estructuradas o semi-estructuradas. A partir de estos datos estructurados resulta más fácil realizar análisis, búsquedas y visualizaciones.

El reconocimiento de entidades nombradas (NER, de *named entity recognition*), en donde se identifican y clasifican términos en categorías predefinidas, como nombres de personas, organizaciones, lugares y fechas, es una tarea que ayuda en la extracción de la información. Algunas otras tareas del procesamiento del lenguaje natural son la detección de relaciones, en donde se identifican y extraen las relaciones semánticas que existen entre entidades nombradas presentes en el texto; y la clasificación de textos, en donde se asigna una categoría o etiqueta a un texto.

En la actualidad todas estas tareas se apoyan en el concepto de *word embeddings* (WE), también llamados simplemente *embeddings*, como forma de pre-procesar las palabras[5] de los textos a analizar. Existen muchas variantes en las técnicas de generación de WE, que detallamos oportunamente en la sección 2.3, pero a grandes rasgos todas buscan obtener una *representación vectorial densa*[6] de las palabras que componen un *corpus*. Collobert et al. muestran por primera vez en su estudio *Natural language processing (almost) from scratch* [14], que estas representaciones logran dos objetivos fundamentales: su uso permite mejorar el desempeño en numerosas tareas y además proporcionan una alternativa más sencilla y generalizable (a diferentes tareas) respecto de la ingeniería de características manual (del inglés, *feature engineering*). Este último punto está íntimamente relacionado con el hecho de que las técnicas para obtener *embeddings* no requieren texto anotado, lo cual es significativo ya que, para la mayoría de aplicaciones, la disponibilidad de texto sin anotar es mayor al anotado.

A lo largo del tiempo, los *embeddings* han evolucionado desde modelos simples con representaciones estáticas como *word2vec* [43], *Glove* [52] y FastText [7] hasta representaciones más avanzadas y contextualizadas (lo que significa que la representación vectorial de una palabra concreta va a depender del resto de las palabras que la rodean en la oración).

Por último, los *embeddings* en muchos casos son entrenados sobre ciertos *corpora* y luego son aplicados en tareas que utilizan un *corpus* con un volumen de texto menor (en general uno o más órdenes de magnitud) o que incluso pertenecen a un dominio diferente. Esta idea se basa en el concepto de *transfer learning* que consiste en la hipótesis de que en ciertos casos el aprendizaje que realiza un modelo a partir de un problema o conjunto de datos puede ser de utilidad al aplicarse en otro (problema o conjunto de datos). Esta idea de ajustar los parámetros de un modelo entrenado previamente (o *pre-entrenado*) a un problema o conjunto de datos más específico respecto al original también suele llamarse *fine-tuning*. Profundizamos sobre estos conceptos en 2.7.

En este trabajo nos centramos en abordar el concepto de *embeddings* y probar distintas técnicas para un *corpus* que consiste en informes de estudios ecográficos en español. Además, nos apoyaremos en una tarea de NER concreta, sobre un subconjunto anotado de este *corpus*, como forma de evaluar la efectividad de las representaciones obtenidas.

[5] Aunque es común realizar *embeddings* al nivel de palabras, no es la única unidad "embebible": componentes más pequeños como sub-palabras (partes que componen una palabra), o más grandes como oraciones o documentos pueden ser embebidos también.

[6] Si bien no hay una definición formal, se suele decir que un vector es *denso* cuando la mayoría de sus valores son distintos de cero. Esto es en oposición a los llamados vectores *dispersos* (*sparse*, en inglés) que se caracterizan estar compuestos casi totalmente de ceros.

1.3. Descripción del problema

La Figura 1.1, extraída y traducida al español del trabajo de Khattak et al. [38], muestra el flujo de tareas y entrada de datos habituales para el desarrollo de *embeddings*. A continuación se explica brevemente en qué consiste cada paso y como se abordan en el presente trabajo. Siguiendo la terminología empleada en el trabajo citado, nos referiremos como "tarea objetivo" a nuestra tarea de NER sobre el *corpus* específico de informes ecográficos.

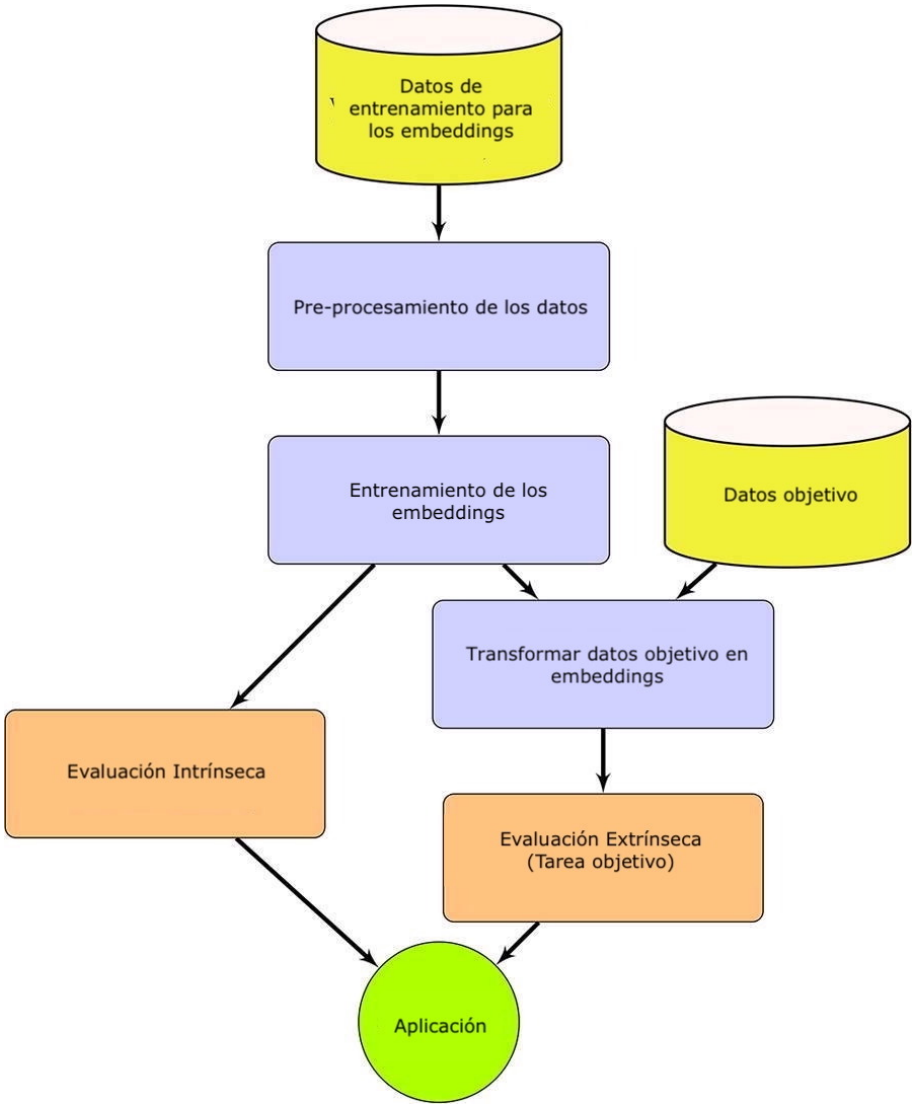

Fig. 1.1: Proceso de desarrollo de *Word embeddings*. Imagen tomada y traducida al español de Khattak et al.[38].

El primer paso es definir cuál es el *corpus* que se va a utilizar para el entrenamiento (o *fine-tuning* de los *embeddings* a desarrollar). En esta etapa se busca conseguir un *corpus* perteneciente al dominio objetivo, o en su defecto un dominio lo más afín posible. Cuanto más grande y variado sea el *corpus*, más rico resultará el entrenamiento de los *embeddings* y mayor será su capacidad de generalización. En este punto no es necesario que el *corpus*

esté anotado. En nuestro caso utilizamos un *corpus* de 85621 informes correspondientes a ecografías en español (sin etiquetar) que fueron presentados en la tesis doctoral de Cotik [18].

El siguiente paso es pre-procesar los datos. En nuestro caso por estar trabajando con información sensible de pacientes y médicos realizamos una primer etapa de anonimización donde, de manera automática, se eliminan nombres propios y matrículas médicas. Hecho esto, se realiza una segmentación del texto llamada *tokenización* (ver sección 2.1) que resulta en el formato esperado por los modelos que utilizaremos luego. Es interesante notar que no existe una forma única de realizar esta segmentación y suele variar según el tipo de modelo, por lo que en nuestro caso, donde entrenamos varios, es necesario realizar más de una vez la tokenización.

Para el entrenamiento de los WE utilizamos a nivel general dos variantes: *embeddings* estáticos basados en FastText (ver sección 2.3.1) y *embeddings* dinámicos (o contextuales) basados en Transformers 2.6. Hay una diferencia importante para señalar aquí en cuanto al resultado que se obtiene en este paso según el tipo de *embedding* empleado. Los *embeddings* estáticos generan un *mappeo* fijo entre palabras del vocabulario y representación vectorial. Sin embargo esto no es posible para los *embeddings* contextuales, pues no existe una única representación para cada palabra o *token* sino que va a depender del resto de palabras que la acompañan en un contexto específico. El resultado, entonces, para este tipo de *embeddings* es un modelo en sí mismo, que debe ser ejecutado sobre la palabra y su contexto para poder obtener una representación.

Una vez los *embeddings* han sido entrenados es necesario poder evaluar su efectividad. Para esto existen dos enfoques posibles: evaluaciones intrínsecas y extrínsecas.

Las primeras buscan evaluar la calidad de los *embedding* en sí mismos, tratando de evaluar cuán bien logran capturar la semántica de las palabras a través de su representación vectorial. Como métricas principales suelen usarse la *similitud* [7] y *relatedness*[8]. Para el cálculo de estas métricas existen algunos *benchmarks* que suelen usarse en la literatura, generalmente basados en la capacidad de producir analogías o respetar sinonimia. Sin embargo, estos *benchmarks* presentan dificultades relacionadas con los idiomas y dominios que abarcan, las cuales profundizaremos en el capítulo 3.

La evaluación extrínseca consiste, en cambio, en analizar la efectividad de los *embeddings* a través del "desempeño" que ayudan a conseguir en la tarea objetivo (también llamada en la literatura en inglés como *downstream task*). Este método es mucho más simple y directo que la evaluación intrínseca, pero como desventaja no permite analizar el impacto de los *embeddings* de una forma aislada (ya que es otro modelo el que termina entrenándose para tomar las decisiones). También tiene el problema de que no permite concluir sobre la utilidad de los *embeddings* para otras tareas distintas a la que se está evaluando. Por lo tanto, ambos enfoques (intrínseco y extrínseco) son complementarios y necesarios si lo que más nos interesa es evaluar la calidad de los *embeddings*.

Para la tarea objetivo (NER en este caso), se necesita que los datos estén etiquetados, a diferencia de lo que sucedía con el *corpus* para los *embeddings*. Nuestros datos objetivo

[7] La "similitud de los *embeddings*" es una métrica que refleja cuán cerca están en el espacio vectorial palabras con significados o usos similares. No hay una única forma de calcularla y por lo general se utiliza la llamada similitud coseno.

[8] "Relatedness" es una métrica que refleja cuán cerca están en el espacio vectorial palabras que están relacionadas en un contexto más amplio (no necesariamente sinónimos o similares en significado). Por ejemplo, las palabras "doctor" y "hospital" podrían considerarse altamente relacionadas, aunque no sean sinónimas.

son obtenidos de la competencia *SpRadIE* [16], que se realizó en el marco de la *Conference and Labs of the Evaluation Forum 2021* (CLEF 2021) y cuya tarea fue la detección de diez entidades nombradas (entre ellas entidades anatómicas, ubicaciones, hallazgos clínicos, medidas, etc.) a partir de textos correspondientes a informes de radiología. Este *corpus* cuenta con 473 informes escritos en español, que ya han sido previamente pre-procesados para remover datos identificatorios. Estas características lo hacen ideal para nuestro estudio. Es importante destacar que estos informes son un subconjunto del *corpus* que se utilizó para entrenar los *embeddings* (aunque se incluyen algunos procesamientos adicionales que el otro no tiene). Estos datos vienen con formato *Brat Standoff*[9] que no nos sirve para entrenar directamente al modelo de reconocimiento de entidades nombradas. Por lo tanto debemos transformarlo al formato *BIO*[10] o similar, ya que éste es el soportado por las herramientas que utilizaremos. Una vez entrenados los modelos, se llevará el resultado de las predicciones nuevamente al formato *Brat Standoff* para poder computar las métricas. Esta transformación no es trivial ya que este formato permite etiquetar entidades que aparecen discontinuas en el texto, así como también identificar casos donde dos o más entidades se solapan en un mismo fragmento. Estas dos cosas no son posibles con el formato BIO, por lo que debemos lidiar con esta pérdida de información de la mejor forma posible. Estos temas los profundizamos en la sub-sección 4.2.2.

Una vez resuelto el formato de los informes de la tarea objetivo, se debe aplicar a cada uno la transformación aprendida anteriormente para obtener los *embeddings* correspondientes. Los mismos luego se utilizan para entrenar el modelo de reconocimiento de entidades nombradas, al cual evaluamos con las métricas de interés para comparar con los modelos que participaron en la competencia.

1.4. Contribución

En este trabajo nos centramos en el desarrollo de *embeddings*, tanto contextuales como no contextuales, para informes de ecografías en español. Esto en sí mismo resulta algo novedoso pues a nuestro conocimiento no hay otro trabajo que haya hecho esto para este sub-dominio específico en español.

Además con nuestros *embeddings* logramos mejorar el estado del arte actual para la tarea de NER sobre el *corpus* de SpRadIE. Y particularmente esto lo logramos con una solución basada en un único modelo de *deep learning*, mientras que las mejores soluciones publicadas hasta ahora dependían de varios sub-modelos similares al nuestro.

Además, como parte del proceso de desarrollo se llevó adelante una meticulosa anonimización del *corpus* con el cual se entrenaron los *embeddings*. Esto posibilita que, con las debidas autorizaciones, pudiera considerarse la publicación para la comunidad de BioNLP tanto del *corpus* como de los *embeddings* aquí entrenados.

Por último, planteamos las bases de un marco de evaluación intrínseca que puede ser utilizada en forma general en escenarios como el nuestro, en donde no se dispone de un *benchmark* adecuado.

[9] Formato brat standoff: `http://brat.nlplab.org/standoff.html`

[10] El formato BIO es una convención de etiquetado comúnmente utilizada en el reconocimiento de entidades nombradas (NER). Cada etiqueta consiste en una combinación de un prefijo (B-, I- o O-) y una categoría de entidad. B- indica el comienzo de una entidad, I- su continuación y O- señala que la palabra no es parte de ninguna entidad.

1.5. Organización del trabajo

El presente documento está dividido en seis capítulos. En el capítulo 2 (Marco Teórico) se desarrollan los conceptos necesarios para profundizar en la comprensión del problema y de los métodos propuestos. En el capítulo 3 (Antecedentes y revisión de la literatura) se analizan otros trabajos relacionados a los problemas que nos interesa tratar aquí. En el capítulo 4 (Metodología) se desarrolla una descripción de las técnicas aplicadas para la preparación de los datos, el entrenamiento de los *embeddings* y su evaluación. En el capítulo 5 (Resultados de Evaluación Extrínseca) se presentan los resultados obtenidos en la experimentación extrínseca para cada uno de los métodos utilizados y se realiza un análisis de los mismos. En el capítulo 6 (Análisis cualitativo de los *embeddings*) se describen algunos hallazgos obtenidos durante la exploración cualitativa de los *embeddings* generados. Por último, en el Capítulo 7 (Conclusiones y trabajo futuro) se desarrollan las conclusiones obtenidas en este trabajo y se plantean posibles trabajos futuros.

2. MARCO TEÓRICO

En este capítulo se introducen algunos conceptos fundamentales para la comprensión de los temas tratados en este trabajo.

2.1. Tokenización y vocabulario

La tokenización es la tarea de segmentar texto en unidades más pequeñas, a las que denominamos *tokens*. Estos *tokens* pueden ser oraciones, palabras individuales o sub-unidades más pequeñas, como caracteres o *sub-palabras* (del inglés *sub-word*, un fragmento de la palabra). En algunos casos incluso se segmenta al nivel de los bytes que componen los caracteres de una palabra [65].

La tokenización es el primer paso para descomponer el texto en elementos discretos y es necesaria en casi todos los métodos y tareas de NLP, desde los más clásicas hasta el estado del arte.

Estos *tokens* se utilizan para conformar el *vocabulario*, que es simplemente un conjunto de *tokens* distintos en un *corpus* (notar que no necesariamente son todos los posibles *tokens* distintos). Los modelos de *Machine Learning* empleados en NLP solo pueden procesar *tokens* que pertenezcan a su vocabulario asociado (que fue definido al momento del entrenamiento). Cualquier *token* por fuera del vocabulario (OOV por *out of vocabulary*) que se quiera procesar generará un error.

Las principales técnicas que existen para tokenizar un texto son:

- Tokenización en *palabras*: Típicamente los *tokens* se obtienen a partir de considerar las palabras separadas por espacio o por signos de puntuación. Pueden haber ligeras variaciones, por ejemplo si se consideran los signos de puntuación como *tokens* o si se descartan.

- Tokenización en *caracteres*: Simplemente cada carácter es un *token* por lo que el proceso es simple. Los espacios en blanco se pueden modelar con *tokens* especiales y los signos de puntuación en general tienen que ser incluidos como *tokens* también.

- Tokenización en *sub-palabras*: Este es el caso más complejo de los tres, ya que existen muchas formas en las que una palabra puede subdividirse. Por ejemplo, una opción sería dividir la palabra en sílabas. Pero también podría dividirse en prefijo, sufijo y raíz. O incluso simplemente en *sub-strings* arbitrarios. Más adelante detallamos algunos métodos que son utilizados por los algoritmos que se desarrollan aquí.

Notar que cada uno de estos métodos puede generar vocabularios completamente diferentes para un mismo *corpus*. Todos tienen sus ventajas y desventajas, y en general giran en torno a tres aspectos principales: capacidad de capturar información, manejo de los *tokens* OOV y uso de la memoria. Es importante destacar que el idioma sobre el cual se quiera trabajar juega un papel fundamental también. Por ejemplo en chino es mucho más complejo segmentar a nivel de palabras que en español o inglés por las características propias del idioma. Además como muestran Li et al. en [40], en chino los caracteres suelen ser una mejor unidad semántica que las palabras. Por lo tanto, las características que presentamos a continuación son pensando en el idioma español.

En general se considera que la segmentación por palabras tiene mayor información semántica, aunque resulta poco eficiente en el uso de la memoria (el tamaño del vocabulario puede estar fácilmente en las decenas o cientos de miles) y por su naturaleza es más sensible a fallar por *tokens* fuera del vocabulario.

Por su parte la tokenización por caracteres viene a resolver los problemas de la tokenización por palabras. El tamaño del vocabulario es mucho más acotado (con 256 *tokens* basta para escribir cualquier oración en español si pensamos en el código ASCII) y por diseño resulta casi imposible encontrarse un *token* fuera del vocabulario. Como contrapartida, generalmente el carácter como *token* no transporta ninguna semántica en sí mismo.

Lo tokenización por sub-palabra busca ser un punto intermedio entre los casos anteriores. En general los distintos métodos buscan optimizar la relación entre cantidad de *tokens* que componen el vocabulario y su granularidad. De esta forma se consiguen *tokens* con mayor carga semántica que los caracteres, aun manteniendo acotado el tamaño del vocabulario.

A continuación, explicamos resumidamente el funcionamiento de dos métodos de tokenización por sub-palabra que son utilizados por los modelos de lenguaje (ver definición en la sección 2.4) que utilizamos en este trabajo.

2.1.1. Byte-Pair Encoding y Word Piece

Los métodos conocidos como *Byte-Pair Encoding* (BPE) y *Word Piece* (WP) son dos de los más conocidos y utilizados en la actualidad. Ambos tienen un funcionamiento muy similar, razón por la cual los explicamos en conjunto aclarando en los lugares donde difieren.

BPE es originalmente un algoritmo de compresión que fue adaptado para tokenización por Sennrich, Haddow y Birch [58]. Su objetivo es reducir la cantidad de *tokens* necesarios para representar un *corpus* grande de texto. BPE comienza con un vocabulario inicial que incluye todos los caracteres individuales presentes en el *corpus*. Luego, en cada iteración, se encuentran y fusionan los pares de caracteres adyacentes más frecuentes en el texto. Este proceso se repite hasta alcanzar un número predefinido de *tokens* en el vocabulario o hasta que no queden pares frecuentes para fusionar. De esta manera, BPE crea *tokens* de longitud variable que pueden ser tan cortos como un carácter individual o tan largos como una palabra, permitiendo una representación compacta del texto que reduce la probabilidad de que se encuentren *tokens* por fuera del vocabulario.

WP surge originalmente del campo del reconocimiento de voz y es usado en NLP por primera vez por Wu et al. en [67]. El funcionamiento es esencialmente el mismo que el de BPE, siendo la principal diferencia el criterio escogido para unificar dos *tokens*: en lugar de utilizar el criterio de "par más frecuente" se utiliza el par que maximiza la siguiente ecuación:

$$frec\_ponderada = \frac{frec(token_1, token_2)}{frec(token_1) \times frec(token_2)}$$

en donde $frec$ es la frecuencia (cantidad de ocurrencias) de un *token* o de un par de *tokens*. Una interpretación probabilística de esta ecuación puede encontrarse en [67]. Intuitivamente, podemos pensarlo como que estamos evitando favorecer combinaciones solo porque dos *tokens* son muy frecuentes por separado.

2.2. Reconocimiento de Entidades Nombradas

El *reconocimiento de entidades nombradas* (NER, por sus siglas en inglés) es una tarea de extracción de información[1], cuyo objetivo es reconocer y clasificar porciones del texto que corresponden a categorías predefinidas (entidades nombradas). Estas categorías dependerán del dominio del problema que se busca resolver. En el contexto médico, algunas categorías comunes incluyen nombres de estructuras anatómicas, hallazgos clínicos, procedimientos médicos y valores de medición.

Por ejemplo, dada la siguiente oración extraída de un informe de ecografía:

Ejemplo 2.2.1. "Ambos riñones de estructura conservada."

Un sistema de NER podría identificar y clasificar las entidades de la siguiente manera:

Ejemplo 2.2.2. Ambos riñones `[ANATOMICAL_ENTITY]` de **estructura conservada** `[FINDING]`.

En este ejemplo, el modelo de NER ha identificado correctamente:

- **Ambos riñones** como una `[ANATOMICAL_ENTITY]` (estructura anatómica).
- **estructura conservada** como un `[FINDING]` (hallazgo clínico).

Este proceso de identificación y clasificación permite a los sistemas de procesamiento del lenguaje extraer información estructurada a partir de datos textuales no estructurados, lo que podría facilitar tareas como la búsqueda de información relevante para los médicos, la generación de resúmenes clínicos y la asistencia en toma de decisiones.

La tarea de NER es de tipo supervisado, lo que significa que requerimos un conjunto de ejemplos etiquetado con los que esperamos que el modelo de aprendizaje automático aprenda. En el contexto de extracción de la información suele usarse el término *anotación* para referirse a tales etiquetas. Pustejovsky y Stubbs definen las anotaciones como "cualquier etiqueta de metadatos utilizada para marcar elementos del conjunto de datos" [54]. Como bien señalan en su trabajo, es fundamental que tales anotaciones sean precisas y relevantes para la tarea que se espera resolver con Aprendizaje Automático.

2.3. Embeddings

En esta sección profundizamos en el concepto central de esta tesis: los *embeddings*, que no son otra cosa más que una representación vectorial de las palabras. Es decir que a cada palabra de un *corpus* se le asigna un respectivo vector dentro de un espacio vectorial (generalmente euclidiano). En realidad en un sentido estricto los *embeddings* no se generan necesariamente sobre palabras sino sobre un vocabulario de *tokens*, que como hemos visto pueden ser palabras pero también caracteres o sub-palabras. Por simplicidad en lo que sigue de esta sección asumiremos que siempre hablamos de *embeddings* de palabras (word embeddings) salvo que se aclare lo contrario.

[1] Las tareas de *extracción de información* son aquellas que se refieren al proceso de identificar y extraer automáticamente información estructurada y significativa a partir de texto no estructurado. El objetivo principal de la extracción de información es convertir el texto en una forma que sea más fácil de procesar e interpretar por las computadoras.

Existen diversas técnicas para realizar esta asignación de vectores (de las cuales comentaremos algunas más adelante) pero todas se basan en intentar capturar la distribución de las palabras en los textos analizados, de tal forma que palabras que tienden a aparecer en *contextos* similares (es decir, que co-ocurren frecuentemente con las mismas palabras) estén más cerca entre si en el espacio, respecto de otras palabras con las que la probabilidad de aparecer en el mismo contexto sea menor.

Como menciona Jurafsky en su libro [35], la utilidad de tales representaciones surge de la llamada *distributional hypothesis* formulada por distintos lingüistas en los años '50. Dicha hipótesis postula que palabras con significados parecidos tienden a aparecer en contextos similares, es decir tienden a co-ocurrir con las mismas palabras. Como una consecuencia de esto, si los *embeddings* logran capturar adecuadamente la distribución de las palabras en el *corpus*, entonces se espera también que logre capturar la similitud entre las mismas. Palabras con representaciones más cercanas entre sí deberían tener semánticas más similares.

¿Pero qué significa en este contexto que dos palabras tengan semánticas "similares"? A continuación resumimos la explicación dada por Jurafsky en su libro [35]. Palabras que son sinónimos (es decir, que son substituibles una por otra en una oración sin alterar las condiciones bajo las cuales la oración es verdadera) pueden considerarse naturalmente similares, sin embargo la similitud es una relación más débil que la relación de sinonimia. "Perro" y "gato" obviamente no son sinónimos pero se pueden considerar similares pues ambos son animales que además comparten ciertas características como ser mamíferos o domesticables. Además de la similaridad, existe otro tipo de relación entre palabras que suele ser interesante desde la perspectiva de la semántica vectorial: la *relatedness* (o asociación). "Taza" y "café" no son cosas similares pero sí están asociados pues suelen participar conjuntamente en los mismos *eventos*. Muchas veces esta asociación se da por pertenencia a un mismo *campo semántico*: palabras como "médico", "riñon", "ecografía" y "hospital" pertenecen al campo clínico.

Lograr representaciones del texto que puedan capturar este tipo de relaciones es valioso en sí mismo. Sin embargo, hay una motivación adicional que es fundamental para el avance del área de NLP: los *embeddings* permiten aprender de forma automática una representación conveniente para alimentar modelos para diversidad de tareas supervisadas y no-supervisadas, en lugar de tener que realizar una ingeniería de características (*feature engineering*) compleja. Previamente los mejores resultados en las tareas de NLP se obtenían a través de codificar manualmente características que capturaban propiedades específicas del texto, usualmente superficiales o que integraban cierto conocimiento propio del dominio, lo que (además de requerir tiempo y esfuerzo) las volvía difícilmente reutilizables entre tareas, dominios e idiomas. A partir del uso de *embeddings* esta tarea se simplifica y se vuelve más común la reutilización, como señalan Habibi et al. en [30] para la tarea de reconocimiento de entidades nombradas en texto biomédico. El motivo de esta simplificación es que los métodos para generar *embeddings* caen en la categoría de métodos auto-supervisados (o *self-supervised* en inglés): se obtienen a partir de resolver problemas esencialmente supervisados (como clasificación o predicción de palabra siguiente en una oración) pero sin la necesidad de recibir un etiquetado previo, pues las "etiquetas" se obtienen del texto mismo en forma automática.

Pueden establecerse dos divisiones principales para los métodos de generación de embeddings. Según la forma en la que se hace la asignación de los vectores pueden dividirse en estáticos o contextualizados. En el primer caso, la representación solo depende de la

palabra que se quiere representar (por ejemplo "banco"); en el segundo, depende no solo de la palabra sino también del contexto en el que se encuentra (por ejemplo "banco" en la oración "Me senté en el banco de la plaza" no va a tener la misma representación que en la oración "Fui a hacer un depósito al banco").

Otra división importante es la de *embeddings* ralos y densos. El primer caso usa vectores con longitudes en el orden de los miles o decenas de miles (por lo general, cada dimensión corresponde a un token del vocabulario) y donde la gran mayoría de los valores de cada vector suele ser cero. Por su parte las representaciones densas tienen un largo acotado, que suele oscilar entre 50 y 1000 dimensiones (según menciona Jurafsky [35]) y se espera que tenga pocos valores nulos.

Evolutivamente, podemos decir que el progreso de los *embeddings* ha seguido este recorrido:

- Esparsos y estáticos: matrices de co-ocurrencias, PPMI, TF-IDF
- Densos y estáticos: Word2Vec, GloVe, FastText
- Densos y contextuales: ElMo, BERT, GPT

La primer categoría si bien tiene gran importancia histórica y teórica no es el foco de esta tesis debido a que las representaciones ralas en la práctica se han visto superadas por las densas.

Más adelante en esta sección profundizaremos sobre las representaciones densas, tanto estáticas como contextualizadas. Notar que nos enfocaremos particularmente en los modelos que son usados en este trabajo, dejando de lado muchos modelos que son de gran importancia histórica como Word2Vec [43].

Hay también otra categorización muy importante cuando hablamos de *embeddings* y que resulta ortogonal a las anteriores: la granularidad del token que se embebe. En general los casos típicos son los mismos que mencionamos cuando hablamos de tokenización en la Sección 2.1: por palabra, por carácter y por sub-palabra. La elección de la granularidad del token trae consigo las ventajas y desventajas comentadas en dicha sección. Por ejemplo, los *embeddings* por palabra sufren del problema de no poder vectorizar *tokens* OOV (del inglés *out-of-vocabulary*) y de problemas de escalamiento cuando el vocabulario resulta muy grande. Por su parte aquellos que se basan en caracteres solucionan ambos problemas, pero introducen la cuestión de que los caracteres por sí solos carecen de significado. Por ese motivo solo son utilizados con *embeddings* contextuales. Por último, los *embeddings* de sub-palabras buscan ser un punto medio.

Vale notar que la granularidad del token usado limita la unidad más pequeña de texto que puede embeberse pero no así la más grande. Dicho de otra forma: a partir de, por ejemplo, *embeddings* de caracteres pueden formarse *embeddings* de palabras, de oraciones o incluso de documentos completos. Esto se logra a partir de la combinación de *embeddings* que puede realizarse de diferentes formas, como por ejemplo calculando el promedio entre vectores. De esta forma es posible hablar de Word *Embeddings* aún si el modelo usado fue entrenado con un vocabulario de caracteres o sub-palabras.

Hemos hablado entonces de qué son los *embeddings* y los distintos tipos que existen. En lo que resta de esta sub-sección veremos el caso concreto de FastText (un tipo de *embedding* denso y estático) y luego nos adentraremos a responder una pregunta fundamental ¿Cómo se puede evaluar la calidad o utilidad de estas representaciones? No cubriremos en esta sección el caso de los *embeddings* contextuales ya que para esto resulta necesario primero introducir el concepto de *modelos de lenguaje*, que serán presentados en la sección 2.4.

2.3.1. FastText

FastText es un método para generar word *embeddings* estáticos introducido por Bojanowski et al. en [7]. Su funcionamiento es similar al algoritmo de Word2Vec [43] ya que utiliza los mismos algoritmos para su implementación. La principal diferencia que introduce FastText es que antes de generar los *embeddings* realiza un pre-procesamiento del vocabulario para generar un vocabulario interno que incluye información de las subpalabras. Más específicamente genera *n-gramas*[2] a partir de las palabras del vocabulario inicial. El largo de los n-gramas a utilizar es configurable al momento de entrenar, siendo la configuración típica un mínimo de 3-gramas y un máximo de 6-gramas.

Por ejemplo, si tuviéramos la palabra "riñones" y configuráramos el mínimo y máximo de los n-gramas en 5, entonces tendríamos los siguientes 5-gramas en nuestro vocabulario: `<riño`, `riñon`, `iñone`, `ñones`, `ones>`. Adicionalmente también se agrega el token `<riñones>`. Los símbolos `<` y `>` son caracteres especiales que delimitan en principio y fin de una palabra y se agregan como forma de desambiguar. Por ejemplo, si en el vocabulario también tuviéramos la palabra "riñon" entonces vamos a tener un nuevo token `<riñon>`. Este token es considerado distinto al 5-grama `riñon` que resulta de la palabra "riñones".

La importancia de esta decisión es que permite que FastText pueda generar *embeddings* para palabras que no estaban en su vocabulario de entrenamiento, siempre y cuando dicha palabra pueda descomponerse en n-gramas que sí fueron vistos durante el entrenamiento. Esto le da una mayor versatilidad respecto a Word2Vec y es especialmente útil en situaciones donde es común que haya *typos* o la forma de escritura resulta irregular (como es el caso de nuestros informes clínicos).

2.3.2. Evaluación de embeddings

Existen a grandes rasgos dos tipos de evaluaciones posibles para los embeddings: extrínseca e intrínseca.

La evaluación extrínseca es la que se obtiene al aplicar los *embeddings* a una tarea específica de NLP. Por ejemplo: en una tarea concreta de NER o de análisis de sentimientos. En general se considera la forma de evaluación más importante (ver sección 6.12 de [35]) ya que usualmente el fin último de los *embeddings* es mejorar el desempeño de otras tareas. Sin embargo es importante notar que un *embedding* puede ser mejor para una tarea o un modelo puntual pero no para otras, por lo que no es tan simple generalizar los resultados relativos al desempeño de un *embedding* en sí mismo.

Por su parte, la evaluación intrínseca también es de utilidad. Su objetivo es tratar de cuantificar cuán bien los *embeddings* propuestos capturan relaciones semánticas dentro del texto, siendo la más común la *similitud*. La manera clásica de medir esto entre vectores es usando la llama *similitud coseno* que se define como el coseno del ángulo formado entre los vectores. Si bien el ángulo no es conocido, existe una forma de calcular esto mismo en función del producto escalar de los vectores y sus respectivas magnitudes. Más formalmente, si v y u son vectores de dimensión n y θ es el ángulo entre ellos, entonces sabemos por propiedad del producto escalar que

$$cos(\theta) = \frac{v.u}{||v||.||u||} \tag{2.1}$$

[2] Un *n-grama* es una secuencia contigua de n caracteres (o palabras) extraídos de un texto. Por ejemplo, en la palabra "gato", los bigramas ($n = 2$) serían "ga", "at", "to".

Notemos que por definición del coseno el valor está siempre en el rango [-1, 1]. El resultado es 1 si los vectores tienen misma dirección y sentido, 0 si son ortogonales, y -1 si tienen igual dirección pero sentido opuesto. Por lo que *tokens* con significados muy similares se espera que tengan valores cercanos a 1.

Otras relaciones que a veces se tratan capturar con la semántica vectorial son la *relatedness* (como se definió al principio de la sección) o las analogías. Un ejemplo paradigmático de estas últimas es el que se da en la publicación original de Word2Vec [43], donde se muestra cómo los vectores cumplen que

$$Vector(Rey) - Vector(Hombre) + Vector(Mujer) \sim Vector(Reina)$$

mostrando que pueden capturar la analogía "Rey es a hombre, reina es a mujer".

Tales relaciones si bien resultan de gran interés presentan dificultades no menores para su cómputo. La principal radica en que para computar tanto similitud, *relatedness* o analogías se requiere tener conjuntos de datos cuidadosamente armados que incluyan ejemplos con tales relaciones, típicamente conocidos como *gold standards*. En el caso de *embeddings* de dominio de uso general y especialmente en el idioma inglés donde el mayor esfuerzo ha sido puesto en los últimos años, esto puede no ser tan desafiante. Por ejemplo, WordSim [24] y SimLex [32] son dos recursos ampliamente usados que contienen relaciones de similitud y *relatedness*. Sin embargo tales *gold standards* raramente están disponibles para dominios mucho más específicos (donde incluso puede ser difícil determinar relaciones semánticas sin un especialista del dominio) o en diversidad de idiomas distintos del inglés. Esto dificulta realizar este tipo de análisis a nivel cuantitativo, quedando en principio la alternativa de hacer un análisis a nivel cualitativo, que igualmente requiere cierto conocimiento del dominio.

2.4. Modelos de Lenguaje

Un Modelo de Lenguaje o *Language Model* (LM) es un modelo probabilístico que aprende la distribución de probabilidad de secuencias de palabras en un lenguaje. Su objetivo principal es predecir la siguiente palabra en una oración dada una secuencia previa, asignando probabilidades a diferentes posibles continuaciones. Formalmente, dado un conjunto de palabras $w_1, w_2, \ldots, w_n$, el LM estima $P(w_n|w_1, w_2, \ldots, w_{n-1})$. Esta misma definición se puede generalizar para *tokens* en lugar de palabras.

Los primeros modelos de lenguaje datan de la década de 1980 como señala Rosenfeld en [57]. La evolución de tales modelos ha sido notable, siendo parte esencial en el desarrollo de tareas como resumen de texto, traducción automática, generación automática de contenido, agentes conversacionales, entre otras.

Más adelante en este capítulo se introducen las dos técnicas que más impacto han tenido en los últimos años: arquitecturas de redes neuronales basadas en *Long Short-Term Memory* (LSTM) y arquitecturas basadas en Transformers.

2.4.1. Masked Language Model

Los *Masked Language Model* (MLM) son una variante de los modelos de lenguaje diseñada para el aprendizaje auto-supervisado, introducida por Devlin et al. en el modelo BERT [19]. A diferencia de los LM tradicionales que predicen la siguiente palabra en una secuencia, los MLM ocultan (*mask*) aleatoriamente algunas palabras dentro del texto de entrada y entrenan al modelo para predecirlas a partir del contexto.

Por ejemplo, dada la oración:

Ejemplo 2.4.1. "El paciente presenta un `[MASK]` en el hígado."

El modelo debe inferir que la palabra faltante podría ser "tumor", "quiste" u otra entidad médica plausible según el contexto. Este enfoque permite a los modelos capturar representaciones más ricas del lenguaje, ya que consideran información tanto del contexto anterior como posterior a la palabra enmascarada.

Los MLM han demostrado ser altamente efectivos para el entrenamiento de modelos de lenguaje en tareas de comprensión, ya que generan representaciones profundas del significado de las palabras en diferentes contextos. Modelos como BERT, RoBERTa y sus variantes utilizan esta estrategia para aprender representaciones generalizables a múltiples tareas de NLP. Más adelante en este capítulo, ahondamos en el funcionamiento de estos modelos.

2.4.2. Obtención de *embeddings* a partir de un LM

A diferencia de los métodos de generación de *embeddings* estáticos, que se enfocan en extraer estas representaciones para poder ser usadas luego por otras tareas, en el caso de los métodos de *embeddings* contextuales el objetivo es típicamente entrenar un modelo de lenguaje (LM), utilizando alguna arquitectura de redes neuronales artificales. Este modelo es el que se ejecutará sobre el texto que se quiera embeber, sin embargo cómo hemos visto más arriba el *output* de un LM no es un *embedding* en sí mismo, sino una distribución de probabilidades sobre el vocabulario.

En estos casos, la forma clásica de obtener *embeddings* para un token, palabra, oración, o informe, es quedarnos con el *estado oculto* (*hidden state*) de la capa anterior a la capa de salida (*head*). Dicho de otra forma, nos quedamos con el *output* de la última *capa oculta* del modelo. Por ejemplo, en el caso de la arquitectura BERT (que se explica más adelante en este capítulo) esta capa posee 768 unidades, por lo que el resultado termina siendo un vector con esa cantidad de dimensiones. Una aclaración importante es que si bien quedarse con el último estado oculto es una decisión frecuente en la literatura, no es la única posible. Otra estrategia común es la de tomar las últimas N capas ocultas y combinar cada estado a través de concatenación o promedio. De hecho en el paper fundacional de BERT [19] los autores hacen un experimento en el que extraen *embeddings* para entrenar un modelo de NER, y hallan que la mejor combinación la obtienen concatenando los resultados de las últimas cuatro capas ocultas. Este resultado, sin embargo, no significa que necesariamente sea siempre la mejor configuración posible.

2.5. Arquitecturas LSTM

La arquitectura *Long Short-Term Memory* (LSTM) [33] es un tipo de Red Neuronal Recurrente (RNN) que introduce el concepto de *memoria de corto plazo* para intentar superar las principales dificultades de las arquitecturas recurrentes. Para entender un poco mejor esto, primero es necesario introducir el concepto de *dependencias de largo plazo* en una secuencia.

Tomemos como ejemplo el siguiente fragmento de texto: "Juana sintió hambre y decidió ir a su restaurante favorito. Una vez allí, ella pidió el plato del día". Notar que en la segunda oración estamos usando las palabras "ella" y "allí" para aludir indirectamente a Juana y

al restaurante (o incluso más, al "restaurante favorito de Juana"). Este tipo de relación indirecta donde la carga semántica de una palabra (o incluso una oración entera) depende de información previa es lo que se conoce como *dependencias de largo plazo* (*long-term dependencies* en inglés). Los humanos podemos realizar la asociación anterior dado que tenemos una memoria de corto plazo que nos permite recordar información clave de la primera oración al momento de leer la segunda.

Las RNN básicas luchan con este tipo de dependencias ya que la única forma que tienen de poder aprenderlas es con estados internos lo suficientemente largos como para mantener al mismo tiempo información de los términos involucrados. Por lo tanto, cuanto más distanciadas se encuentran en el texto las palabras relacionadas, más largo debe ser el estado que se mantiene. El problema que esto presenta es que este tipo de arquitectura no escala bien al momento de entrenar por dos fenómenos ampliamente estudiados en el área de Deep Learning: *vanishing gradient* y *exploding gradient* [6]. No profundizamos aquí en los detalles, pero básicamente consisten en limitaciones propias del algoritmo de optimización utilizado.

Las LSTM presentaron una respuesta a este problema introduciendo el concepto de *unidades de memoria* (referidas usualmente como unidades (o celdas) LSTM). Estas componentes tienen la capacidad de "aprender selectivamente" información del texto procesado hasta el momento, por lo que no es necesario mantener estados internos tan grandes para preservar dependencias de largo plazo. Básicamente la memoria actúa como un resumen de lo visto hasta el momento, y este resumen se traduce en un estado interno de la celda que no es otra cosa que un vector. El motivo por el que puede aprender selectivamente que cosas mantener de los estados pasados y los nuevos es que tiene un mecanismo de tres "compuertas" (*gates*): una que indica cuánto aprender del elemento actual de la secuencia (*input gate*), otra que indica cuánto olvidar del estado anterior de la propia unidad de memoria (*forget gate*), y una tercera que a partir del nuevo estado interno genera el *output* de la unidad (*output gate*). Este diseño de compuertas y estado interno puede verse en la Figura 2.1.

Los parámetros de estas tres compuertas son optimizados durante el proceso de entrenamiento junto con el resto de los pesos del modelo, con garantías de convergencia mucho mejores que las de las RNN clásicas. Sin embargo, cabe mencionar que a pesar de esta mejora las LSTM igualmente lidian con el problema del *vanishing gradient*, solo que con una tolerancia mucho mayor sobre el largo de las secuencias.

Una red neuronal con arquitectura LSTM es simplemente una red neuronal recurrente en la cuál las unidades clásicas utilizadas son reemplazadas por celdas LSTM. Sigue siendo recurrente en el sentido que se utiliza tanto el input del paso actual como el *output* del paso anterior, pero se agrega una recurrencia interna a nivel de las celdas LSTM. Más detalles de esto pueden encontrarse en el libro Deep Learning [27].

Una evolución de las redes LSTM son las llamadas BiLSTM. Este tipo de arquitectura surge de notar el hecho que muchas veces la semántica de una palabra depende fuertemente de otras que aparecen después en la oración. En estos casos una LSTM que recorre la secuencia en sentido inverso puede ser más adecuada. Las redes BiLSTM son simplemente el resultado de combinar dos LSTMs: una que recorre la secuencia de inicio a fin (*forward*) y otra que va del final hacia el principio (*backward*). El *output* se genera a partir de realizar una ponderación entre ambos resultados.

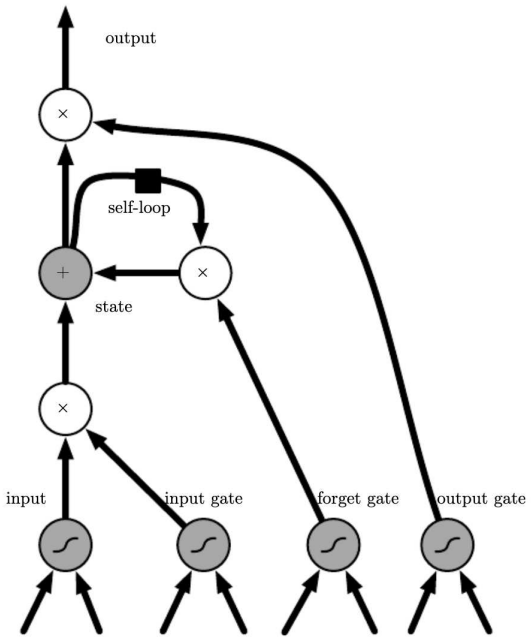

Fig. 2.1: Arquitectura interna de una celda (unidad) de memoria de la arquitectura LSTM. Puede observarse de que manera están conectados el *input* actual de la secuencia, el estado interno de la unidad de memoria, y las tres compuertas: *input gate, forget gate y output gate*. Notar el self-loop que es la conexión que permite reutilizar el estado interno del paso anterior como parte del cálculo del estado interno actual (ponderado por la *forget gate*). Esta imagen fue tomada del libreo Deep Learning [27]

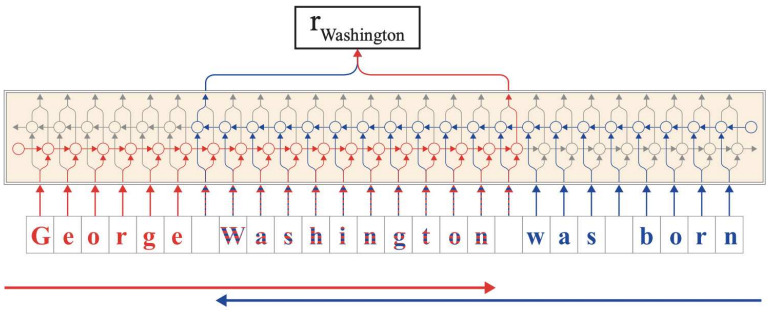

Fig. 2.2: Imagen tomada del paper original de Flair [2]. Se ilustra en este caso el proceso para obtener el *embedding* de la palabra "Washington" en el contexto de esta oración. Por un lado se ejecuta el modelo *forward* desde el comienzo de la oración hasta el último carácter de la palabra deseada (en este caso "n") y nos quedamos con el último estado interno correspondiente a procesar este carácter (es decir, un vector de 2048 dimensiones). Análogamente, se ejecuta el modelo *backward* desde el final hasta el primer carácter de la palabra (en este caso "W") y nuevamente nos quedamos con el último estado interno generado. Debido a la capacidad de la LSTM de conservar información de los estados vistos anteriormente, basta con quedarse con estos dos estados internos para tener información de la palabra y todo su contexto.

2.5.1. Flair

Flair es una arquitectura puntual de BiLSTM para modelos de lenguaje que fue introducida por Akbik, Blythe y Vollgraf [2]. Entre sus principales características se encuentran que utiliza tokenización por caracteres (lo que le da mayor robustez frente a palabras por fuera del vocabulario de entrenamiento) y que se entrenan dos modelos de lenguaje independientes con 2048 unidades de LSTM cada uno: uno *forward* y otro *backward*.

Esta arquitectura fue pensada desde el principio como una forma de extraer *word embeddings* contextuales para tareas de *anotación de secuencias* (como por ejemplo, NER). Notar que si bien los modelos procesan la secuencia a nivel de sus caracteres, el objetivo es generar *embeddings* por palabra. La forma en la que se realiza esto es la que se describe en la figura 2.2. El resultado consiste en un vector de 4096 dimensiones que contiene información del contexto anterior y posterior de la palabra en la oración.

2.6. Arquitecturas de Transformers

Los *Transformers* representan una clase de arquitecturas de modelos de aprendizaje profundo que han emergido como un componente esencial en numerosas aplicaciones de NLP. Fueron introducidos por Vaswani et al. [64], donde se introduce el componente clave de esta arquitectura: el mecanismo de *auto-atención* (*self-attention* en el original) como forma de reemplazar el uso de redes neuronales recurrentes. Las RNN si bien son arquitecturas muy poderosas para trabajar con datos secuenciales, presentan grandes desafíos para la convergencia de su entrenamiento, como muestran Bengio, Simard y Frasconi [6]. Además, las arquitecturas basadas en RNN son intrínsecamente más difíciles de paralelizar (por su naturaleza secuencial) y de escalar para mayores volúmenes de datos. Gracias al uso de las capas de auto-atención, los *Transformers* logran modelar dependencias a largo plazo en datos secuenciales (y además lo hacen de manera bidireccional) pero manteniendo

un nivel de paralelismo similar al de las redes neuronales no-secuenciales.

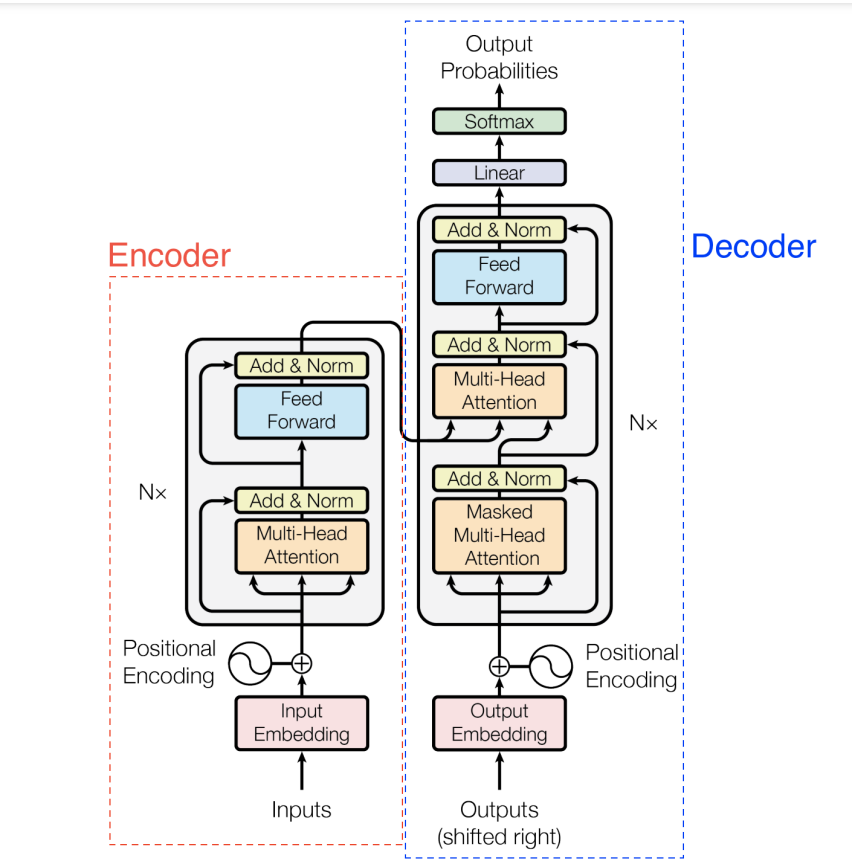

Fig. 2.3: Imagen tomada y adaptada de [64], ilustra la arquitectura básica de un *Transformer*. Lo recuadrado en rojo (a la izquierda) es el *encoder*, mientras que lo recuadrado en azul (derecha) es el *decoder*.

La arquitectura de los *Transformers* consta principalmente de dos partes fundamentales: el codificador y el decodificador. En el contexto de tareas del tipo secuencia-a-secuencia (por ejemplo, la traducción automática) el codificador toma una secuencia de entrada (como una oración en inglés) y la convierte en una representación numérica (vectorial). Por su parte, el decodificador utiliza esta representación para generar una secuencia de salida (por ejemplo, la oración traducida al español).

Este enfoque innovador (fuertemente inspirado por técnicas del campo de *Computer Vision*) ha demostrado ser altamente efectivo en una variedad de tareas de NLP, superando en muchos casos a las arquitecturas basadas en LSTMs. Para comprender mejor a los *Transformers* a continuación profundizaremos en algunos detalles técnicos.

2.6.1. Auto-atención

La auto-atención es un componente esencial de los *Transformers* que permite evaluar la importancia de cada palabra en una secuencia en función de su contexto. Este mecanismo opera calculando tres tipos de vectores: *Query*, *Key* y *Value*.

- ***Query* (Consulta):** Es un vector que representa la palabra de interés, cuya relación

con otras palabras en la secuencia queremos analizar.

- ***Key* (Clave):** Es un conjunto de vectores que representan todas las palabras de la secuencia. Se utilizan en combinación con la Query para determinar la ponderación de cada palabra de la secuencia respecto a la palabra de interés.
- ***Value* (Valor):** Es un conjunto de vectores con los que se genera una proyección lineal a partir de los vectores de entrada (correspondientes a cada palabra de la secuencia).

Para calcular la atención de una palabra de consulta en relación con las demás palabras, se realizan productos escalares entre el vector *Query* y los vectores *Key*, seguido de la aplicación de una función de activación[3] para obtener los puntajes de atención y luego se utilizan estos puntajes para ponderar los vectores *Value*. Finalmente, se suman los valores ponderados para obtener una representación contextualizada de la palabra de consulta, que tiene en cuenta su relación con las demás palabras en la secuencia. Esto es lo que permite a los *Transformers* capturar las dependencias de largo plazo, sin necesidad de recurrir a mecanismos de memoria como las LSTM.

Sin embargo el tener un único conjunto de pesos de atención limita la cantidad de dependencias que el modelo puede mantener. Por eso lo que se termina utilizando realmente en las arquitecturas de *Transformers* es la técnica de *Multi-Head Attention* que permite al modelo atender a diferentes partes de la secuencia simultáneamente, utilizando múltiples *cabezas de auto-atención*. Estos conjuntos de pesos son independientes entre sí por lo que su cálculo y optimización puede hacerse en forma paralela, lo que resulta muy eficiente. Una vez computados los vectores de cada cabeza se realiza un paso de agregación y normalización para consolidar la información contextual.

2.6.2. Encoder y decoder

El codificador es responsable de procesar la secuencia de entrada y generar una representación vectorial de la misma. Para ello, utiliza las capas de atención y transformación (una capa de tipo *fully-connected*) para capturar la información contextual y las relaciones entre las palabras de la secuencia. El codificador evalúa la importancia de cada palabra en función de su contexto y produce una representación contextualizada de las mismas, que luego utiliza como entrada para el decodificador.

Por otro lado, el decodificador toma la representación numérica generada por el codificador y la utiliza para generar la secuencia de salida deseada, como la traducción en otro idioma. Para lograr esto, el decodificador también utiliza capas de atención y transformación, pero de manera ligeramente diferente. Además de procesar la información contextual, el decodificador debe generar una secuencia de salida que sea coherente y tenga sentido en el contexto del idioma de destino. Por lo tanto, el decodificador realiza un proceso de generación autoregresiva, donde cada palabra en la secuencia de salida se genera una a la vez, teniendo en cuenta las palabras generadas anteriormente.

Un punto interesante es que no siempre se utilizan el encoder y el decoder juntos. Según el tipo de modelo y problema puede usarse solo un encoder, solo el decoder o ambos (arquitectura encoder-decoder).

[3] La función de activación, típicamente una función softmax, transforma los puntajes de atención en una distribución de probabilidad, asegurando que la suma de los pesos sea igual a uno. Esto permite una asignación adecuada de atención a diferentes partes de la secuencia.

Los encoders son usados principalmente en tareas de *natural language understanding* o *information extraction* donde el generar una representación interna del lenguaje resulta fundamental para un buen desempeño de la tarea.

Los decoders (también conocidos como modelos auto-regresivos) son utilizados en problemas de generación automática de texto. Por ejemplo, sistemas conversacionales (*chatbots*) suelen basarse en este tipo de modelo.

Por último, los encoder-decoder son usados para tareas del tipo *sequence-to-sequence*, donde a partir de una secuencia se desea generar otra nueva, no necesariamente de la misma longitud que la original. El caso paradigmático de esto son los modelos de traducción automática (*machine translation*).

En esta tesis nuestro foco está puesto únicamente en los modelos del tipo encoder, ya que son los que permiten generar *embeddings* y resolver el tipo de tareas de interés.

2.6.3. Diferencias de los Transformers con las RNNs y LSTMs

La principal diferencia entre los métodos basados en *Transformers* y aquellos basados en LSTMs (u otras redes recurrentes) es que los primeros no utilizan recurrencia en ningún momento. El *Transformer* procesa todo el *input* en simultáneo, lo cual mejora significativamente las propiedades de paralelización de esta arquitectura con respecto a las RNN, permitiendo entrenar modelos más grandes y de forma más rápida. Otra ventaja de no hacer uso de recurrencia, es que el algoritmo de optimización tiene mejores propiedades de convergencia, haciendo que sea más fácil aprender dependencias de largo plazo, aun cuando se encuentran muy distantes en la secuencia de entrada.

Posiblemente la mayor contra de perder la recurrencia es que se pierde la capacidad de procesar secuencias de longitud arbitraria. Los encoders están limitados a un tamaño máximo de la secuencia de entrada. En casos donde se quiere procesar secuencias de mayor tamaño se debe recurrir a distintas estrategias de particionamiento, lo cual agrega un punto de complejidad. Sin embargo, en la práctica esta limitación no impide que los *Transformers* sean la mejor arquitectura para resolver una larga lista de tareas de NLP.

2.6.4. BERT

BERT (*Bidirectional Encoder Representations from Transformers*) es un modelo de lenguaje basado en la arquitectura de *Transformers* que fue introducido por Devlin et al. [19]. Su principal contribución es el entrenamiento basado en el enfoque de *Masked Language Model* (MLM) y la predicción de la siguiente oración (*Next Sentence Prediction, NSP*), lo que le permite aprender representaciones profundas del lenguaje con una comprensión más rica del contexto.

A diferencia de los modelos autoregresivos como GPT [8], que procesan texto en una única dirección (izquierda a derecha), BERT aprovecha un codificador bidireccional. Esto significa que puede considerar el contexto tanto anterior como posterior a una palabra en una oración, lo que mejora la captura de dependencias a largo plazo.

El entrenamiento de BERT consiste en dos tareas principales:

- **Masked Language Model (MLM):** Durante el pre-entrenamiento, se ocultan (*mask*) aleatoriamente algunas palabras de la entrada y el modelo debe predecirlas basándose en el contexto circundante. Esto permite que BERT aprenda representaciones más generales y transferibles a diversas tareas de NLP.

- **Next Sentence Prediction (NSP):** Se entrena al modelo para predecir si, dadas dos oraciones, la segunda es continuación de la primera o no. Esta tarea ayuda a mejorar el rendimiento en aplicaciones como la respuesta a preguntas y la inferencia de texto.

BERT se ha convertido en la base de muchas variantes optimizadas para distintos dominios y tareas, como ClinicalBERT [4] para textos clínicos y BioBERT [39] para datos biomédicos.

2.6.5. RoBERTa

RoBERTa (*Robustly Optimized BERT Pretraining Approach*) es una optimización del modelo BERT presentada por Liu et al. [41]. Su desarrollo se basa en la hipótesis de que BERT no estaba explotando completamente la capacidad de la arquitectura *Transformer* debido a ciertas decisiones en su pre-entrenamiento.

Las principales mejoras introducidas por RoBERTa incluyen:

- **Más datos y mayor tiempo de entrenamiento:** Se entrena con más datos y durante más tiempo, lo que permite obtener representaciones más robustas.
- **Eliminación de la tarea de NSP:** Se encontró que la predicción de la siguiente oración no contribuía significativamente al rendimiento del modelo, por lo que fue eliminada.
- **Aumento en el tamaño del *batch*:** Se usan lotes de datos más grandes, lo que mejora la estabilidad del entrenamiento.
- **Uso de dinámicas de *masking*:** A diferencia de BERT, donde las palabras enmascaradas son fijas para cada ejemplo de entrenamiento, en RoBERTa se utilizan diferentes máscaras en cada iteración, mejorando la capacidad de generalización del modelo.

RoBERTa ha demostrado superar a BERT en varias tareas de NLP sin modificar su arquitectura subyacente, únicamente optimizando su pre-entrenamiento. Al igual que BERT, ha dado lugar a múltiples versiones especializadas, como BioMed-RoBERTa [28] para textos biomédicos.

2.7. Transfer Learning

Goodfellow, Bengio y Courville definen, en su libro "Deep Learning"[27], el *transfer learning* como "usar lo que fue aprendido en una configuración para mejorar la generalización de otra". Aquí configuración es un concepto que incluye no solo el modelo sino también los datos con los que se cuenta y la tarea específica que se quiere resolver. Por simplicidad, nos referimos a la primer configuración como *fuente* o *base*, y a la segunda como *objetivo*.

En general el concepto de *transfer learning* es muy amplio y las definiciones no son tan formales. Sin embargo, a grandes rasgos podemos dividir las técnicas en dos grandes categorías[4]:

[4] Inspirado en el artículo de Medium https://medium.com/@davidfagb/guide-to-transfer-learning-in-deep-learning-1f685db1fc94

- Extracción de features
- *Fine-tuning*

El caso de extracción de features consiste en usar un modelo pre-entrenado (base) para extraer *features* de los datos y pasárselas luego al modelo objetivo. El punto clave aquí es que el modelo base y el objetivo suelen ser completamente distintos, solo las features extraídas los relacionan. Un ejemplo de esto son justamente los *embeddings* extraídos usando FastText. Para poder generar tales *embeddings* primero se entrena un modelo de clasificación de forma auto-supervisada. Sin embargo, luego el modelo se descarta y solo se utiliza el mapeo entre *tokens* del vocabulario y vectores (embeddings) dados por la capa oculta del modelo. En este caso, los *embeddings* son las features. Lo mismo aplica al procedimiento para obtener embeddings a partir de modelos de lenguaje, explicado en la sección 2.4.2.

Fine-tuning es un término ampliamente utilizado en el contexto de *Deep Learning* que se puede pensar como cualquier actualización de los pesos de un modelo previamente entrenado para que el mismo sea útil en una nueva tarea o dominio objetivo. Notar que en este caso la tarea fuente y objetivo no necesariamente tienen que ser la misma. Un ejemplo de esto son los *transformers* como BERT que son entrenados originalmente como *Masked Language Models* pero luego pueden ser utilizados para otras tareas como *Named Entity Recognition* simplemente cambiando la capa de salida y actualizando los pesos (entrenando) sobre el conjunto de datos específico. A esta forma más específica de *fine-tuning*, en la que tomamos un modelo entrenado para una cierta tarea y lo ajustamos para que sirva en otra, la llamamos *task adaptation*.

Otro caso particular de *fine-tuning* es el de *domain adaptation* (del inglés, adaptación al dominio). Es un tipo de *transfer learning* que se realiza en un escenario donde las tareas fuente y objetivo son las mismas (*transductive transfer learning*) pero la distribución de los datos difiere. Básicamente consiste en tomar el modelo entrenado para la tarea fuente y continuar entrenándolo sobre los datos de la tarea objetivo. La utilidad está en que los pesos ya están inicializados de una forma que puede acortar el entrenamiento necesario sobre la tarea objetivo, siempre y cuando los dominios fuente y objetivo estén relacionados de alguna forma. Por ejemplo, la tarea fuente puede ser predecir palabras en español de dominio general y el dominio objetivo trata de predecir palabras en español pero en un dominio más acotado (como por ejemplo el biomédico).

2.8. Bootstrapping

El Muestreo Bootstrap (o simplemente *Bootstraping*) es una técnica estadística que nos permite estimar la variabilidad de nuestros resultados y la incertidumbre asociada a nuestras métricas de evaluación. Consiste en generar múltiples muestras de datos a partir de una sola muestra existente, mediante muestreo con reemplazo. Al promediar los resultados obtenidos en todas las muestras generadas, obtenemos una estimación más precisa de las métricas de evaluación y una mejor comprensión de la variabilidad del rendimiento del modelo. Nos permite calcular los intervalos de confianza[5] para tener una mayor comprensión de la incertidumbre asociada a nuestros resultados. Esto nos ayuda a

[5] Un intervalo de confianza es un rango de valores que se utiliza para estimar el parámetro de una población. Indica la certeza o nivel de confianza con el que se espera que el verdadero valor del parámetro esté contenido dentro de ese rango.

poder realizar una evaluación más robusta del modelo y a tomar decisiones más informadas sobre su capacidad de generalización.

2.9. Reducción de dimensionalidad

La reducción de dimensionalidad en el aprendizaje automático es una técnica utilizada para reducir el número de variables (*features*) en un conjunto de datos, manteniendo la mayor cantidad de información posible. Esto resulta de suma utilidad para simplificar el entrenamiento de modelos, reduciendo el tiempo de cómputo y evitando la llamada *curse of dimensionality* ("maldición de la dimensionalidad")[6], donde demasiadas características pueden llevar al *overfitting* si no se tienen suficientes ejemplos. Otro uso típico es poder generar visualizaciones de datos que viven en espacios multi-dimensionales que son imposibles de visualizar *a priori* pero que gracias a la reducción de dimensionalidad se pueden proyectar en dos o tres dimensiones.

Dos métodos comunes para la reducción de dimensionalidad son Análisis de Componentes Principales (PCA, por sus siglas en inglés) [1] y *t-Distributed Stochastic Neighbor Embedding* (t-SNE) [63]. A continuación describimos brevemente ambos métodos.

PCA es una técnica lineal que transforma las *features* originales en un nuevo conjunto de variables no correlacionadas llamadas componentes principales. Estos componentes se ordenan según la cantidad de *varianza* que capturan del conjunto de datos original. Los primeros componentes principales suelen capturar la mayor parte de la varianza, lo que permite reducir las dimensiones manteniendo solo estos componentes. PCA se utiliza ampliamente para la compresión de datos y la visualización de datos de alta dimensionalidad, donde la relación de las variables se asemeja a una función lineal.

t-SNE es una técnica no-lineal utilizada principalmente para visualizar datos de alta dimensionalidad. Funciona convirtiendo las similitudes entre puntos en probabilidades y luego minimizando la divergencia entre estas probabilidades en el espacio de alta dimensionalidad y un espacio de menor dimensionalidad. t-SNE es particularmente eficaz en preservar las estructuras locales, lo que lo convierte en una herramienta efectiva para visualizar *clusters* (explicados en la sección siguiente) en conjuntos de datos complejos.

Es importante destacar que tanto PCA como t-SNE son métodos no-supervisados que solo requieren un conjunto de puntos en el espacio para funcionar, no necesitando etiquetas. PCA típicamente tiene mayor variedad de usos, mientras que t-SNE es una técnica prácticamente diseñada solo para visualización de datos de alta dimensionalidad con relaciones complejas.

Si bien podría considerarse que t-SNE es una herramienta más poderosa que PCA en términos de visualización (dado que puede capturar relaciones más complejas, al menos para datos no lineales) resulta común emplearlas en conjunto cuando la cantidad de datos y sus dimensiones son muy elevados: primero aplicar PCA sobre los datos para llevarlos a una dimensionalidad menor (por ejemplo, bajar de 1000 dimensiones a 100) y luego aplicar

[6] El término "curse of dimensionality" fue acuñado por Richard Bellman (1961) para referirse a un fenómeno que ocurre en espacios de alta dimensionalidad, donde a medida que aumenta el número de *features* o dimensiones en un conjunto de datos, los datos se vuelven escasos y dispersos en el espacio de representación. Esto provoca comportamientos que son contraintuitivos respecto del análogo en dos o tres dimensiones. Las distancias entre puntos de datos se vuelven menos significativas, dificultando tareas como la clasificación y el *clustering*. Una explicación mucho más detallada puede encontrarse en el capítulo 2 del libro "The Elements of Statistical Learning"[31] de Hastie, Tibshirani y Friedman.

t-SNE sobre el conjunto reducido. El motivo de esto es que ejecutar el algoritmo de t-SNE es mucho más costoso que PCA, especialmente cuando hay un alto número de ejemplos.

2.10. Clustering

El *clustering* (del inglés, agrupación) en *Machine Learning* es una técnica de aprendizaje no-supervisado utilizada para agrupar puntos de datos similares en *clusters* (cúmulos) donde los puntos dentro del mismo *cluster* son más parecidos entre sí que a los de otros *clusters*. Esto es útil para descubrir patrones, segmentar datos y reducir la complejidad de los conjuntos de datos.

Existen diversas estrategias para realizar *clustering*. En este trabajo consideramos tanto métodos basados en partición, como K-Means, así como enfoques basados en densidad, como DBSCAN y HDBSCAN. A continuación, describimos los métodos más relevantes utilizados.

K-Means es un algoritmo popular de agrupación basado en centroides. Funciona inicializando un número predefinido de *clusters* (K) y asignando puntos al centro de *cluster* más cercano. El algoritmo actualiza iterativamente los centros de los *clusters* (centroides) minimizando la varianza dentro de cada clúster. K-Means es eficiente y funciona bien cuando se conoce de antemano el número de *clusters* deseados y los mismos son aproximadamente esféricos y de tamaño similar.

DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*) es un algoritmo de *clustering* basado en densidad, diseñado para identificar grupos de datos en espacios de alta dimensionalidad sin necesidad de especificar el número de clusters de antemano. A diferencia de K-Means, DBSCAN detecta regiones de alta densidad separadas por regiones de baja densidad, lo que le permite identificar clusters de forma arbitraria y manejar datos con ruido de manera más robusta. El algoritmo se basa en dos parámetros principales: ϵ, que define la distancia máxima entre puntos considerados vecinos, y *minPts*, que establece el número mínimo de puntos requeridos para que una región sea considerada densa. Un punto se clasifica como *core* si tiene al menos *minPts* vecinos dentro de un radio ϵ; si está dentro de la vecindad de un *core*, pero no alcanza el umbral, se considera un punto de borde; y si no cumple ninguna de estas condiciones, se clasifica como ruido. La principal ventaja de DBSCAN es su capacidad de encontrar clusters con formas irregulares y su robustez frente a la presencia de valores atípicos.

HDBSCAN (*Hierarchical Density-Based Spatial Clustering of Applications with Noise*) es una extensión jerárquica de DBSCAN que optimiza la selección de parámetros y permite la detección de clusters con diferentes densidades. A diferencia de DBSCAN, que requiere un valor fijo de ϵ, HDBSCAN elimina esta dependencia construyendo un árbol de jerarquía de densidades y extrayendo los clusters en función de su estabilidad. Esto hace que el algoritmo sea más flexible y adecuado para datos con variaciones en la densidad de los grupos.

3. ANTECEDENTES Y REVISIÓN DE LA LITERATURA

En este capítulo introducimos los trabajos previos que dan origen a los datos utilizados y parte de los problemas que queremos atacar. También presentamos una revisión más general de la literatura mostrando diversos trabajos que en algunos aspectos pueden estar relacionados a lo realizado en esta tesis.

3.1. Antecedentes en idioma inglés

Dentro del contexto de *Biomedical NLP* (BioNLP), la mayor parte del esfuerzo volcado en desarrollar *embeddings* o *language models* específicos ha sido con *corpora* en idioma inglés. Es importante destacar que no existe un único tipo de texto cuando hablamos de BioNLP. Por ejemplo, pueden encontrarse artículos científicos, informes de alta médica (epicrisis), notas tomadas durante la visita del paciente al consultorio, e informes de estudios de imágenes. También libros e incluso noticias periodísticas. Particularmente, lo que nos interesa en este trabajo es lo que englobaremos bajo el término "texto clínico": todo texto que se genera por los profesionales de la salud durante la operatoria cotidiana de las instituciones clínicas y hospitalarias. Esto incluye distintos tipos de textos como por ejemplo: notas de consultas, historiales clínicos, historiales familiares, informes correspondientes a diversos tipos de estudios, epicrisis, entre otros. Uno de los factores que ha dado un gran impulso al área de BioNLP ha sido la digitalización de todos o varios de estos textos, facilitando su consumo para distintos tipos de aplicaciones informáticas. Sin embargo, la mayoría de los *embeddings* y modelos de lenguaje que son entrenados, como puede apreciarse en los relevamientos realizados por Kalyan y Sangeetha [36] y Khattak et al. [38], siguen siendo basados en otros tipos de textos porque la publicación de los informes clínicos digitales no es algo que pueda tomarse a la ligera por cuestiones relacionadas con la privacidad y confidencialidad de la información que allí se muestra.

A pesar de esto, se cuenta con algunos recursos públicos en idioma inglés, siendo una de las más relevantes, por su tamaño y difusión, la base de datos MIMIC-III [34], que fue publicada en el año 2016 y contiene información asociada a alrededor de 40 mil pacientes de unidades de terapia intensiva. Los informes han sido anonimizados y también cuenta con diversos tipos de anotaciones. Es a partir de esta base que se entrenaron, por ejemplo, las variantes más conocidas del modelo llamado ClinicalBERT [4], el cual se observó que captura mejor la terminología médica y la gramática específica de los registros clínicos, comparado con modelos generales (como BERT [19]) o modelos entrenados sobre texto médico no-clínico (como BioBERT [39]). Esto lo mostraron también Wang et al. en su trabajo del 2018 [66]: cuando el objetivo final es analizar texto clínico, entrenar *embeddings* sobre este mismo tipo de texto conduce a mejores propiedades semánticas respecto de cuando se entrena usando otras fuentes posibles como artículos científicos, noticias o artículos de Wikipedia.

Otro punto importante que señalan Khattak et al. en [38] es la relevancia de tener conjuntos de datos específicamente orientados a hacer evaluaciones intrínsecas. Este tipo de evaluación, que se introdujo en la sección 1.3, se realiza con tareas que únicamente están relacionadas a los *embeddings* mismos, como por ejemplo calcular similitud y *relatedness*. Para esto resulta útil tener conjuntos de datos con pares de palabras que sean similares

o estén relacionadas, como forma de realizar una evaluación cuantitativa. Se señalan la existencia de los conjuntos de datos UMNSRS-sim, UMNSRS-rel [50] y MayoSRS [51], que han sido creados específicamente para el dominio médico. Los tres consisten en pares de conceptos extraídos del metatesauro UMLS[1] (Unified Medical Language System). UMLS es internacionalmente reconocido y da un marco unificado que integra múltiples terminologías médicas, lo que la hace muy útil en el área de BioNLP. Los pares de conceptos relacionados fueron anotados manualmente por expertos del dominio clínico. Estos conjuntos de evaluación tienen un enfoque generalista en cuanto a los términos que incluyen, por lo que pueden no ser lo más adecuado a utilizar cuando se trabaja con un sub-dominio en particular, como por ejemplo informes de imágenes radiológicas. Adicionalmente estos *benchmarks* son previos incluso al desarrollo de Word2Vec, lo cual significa que no fueron pensados específicamente para el trabajo con este tipo de *embeddings*. Mucho menos se consideró el caso de los *embeddings* contextuales, para los cuales una comparación de a pares puede no ser el mejor enfoque. Más recientemente se publicó un trabajo [37] escrito por Khan et al. que afrontó el desafío de construir un conjunto específicamente desarrollado para el dominio de reportes radiológicos y que considera tanto *embeddings* estáticos como contextuales. Tal trabajo sin embargo, es pensando nuevamente en el idioma inglés y además no hemos encontrado ningún código publicado abiertamente que pueda utilizarse como punto de partida para hacer algo similar en español.

3.2. Corpus y SpRadIE

En la tesis doctoral de Cotik [18], se presenta un *corpus* proveniente de un importante hospital de Argentina. Este *corpus* consta de aproximadamente 80 mil informes de ecografías escritos en español. Se le realizaron una serie de pre-procesamientos, tanto a nivel de formato como de anonimización. Dichos informes no poseen anotaciones adicionales. Este *corpus* es importante para nuestro trabajo ya que es el que utilizamos para desarrollar nuestros *embeddings*. Profundizamos sobre el mismo en la sección 4.1.

En el mismo trabajo y en [15] se trabaja en la anotación de un subconjunto de esos informes, cuidadosamente seleccionados bajo ciertos criterios para asegurar la representatividad deseada. En total hay 10 tipos de entidades diferentes que fueron anotadas entre ellas entidades anatómicas, hallazgos clínicos, ubicaciones, medidas, y otras que serán detalladas en la sección 4.1. Este subconjunto es el que se utiliza como *corpus* de la competencia SpRadIE, detallada a continuación.

SpRadIE fue un *challenge*[2] de reconocimiento de entidades nombradas aplicada a informes de ecografías en español. Se presentó en el contexto del *eHealth Evaluation Lab* de la *Conference and Labs of the Evaluation Forum 2021* (CLEF 2021), en la categoría de *Multilingual Information Extraction*.

En este marco, participaron un total de siete equipos, que trataron de resolver el problema propuesto con distintos enfoques: EdIE [61], SINAI [42], SWAP [53], CTB [60], LSI [22], HULAT [25] e IMS [20]. Debido a que SpRadIE y el *corpus* utilizado para su desarrollo son centrales en el presente trabajo, resumimos algunos aspectos interesantes de los esfuerzos realizados por los dos equipos que alcanzaron los mejores resultados del

[1] `https://www.nlm.nih.gov/research/umls/index.html`

[2] La palabra "challenge" (del inglés, desafío) es un término que suele usarse en *Machine Learning* para referirse a tareas que son presentadas en formato de competencia, con reglas definidas y un mecanismo particular de evaluación.

challenge. La métrica que se utilizó para determinar al ganador fue *micro-averaged F1-Score* con *match* parcial (una explicación de esta métrica puede encontrarse en la sección 4.5).

El equipo EdIE-KnowLab [62] fue el que mejores resultados obtuvo, aunque también puede considerarse la solución más compleja en términos de la cantidad de modelos usados internamente. La propuesta se compone de 2 subsistemas (en la versión que resultó ganadora): uno es un *ensemble* de BETOs [11] finetuneados (uno para cada tipo de entidad), y el otro es un diccionario que mappea *tokens* a entidades basado en un análisis estadístico sobre el *corpus* de entrenamiento (aprovecha repetición de patrones simples). Para el preprocesamiento hacen correcciones de ortografía y gramática. También expanden el *span* de algunas etiquetas mal anotadas (por ejemplo, a veces la última letra de una palabra queda por fuera de la anotación por error). Como puntos de interés, destacamos el enfoque de tener un BETO por cada tipo de entidad como forma de manejar el problema de solapamiento entre entidades (una misma palabra puede pertenecer a más de una entidad), así como también disminuye los efectos negativos del desbalanceo entre entidades (ciertas entidades son mucho más comunes que otras como puede verse en la figura 4.1 y esto puede sesgar al modelo en su aprendizaje). También remarcamos que el mejor resultado de la competencia haya sido logrado a través del uso de *Transformers*, que son el estado del arte actual en un gran número de tareas de NER. Algunas oportunidades de mejora que identificamos a partir de lo expuesto en esta solución son: mejorar la performance sobre las categorías Degree y Conditional Temporal (ver sección 4.1.2 para mayor detalle sobre todas las categorías), así como también mantener la buena performance pero bajando la complejidad del proceso y la solución.

En segundo lugar quedó el equipo LSI_UNED [23]. Su arquitectura consiste en dos BiLSTMs + CRF: una específicamente para detectar negaciones, y otra para el resto de las entidades. Argumentan el uso de BiLSTM en lugar de Transformers, por el tamaño reducido del *corpus* de la competencia. A su vez, separan a las anotaciones que no son negaciones en dos conjuntos: aquellas en las que los solapamientos de entidades son más comunes (Location, Findings y Abbreviations) y el resto. De esta forma, las anotaciones distintas de Negations se intentan predecir con una arquitectura basada en 4 BiLSTM + CRFs: 3 están dedicados a cada una de las entidades que suelen solaparse, y la ultima para el resto. Para la capa de entrada usan *embeddings* pre-entrenados de FastText (uno generalista y uno de dominio médico pero sin casos de texto clínico). Estos *embeddings* se combinan con otras *características* generadas en forma manual y que codifican condiciones booleanas como por ejemplo: "empieza en mayúscula", "termina en punto", "término numérico", "término mayoritariamente numérico", entre otros. De este aporte destacamos que a diferencia del equipo anterior aquí se usa una arquitectura basada en BiLSTM y uso de *embeddings* estáticos y aún así se obtienen resultados muy cercanos al mejor. También que nuevamente surge la idea de hacer una segmentación como forma de disminuir el impacto de las entidades solapadas. Como oportunidades, nuevamente reconocemos la posibilidad de intentar disminuir la cantidad de componentes (en este caso, además de varios modelos hay un trabajo manual de definir *features* para el modelo). Además algo interesante que mencionan es que no ven una diferencia sensible entre el desempeño de los *embeddings* de dominio general y los de dominio médico. Es un interesante punto de partida para nosotros que queremos abordar esto en mayor profundidad y entender si se repite con *embeddings* más específicos o de distintas características.

3.3. Otros corpora en español

Citando a Neves y Leser [49]: "la falta de *gold standards* es uno de los principales cuellos de botella para desarrollar nuevos métodos de *text mining*". *Gold standard* es un nombre que se suele dar a aquellos *corpus* que han sido anotados generalmente por un humano (o al menos corregido por un humano). Estos son recursos fundamentales ya que no solo sirven para entrenar modelos, sino también para poder tener un *benchmark* contra el cual evaluar y realizar comparaciones objetivas entre diversos métodos .

En los últimos años han surgido varios esfuerzos para tratar de ampliar la disponibilidad de recursos en idioma español, particularmente en el área de BioNLP (de los cuáles, SpRadIE es uno más). Realizamos una revisión de varios de estos trabajos con la expectativa de poder encontrar *corpora* afines a nuestro problema de modo de poder usarlos para enriquecer (aumentar) nuestros datos de entrenamiento o en segunda instancia para poder tener un *benchmark* adicional al momento de evaluar la utilidad de nuestros *embeddings*. No encontramos ningún conjunto que nos convenciera para tales usos. No obstante, consideramos que es relevante mencionarlos y lo hacemos a continuación.

En la competencia *CodiEsp* [46], se disponibilizó un *corpus* con 1000 informes clínicos escritos en español, cubriendo una diversidad de especialidades médicas. Todos estos informes fueron cuidadosamente anotados por especialistas utilizando códigos CIE-10[3]. Si bien es un recurso muy interesante, en nuestro caso la diferencia de dominios y estilos de escritura nos hicieron descartarlo para su utilización: estos informes suelen ser más largos y estar mejor redactados que los de las ecografías.

Algo similar nos sucede con el *corpus* utilizado en el *challenge* de CANTEMIST [45]: una colección de 1301 historias clínicas de pacientes oncológicos. En este caso las anotaciones corresponden a morfología de neoplasia (tumores), siguiendo la Clasificación Internacional de Enfermedades para Oncología en formato electrónico[4] (eCIE-O). Aquí si bien aparecen algunos estudios de imágenes que podrían ser afines a nuestro caso, dado que cada informe es un historial clínico completo, habría que extraer los fragmentos de interés (únicamente lo referido a las imágenes). Algo interesante para remarcar es que tanto este *corpus* como el anterior tienen un origen en común: el *Spanish Clinical Case Corpus* (SPACCC)[5] que a su vez es un subconjunto de informes tomados de SciELO[6] una biblioteca de uso público con reportes de casos de diversos países. Otra competencia que se basó en los mismos datos fue PharmaCoNER [26], que se enfoca en reconocer entidades asociadas a distintos fármacos y sustancias. En nuestro caso optamos por no hacer uso de la herramienta SciELO porque el esfuerzo de revisar y filtrar los informes afines es grande y aún así puede haber diferencias de formato significativas.

A pesar de que no utilizamos directamente ninguno de estos *corpora*, todos estos trabajos muestran la aplicabilidad de modelos de lenguaje como *BERT* y *Flair* entrenados específicamente en el dominio médico en español.

Un *corpus* en español que sí tiene afinidad con nuestros informes ecográficos es PadChest [9], que se conforma con informes radiográficos de la zona pectoral y cubre patologías asociadas a pulmones y corazón (al igual que parte de nuestros informes). Lamentablemente en este caso surge otro problema: los informes publicados se encuentran con varios

[3] CIE-10 es una clasificación y codificación internacionalmente reconocida para identificar enfermedades, trastornos, signos y síntomas, entre otros hallazgos y circunstancias.

[4] `https://iris.who.int/bitstream/handle/10665/96612/9789241548496-spa.pdf`

[5] `https://github.com/PlanTL-SANIDAD/SPACCC`

[6] `https://scielo.org/es/`

preprocesamientos que generan una pérdida de información. Entre estos preprocesamientos se encuentra la remoción de *stopwords*[7] y la aplicación de *stemming*[8]. Un ejemplo de informe tomado del *corpus* es: "nodul proyect lsd 1 4 cm contorn parcial bien defin cit tac torac . patron intersticial con probabl are panalizacion bibasal probabl fibrosis pulmon .". Debido a que nuestra intención es entrenar justamente *embeddings* que puedan aplicarse sin necesidad de hacer este tipo de preprocesamientos previos, no nos sirve.

Otro trabajo destacado fue realizado en Chile por Báez et al. [5]. En él introducen el *corpus The Chilean Waiting List Corpus* (o ChileanWL), que es una nueva fuente de datos para el reconocimiento de entidades nombradas clínicas en lenguaje español. Este *corpus* está compuesto por historias clínicas electrónicas anonimizadas de hospitales públicos chilenos. Son de tipo *triage* ya que corresponden a una primera evaluación médica antes de derivar al paciente con un especialista. Este *corpus* si bien no lo usamos directamente para entrenar o evaluar, sí lo utilizamos indirectamente: algunos de los modelos de lenguaje a los que les aplicamos *fine-tuning* fueron a su vez previamente fine-tuneados con este *corpus*.

3.4. Otros trabajos relacionados en español

Hasta donde tenemos registro, Soares et al. en su trabajo *Medical Word Embeddings for Spanish: Development and Evaluation* [59] desarrollan uno de los primeros *embeddings* específicamente para el dominio médico en lenguaje español y lo comparan contra *embeddings* de dominio general, en forma tanto extrínseca como intrínseca. Para desarrollar dichos *embeddings* usan dos fuentes de datos: la base de datos SciELO y un subconjunto de categorías de Wikipedia (Farmacología, Farmacia, Medicina y Biología). Por lo tanto, no se trata de un *corpus* orientado específicamente a texto clínico. También cabe destacar que el trabajo se enfoca en *embeddings* estáticos (FastText) y no se incluyen pruebas con *embeddings* contextuales. Una de las contribuciones más remarcables de este trabajo es que, para poder realizar la evaluación intrínseca, realizan la adaptación al español de los conjuntos de datos UMNSRS-sim, UMNSRS-rel y MayoSRS (todos mencionados en la sub-sección 3.1). Lamentablemente son recursos que decidimos no utilizar debido a que se enfocan en terminologías poco relacionadas con nuestros informes de ecografías, por lo que *a priori* no consideramos que nos permita realizar evaluaciones útiles.

Tal vez el trabajo más similar a lo que tratamos de hacer aquí es el realizado por Akhtyamova et al. en [3]. En dicho trabajo los autores exploran también la utilización de *embeddings* contextuales para mejorar la *performance* en un problema de NER en español del dominio biomédico. En su caso trabajan sobre el *corpus* de la *task* de Pharmaco-NER [26]. Aquí se hace una comparativa también entre BiLSTM (específicamente Flair) y Transformers, e incluso se evalúa también FastText. Sin embargo, hay algunas diferencias: primero el dominio que utilizan para entrenar es mucho más amplio que el nuestro (que nos limitamos apenas a informes de ecografías). Segundo, el enfoque principal del citado artículo termina estando más puesto en mejorar la tarea de NER en sí que en hacer una evaluación detallada del aporte de los *embeddings*. Por último, nosotros realizamos

[7] En NLP, se llama *stopwords* a palabras que se caracterizan por aportar información relevante, generalmente por tener una frecuencia de ocurrencia muy alta. Un caso típico son preposiciones.

[8] En NLP, se conoce como *stemming* a una técnica que consiste en eliminar prefijos y sufijos de las palabras, quedándose únicamente con su raíz. Esto generalmente se realiza como una forma de agrupar palabras con un significado similar. Por ejemplo, "programación", "programador" y "programas" pueden reducirse a la raíz "programa".

más pruebas con modelos basados en Transformers, mientras que en el otro trabajo se descartan relativamente rápido y los resultados expuestos se centran en Flair.

4. METODOLOGÍA

Este capítulo describe los conjuntos de datos utilizados para realizar el trabajo, su preprocesamiento, los modelos implementados y la metodología de evaluación. La organización que seguimos para esta sección se basa en la que proponen Khattak et al. en su trabajo "A survey of word embeddings for clinical text" [38] y que ilustramos en la Figura 1.1 del capítulo 1. La misma distingue tres grandes etapas: la preparación de los datos (tanto para entrenar los *embeddings* como la tarea objetivo), el entrenamiento de modelos de *embedding* y la evaluación de los mismos (tanto extrínseca como intrínsecamente).

4.1. Corpora utilizados

A continuación, se describen los dos *corpora*, principales que tomamos como punto de partida para desarrollar los modelos y experimentos de este trabajo.

4.1.1. Datos de entrenamiento para los *embeddings*

Este conjunto de datos consta de 82246 informes de ecografías de distintas partes del cuerpo humano, provenientes de uno de los hospitales más importantes de Argentina. Estos informes hacen referencia a, entre otros, diversos órganos y hallazgos clínicos. Por ejemplo, algunos órganos mencionados son: hígado, riñones, útero, pulmones, aorta, articulaciones, etc. Mientras que cuando mencionamos hallazgos clínicos hacemos referencia a expresiones del tipo: "tamaño aumentado", "diferenciación corticomedular", "dilatada", "heterogéneo", entre otras. Los informes en general son relativamente cortos, con un promedio de 8 oraciones por informe y 9 palabras por oración. Un mayor detalle sobre la composición de este *corpus* puede encontrarse en la sección 3.2 de [18]. Además en los ejemplos 1, 2, 3 y 4, presentados a continuación, se muestran algunos informes reales que pueden hallarse en el *corpus*.

Ejemplo 1 Informe correspondiente a una ecografía de glándula tiroidea.

Ambos lobulos tiroideos e itsmo con ecoestructura homogenea. DINESIONES: Lobulo derecho: longitudinal: 3.8 cm., transverso: 1 cm., anteroposterior: 1 cm.; volumen: 2.3 cc. Lobulo izquierdo: longitudinal: 3 cm., transverso: 1.2 cm., anteroposterior: 1 cm.; volumen: 1.9 cc. Istmo: 0.2 cm.

Ejemplo 2 Informe correspondiente a una ecografía abdominal completa con características normales.

HIGADO: tamano y ecoestructura normal. Arteria hepatica, venas porta y suprahepaticas sin alteraciones. VIA BILIAR intra y extrahepatica: no dilatada. VESICULA BILIAR: alitiasica. Paredes y contenido normal. PANCREAS: tamano y ecoestructura normal. BAZO: tamano y ecoestructura normal. Diametro longitudinal:8 (cm) RETROPERITONEO VASCULAR: sin alteraciones. No se detectaron adenomegalias. No se observo liquido libre en cavidad. Ambos rinones de caracteristicas normales.

Ejemplo 3 Informe correspondiente a una ecografía de cadera, en la que se evalúa tanto de forma estática como dinámica la morfología y estabilidad de las caderas.

ESTUDIO ESTATICO: Ambas caderas centradas. Cobertura osteocartilaginosa adecuada. Leve displasia acetabular derecha. Nucleo de osificacion presente en la cadera izquierda. No se observo osificacion del nucleo femoral derecho. ESTUDIO DINAMICO: Ambas caderas estables. El examen solicitado no es un metodo de Screening . Esta indicado para estudiar POBLACIONES DE RIESGO o para corroborar hallazgos clinicos. EL RESULTADO NORMAL DEL EXAMEN, NO EXIME DEL SEGUIMIENTO CLINICO.

Ejemplo 4 Informe correspondiente a una ecografía renal.

Ecografia renal pre y postmiccional; RD de menor tamaño que el izquierdo sin dilataciones y con buena diferenciación. Disminucion de corteza en polos:pieonefritis secuelar?? No se visualizan ureteres vejiga sin alteraciones RI 11 cm, RD 8.2 cm

Analicemos brevemente algunas características que pueden apreciarse en estos estos ejemplos. Una primera observación es que los cuatro informes tratan sobre diferentes partes del cuerpo y el formato de escritura es diferente entre cada uno. Algo que sí comparten en general, es la presentación de medidas de longitud y volumen para describir diferentes partes anatómicas. También comparten el uso de oraciones cortas y sintéticas. En general no se observa el uso de tildes salvo excepciones (como "diferenciación" en el ejemplo 4). En el ejemplo 4 se aprecia el uso de abreviaturas como "RD" y "RI" para referirse a los riñones izquierdo y derecho respectivamente. También en este informe es interesante la utilización de signos de interrogación ("??") como forma de expresar incertidumbre sobre un hallazgo. En el ejemplo 3 se incluye una aclaración genérica sobre cómo deben interpretarse los resultados y un descargo de responsabilidad.

Este *corpus* no posee anotaciones de entidades por lo que se utiliza únicamente para entrenar los *embeddings* y realizar el *fine-tunning* de los modelos de lenguaje.

Para facilitar la referencia a estos datos, en lo que resta del informe llamamos a este *corpus* `ultrasounds-raw-80k`.

4.1.2. Datos para la tarea objetivo

Este *corpus* proviene de la competencia (o *challenge*) SpRadIE [16] descripta en la sección 3.2. Consiste en un subconjunto de 474 informes tomados de `ultrasounds-raw-80k` que fueron etiquetados manualmente por expertos. Además, para su uso en la competencia, se realizó anonimizado, eliminado de signos diacríticos, separación de oraciones con saltos de línea y remoción de informes con menos de tres palabras. Más detalles sobre los procesamientos realizados pueden encontrarse en la sección 3 de "Extracción de información en informes radiológicos escritos en español" [18]. En dicho trabajo también se explica cuál fue el criterio de selección del subconjunto de informes a anotar. En *Annotation of Entities and Relations in Spanish Radiology Reports* [15] se explica el criterio de anotación (aunque en la competencia [16] se aclara que hubo una segunda fase para mejorar dichas anotaciones). El proceso de anotación fue cuidadoso, siguiendo un procedimiento iterativo de anotación y revisión. Las anotaciones fueron realizadas por dos anotadores (entre ellos, un estudiante avanzado de medicina) durante tres iteraciones, hasta que se logró una convergencia en el esquema y en el criterio usado por ambos. Para cuantificar la consistencia

entre las anotaciones en cada ronda, se calculó el *inter-annotator agreement*[1], utilizando el coeficiente Kappa de Cohen [13].

Los informes se encuentran separados en cuatro particiones como se detalla en la tabla 4.1. Este *corpus* lo usamos para la evaluación extrínseca de los *embeddings* y modelos de lenguaje. Para facilitar su referencia posterior, lo llamamos `spradie-corpus` para el resto del informe.

Conjunto	**Descripción**	**Cantidad de informes**
train	Primer conjunto liberado durante la competencia, que incluye tanto informes como anotaciones, para los análisis y entrenamientos preliminares	175
devSameSample	Conjunto de desarrollo que fue liberado con sus anotaciones a mitad de la competencia para poder validar los modelos entrenados y usarse como data adicional. Este conjunto intenta mantener una distribución similar a la de entrenamiento y no incluye palabras por fuera de las de ese conjunto.	47
devHeldOut	Liberado al mismo tiempo que devSameSample, este conjunto se caracteriza por incluir palabras que no se encuentran en el conjunto de entrenamiento y permite analizar la capacidad de generalización de los modelos. Palabras que solo ocurren en este conjunto: hipertensión, epiplón, portal, cardíaco, aorta, corona.	45
test	Conjunto sobre el cual se deben generar las anotaciones que serán evaluadas para decidir el resultado de la competencia. Las anotaciones no fueron liberadas ya que son de uso interno para medir resultados. Incluye palabras de la distribución de entrenamiento pero también palabras nuevas como: ovario, útero, endometrio, uretra, suprahepático, ganglio, tiroides, entre otras detalladas en [17].	207

Tab. 4.1: Forma en la que se encuentran distribuidos los 474 informes que componen el *corpus* de SpRadIE.

Las diez entidades que fueron anotadas en los informes y cuya identificación es el objetivo de la tarea de SpRadIE, son las siguientes:

[1] El *inter-annotator agreement* (IAA) es una métrica que evalúa el grado de concordancia entre múltiples anotadores al etiquetar un conjunto de datos. Se utiliza comúnmente en tareas de procesamiento de lenguaje natural y anotación manual de datos para medir la consistencia y fiabilidad de las etiquetas asignadas. Existen diferentes coeficientes para calcularlo, como el coeficiente de Kappa de Cohen, el coeficiente de Kappa de Fleiss o el Alpha de Krippendorff, dependiendo del número de anotadores y del tipo de datos. Un alto valor de IAA indica que la tarea de anotación tiene una alta reproducibilidad.

- Anatomical_Entity (AE): partes del cuerpo. Por ejemplo: "pecho", "hígado", "lóbulo tiroideo derecho".

- Finding (FI): hallazgos clínicos. Por ejemplo: "quistes", "adenomaglias".

- Location (LO): ubicación en el cuerpo, indicando una región del mismo o bien una posición relativa a una AE. Por ejemplo: "región biliar", "paredes", "cavidad".

- Measure (ME): expresión de medida. Por ejemplo: "0.3 mm", "0.5 cc", "2 cm.", "0.8 (cm.)".

- Type_of_Measure (TM): palabras que indican el tipo de medida a la que una entidad de tipo ME hace referencia. Por ejemplo: en "diámetro longitudinal 3 (cm)", "diámetro longitudinal" fue anotada como TM, mientras que "3 (cm)" como ME.

- Degree: palabras que indican el grado de un hallazgo (FI). Por ejemplo: "leve", "ligera" ("ligera esplenomaglia").

- Abbreviations (AB): abreviaturas y acrónimos. Por ejemplo: "RI" (por "riñón izquierdo"), "cm" (por "centímetros").

- Negation (NT): negación. Por ejemplo: en "no se detectaron adenomaglias", "no" se anotó como NT.

- Uncertainty (UT): indicador de que existe una probabilidad (no certeza) de que cierto hallazgo pueda estar presente en un paciente. Por ejemplo: en "compatible con hipertrofia pilórica", "compatible con" fue anotada como UT.

- Conditional_Temporal (CT): indicador de que algo sucedió en el pasado o podría suceder en el futuro. Por ejemplo: en "antecedentes de atresia", "antecedentes" se etiquetó como CT.

Estás últimas tres entidades (NT, UT y TC) son también llamadas *hedge cues* (lo que podría traducirse como "señales de incertidumbre").

En la figura 4.1 se presenta un gráfico que muestra el porcentaje que representa cada tipo de entidad sobre el total de anotaciones del *corpus*. Como puede observarse, tal distribución está lejos de ser balanceada, siendo que solo la combinación de Anatomical Entity, Abbreviation y Finding representa el 60 % de las anotaciones.

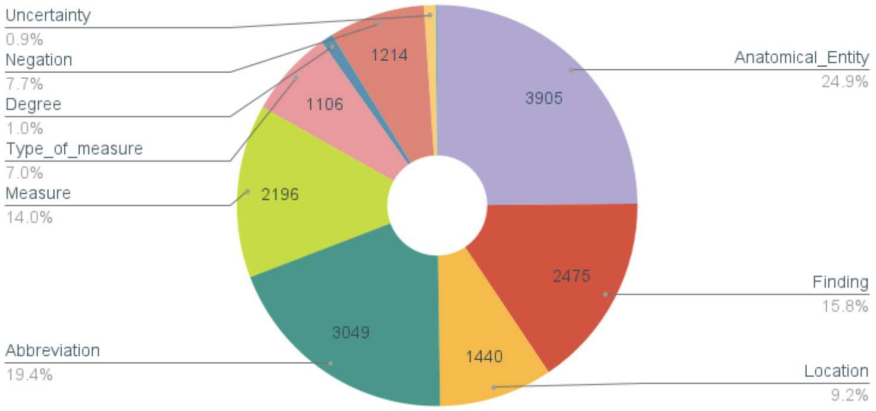

Fig. 4.1: Distribución de las entidades en el *corpus* `spradie-corpus` según el tipo de entidad anotada. Gráfico tomado del *overview* realizado de la competencia [17].

4.2. Pre-procesamiento de los datos

Cuando hablamos del pre-procesamiento (o simplemente procesamiento) de los datos, es importante hacer una distinción de dos etapas. Hay un primer procesamiento, que es común para todos los modelos que se entrenarán. En esta etapa se incluyen, por ejemplo, la anonimización y la eliminación de informes demasiado cortos. Luego, hay un segundo momento, que es específico de cada método. Por ejemplo, la *tokenización*. En esta sección hablaremos únicamente del primer tipo que es común a todos los métodos. El segundo tipo se explica en la sección 4.3 donde desarrollamos las distintas técnicas de *embeddings* y modelos de reconocimiento de entidades nombradas.

Como resultado del pre-procesamiento obtenemos nuevos *corpora*, a los cuales nuevamente asignaremos nombres para su posterior referencia:

- `anonymized-ultrasounds-80k`: Obtenido a partir de `ultrasounds-raw-80k` después de pasar por un proceso de anonimización. Mantenemos mayúsculas/minúsculas y también signos de puntuación y acentos.

- `spradie-corpus-bio-format`: Obtenido a partir de `spradie-corpus` pero convertido a un formato que resulta más conveniente para trabajar con modelos (formato BIO).

Las particularidades de cada uno y el proceso por el cual se obtienen se explica en las siguientes subsecciones.

4.2.1. Obtención de anonymized-ultrasounds-80k

A continuación se detallan los pasos para el procesamiento del *corpus* ultrasounds-raw-80k y que conducen a la obtención de los *corpora* anonymized-ultrasounds-80k-with-diacritics y anonymized-ultrasounds-80k-without-diacritics.

1. Los informes vienen en un archivo con formato PSV (del inglés *Pipe Separated Values*) donde, además del texto principal escrito por los médicos, hay otros metadatos como: un identificador, la edad del paciente y la fecha en la que se realizó el estudio.

Estos datos los removemos, quedándonos únicamente con los informes, ya que no consideramos que aporten información relevante para los *embeddings* por ser datos poco o nada generalizables. Además evitamos el riesgo que los *embeddings* codifiquen información que podría permitir la identificación del paciente (por ejemplo, a partir de la fecha, la edad y el diagnóstico).

2. El texto de cada informe comienza, a su vez, con un identificador alfanumérico generado previamente en el trabajo realizado por Cotik [18], que permite tener una forma de identificarlos. Este identificador no es de interés para los *embeddings*, por lo que decidimos eliminarlo.

3. Es común ver en los informes que se utilizan varios espacios o tabulaciones como forma de separación. En el caso de la generación de *embeddings*, no lo consideramos algo que pueda afectar al desempeño de los modelos ya que los métodos que utilizaremos eliminan los espacios excedentes durante la etapa de *tokenización*. Por este motivo, optamos por simplificarlo y reemplazar todos estos espacios y tabulaciones por un único espacio simple de separación.

4. División en conjuntos (también llamadas particiones) de entrenamiento (o *training*) y evaluación (o *test*). Es una práctica común en *Machine Learning* el realizar una separación como esta, donde el conjunto de entrenamiento se utiliza durante el proceso de desarrollo y selección del modelo, y el conjunto de *test* se utiliza solamente al final del proceso como una forma de medir el desempeño del modelo y evaluar cuán bien generaliza por fuera de los datos de entrenamiento (bajo la hipótesis de que el modelo no vio los datos de *test* durante su entrenamiento). Particularmente, la distribución que hacemos de los informes es: 80 % para la partición de *training* y 20 % para la de *test*. Antes de realizar esta separación, decidimos hacer un reordenamiento aleatorio de los informes.

5. Anonimización: este punto es el más complejo del pre-procesamiento, por lo que lo explicamos más en profundidad en la siguiente sub-sección.

6. Remoción de los distintos tipos de signos diacríticos[2]: a modo experimental, una de las cosas que intentamos validar es si la *performance* de los *embeddings* es mejor o no cuando quitamos todos los diacríticos (´,‘, ˜, ¨) de los informes. Por dicha razón, en este punto creamos dos conjuntos de datos distintos: *anonymized-ultrasounds-80k-with-diacritics* (que no realiza este último paso y por lo tanto mantiene todos los diacríticos) y *anonymized-ultrasounds-80k-without-diacritics* (que los remueve).

Anonimización

El objetivo de este paso es el de eliminar lo que se conoce como datos con Información Personal e Identificatoria (PII por sus siglas del inglés, *Personally Identifiable Information*). Si bien no es nuestro principal objetivo, en un primer momento, compartir públicamentye los informes que componen este *corpus*, es de particular importancia destacar que no encontramos resultados que permitan descartar la posibilidad de que el entrenamiento

[2] Un signo diacrítico (o simplemente diacrítico) es un símbolo gráfico que se añade a una letra o a un carácter para modificar su pronunciación, su valor fonético, su acentuación o su significado dentro de un sistema de escritura. Su función principal es diferenciar palabras o sonidos que, de otro modo, podrían ser confundidos en la escritura o en la lectura.

de *embeddings* usando datos que contengan PII pueda filtrar esta información y luego ser recuperada a partir de los *embeddings* o *language models*. Esto podría representar un riesgo para la privacidad de los involucrados.

Como se explica en la sección 3.2.3 del trabajo [18], el hospital del cual fueron tomados los informes atiende alrededor de 1500 pacientes por día en una ciudad con casi 3 millones de habitantes. Debido a estas dimensiones es muy poco probable que algún paciente sea identificado a partir de un cuadro clínico poco frecuente.

Por otra parte, algunos de los informes de `ultrasounds-raw-80k` incluyen nombres de médicos o pacientes, así como también números de matrículas nacionales y provinciales. En general, se siguen algunos patrones que simplifican un poco la búsqueda de estos datos: generalmente los datos del médico se encuentran al final de todo, el nombre del médico suele estar precedido por títulos como "Dr" y "Dra", las matrículas son precedidas por "MN" (nacionales) o "MP" (provinciales). También a veces una ecografía es vista con más de un especialista, lo cual suele marcarse en el informe con una fórmula "Visto con" seguido del nombre del médico de la interconsulta. Sin embargo, al tratarse de texto libre aparecen irregularidades que complejizan la tarea: se hace un uso irregular de los puntos en las abreviaturas (por ejemplo, a veces se escribe "Dr" y otras "Dr.") así también como de las mayúsculas ("Dra", "DRA", "DRa"), y en ocasiones hay errores de tipeo (como "Drs" en lugar de "Dra"). Además a veces el nombre del médico no aparece precedido de ningun patrón o no se situa al final del informe. Del mismo modo los nombres de pacientes (si bien son mucho menos frecuentes) suelen aparecer en el medio del reporte. Por otro lado, existen órganos o enfermedades que tienen nombres propios (por ejemplo, Síndrome de Turner o de Williams) y es importante que no los eliminemos por error debido a que son entidades importantes del dominio al cual queremos ajustar nuestros *embeddings*. Todos estos son solo algunos ejemplos para ilustrar las irregularidades que se dan y que dificultan la tarea de poder identificar todos los datos PII y eliminarlos adecuadamente.

El objetivo de la anonimización fue asegurar la identificación, y posterior eliminación, de todos los datos PII de nuestro *corpus*, minimizando la pérdida de información valiosa por errores en la identificación (falsos positivos). Se evaluaron distintos métodos, que se comentan a continuación, detallando cuales fueron los problemas y ventajas que se encontraron en cada uno.

- Primero se intentó utilizar dos herramientas que son populares para anonimizar textos escritos en inglés: Presidio[3] y Spacy[4]. En ambos casos, el problema que se encontró fue que no estaban lo suficientemente maduras para el lenguaje español. En el caso de Presidio, es una herramienta *open-source* desarrollada por Microsoft específicamente para el problema de anonimización, pero la misma documentación aclara que solo da soporte oficial al idioma inglés por ahora, aunque se dan algunas herramientas para poder adaptar a otros idiomas. Por su parte, Spacy es una biblioteca de uso frecuente en múltiples tareas de NLP y en su caso sí tiene soporte para el idioma español. Sin embargo, al momento de realizar este trabajo, el repertorio con el que cuenta de modelos para NER entrenados con datos en español es muy acotado (solo tres modelos que únicamente varían en tamaño pero no en datos) y de dominio general (usa un *corpus* de noticias). Se hicieron pruebas con este modelo, pero los resultados fueron malos, tanto en términos de falsos positivos como de falsos negativos.

[3] https://github.com/microsoft/presidio
[4] https://spacy.io/

- Modelo de NER basado en RoBERTa obtenido de HuggingFace: `PlanTL-GOB-ES/roberta-base-bne-capitel-ner`[5]. Este modelo utiliza una arquitectura de Transformers, específicamente la de RoBERTa [41], para resolver la tarea de reconocimiento de entidades nombradas. Particularmente, una de las entidades que está entrenado para reconocer son nombres propios, lo cual es útil para nuestro caso de uso. Además, está fine-tuneado sobre un conjunto de textos en idioma español, extraídos de la Biblioteca Nacional de España [29]. En este caso el desempeño es superior a lo visto con los métodos previos cuando hablamos de identificación de nombres de personas, tanto en falsos positivos como falsos negativos . El principal problema que tiene este método, es que en general ignora completamente los números de matrícula. Además, en algunos casos no reconoce el nombre completo y solo hace un *match* parcial (por ejemplo, en "Dr LUdman" solo reconoce "Dr LU" como persona).

- Expresiones regulares[6] (en adelante RegEx): probablemente el método más simple de los mencionados hasta ahora. Permite explotar rápidamente el conocimiento que tenemos sobre la estructura general de los informes. El problema justamente lo tiene cuando tenemos que considerar también casos que se salen un poco de la estructura "común", como los mencionados anteriormente. El caso más complejo de todos es cuando no hay ningún título que preceda al nombre en cuestión, y en tal caso la única posibilidad es escribir un patrón que coincida directamente con el nombre. Otra dificultad que tiene, es que si bien es fácil escribir patrones que hagan *match* con términos como "dr", "visto con", "MN", etc., resulta muchísimo menos obvio como escribir patrones que coincidan con el texto completo que se quiere eliminar. Esto nuevamente se debe a las irregularidades en cuanto a la escritura. No hay garantía de que vaya a haber un punto después del nombre, o que no haya uno antes, ni tampoco hay una cantidad de palabras definida que puede tener un nombre.

- Corrección manual: por último, siempre es una posibilidad repasar manualmente los reportes, identificar los datos PII y borrarlos a mano. Sin embargo, hay un inconveniente en la escala de esto para la cantidad de informes que tenemos. Además para grandes volúmenes de texto, el criterio humano puede ser más proclive a cometer omisiones comparado con un método automático.

El resultado de estas pruebas fue que ninguno de estos métodos era suficiente por sí mismo y se terminó optando por un método híbrido que combina RegEx, el modelo de NER `PlanTL-GOB-ES/roberta- base-bne-capitel-ne`, y la revisión manual. Todo esto, combinado con el hallazgo de que estadísticamente la mayoría de los datos personales se encuentran al final de los reportes, nos permite llegar al siguiente proceso iterativo:

1. Usar un conjunto de RegEx que busquen los patrones mencionados más arriba, teniendo en cuenta el uso irregular de mayúsculas y signos de puntuación.

2. Para cada informe, a partir del primer patrón identificado eliminar todo lo que venga después hasta el final. Esta decisión se basa en la observación de que la mayoría

[5] https://huggingface.co/PlanTL-GOB-ES/roberta-base-bne-capitel-ner

[6] Una expresión regular (también llamada *RegEx*) es una secuencia de caracteres que define un patrón de búsqueda, utilizada principalmente para la coincidencia y manipulación de cadenas de texto. Las expresiones regulares se emplean ampliamente en el procesamiento de texto para localizar patrones específicos dentro de cadenas, extraer datos relevantes o realizar reemplazos y transformaciones complejas.

de los nombres y matrículas se encuentran al final de los informes, por lo que el riesgo de perder información útil es bajo. Dado que queremos minimizar la cantidad de información valiosa que se pudiera llegar a perder, ponemos un límite de 90 caracteres a eliminar como mucho. En caso de que fuera a eliminarse más de esa cantidad, entonces simplemente se genera un mensaje y no se modifica el texto en forma automática, para su posterior revisión manual.

3. Sobre el *corpus* resultante, usar el modelo de reconocimiento de entidades nombradas para identificar nombres que no fueron removidos usando las RegEx. Es importante destacar que muchas veces el modelo de NER identifica cosas con el tag PER que no son realmente nombres de médicos o pacientes (falsos positivos). Para simplificar el análisis, se agregó una lista de falsos positivos conocidos, para que en caso de que el NER los detecte no sean tenidos en cuenta. Notar que en este punto no se elimina nada.

4. A partir del punto anterior, actualizar los patrones de las RegEx para capturar los nombres que no fueron removidos en el punto 2. También actualizar la lista de falsos positivos del NER. Repetir desde el paso 1, para poder borrar los nuevos patrones identificados.

5. Una vez que no resulta posible seguir eliminando automáticamente los nombres que detectan el NER y las RegEx (porque implicarían superar el límite de 90 caracteres o porque son patrones únicos) se pasa a una etapa de corrección manual en la cual se eliminan los datos restantes. En nuestro caso, implicó revisar manualmente solo 90 informes, de los más de 80 mil que teníamos originalmente.

Como resultado de esto obtuvimos un *corpus*, para el que tenemos alto grado de certeza que no contiene datos identificatorios.

4.2.2. Pre-procesamiento de spradie-corpus-brat-format

En el caso de este *corpus*, como se trata de un conjunto ya preparado para una competencia pública, requiere un esfuerzo mucho menor en la parte de preprocesamiento. En realidad, la principal tarea a realizar es un cambio de formato de las anotaciones. Esto se debe a que los datos son provistos con un formato llamado Brat Standoff, que es generado por la herramienta de anotación utilizada. Sin embargo, la mayoría de los modelos de NER están pensados para esperar otro tipo de formato. El formato que es más comunmente aceptado para este tipo de tareas es el BIO (o algunas de sus variantes). A continuación explicamos brevemente cada uno de estos formatos, sus características principales y que implicancias trae aparejadas el paso de un formato al otro.

Formato Brat Standoff

Brat es una herramienta web[7] para anotación de textos. Esta herramienta permite generar distintos tipos de anotaciones pero en el contexto de nuestro *corpus* solo se utilizó una: las "anotaciones vinculadas al texto" (o *text-bound annotations*). Una anotación de este tipo consiste en asociar uno o más fragmentos de texto a algún tipo de entidad. Por simplicidad, todas las explicaciones que vienen a continuación se enfocan únicamente en

[7] `https://brat.nlplab.org/introduction.html`

este tipo de anotación. En caso de querer saber más sobre los otros tipos de anotaciones soportados por Brat (relaciones, eventos, modificaciones, etc.) o del formato en general, recomendamos la lectura de la documentación oficial[8].

El formato con el que trabaja la herramienta, llamado *Brat Standoff format*, especifica que por cada "documento" que se desea anotar (en nuestro caso, por cada informe de ecografía) se generan dos archivos con el mismo nombre pero diferente extensión: uno con extensión .txt (que contiene básicamente el documento sin ninguna modificación) y otro con extensión .ann (que contiene las anotaciones). Un ejemplo de ambos archivos, tomado de nuestro *corpus*, puede verse en la figura 4.2.

A continuación, explicamos el formato del archivo de anotaciones. Cada línea del archivo corresponde a una anotación. Para ilustrar, tomemos como ejemplo la primer línea que aparece en el cuadro inferior de la figura 4.2: `T1 Finding 15 35 disminuido de tamano`. Todas las líneas inician con un identificador de la forma Tn (en este caso, T1), siguiendo una convención propia de Brat. Luego, viene una tabulación seguida por el tipo de la entidad anotada (Finding). A continuación, separados por espacios simples vienen dos números (15 y 35 en nuestro ejemplo): estas son las posiciones de inicio y fin del fragmento de texto anotado. Dichas posiciones corresponden al texto guardado en el archivo .txt. Por último, viene una nueva tabulación seguida de una copia textual del fragmento correspondiente a la anotación ("disminuido de tamano"). Notar que esto último es principalmente una facilidad que brinda el formato, ya que teniendo los índices y el .txt no resulta estrictamente necesario contar con esto. Esta explicación resume la mayoría de las anotaciones, aunque existe una variación: Brat permite también hacer anotaciones de entidades discontinuas, es decir que varios fragmentos de texto no-contiguos son parte de una misma anotación. Un ejemplo de esto es la última línea (T24) que aparece en el ejemplo de la figura 4.2. Notar que en este caso, la parte de la anotación que indica las posiciones tiene la forma `341 355;381 405`. El ";" separa las posiciones de principio y cierre de cada uno de los fragmentos involucrados en la anotación, que en este caso son solo dos, pero podrían ser más.

Este formato permite realizar anotaciones complejas, como entidades discontinuas (recién vistas) y entidades solapadas (también conocidas como anidadas). Estas últimas son aquellas donde existe un solapamiento (total o parcial) entre dos o más anotaciones: es decir que un mismo fragmento de texto está incluido en más de una anotación. Volviendo al ejemplo de la figura 4.2, podemos observar las anotaciones T1 y T3 y notar que en ambos casos el fragmento anotado es el mismo ("disminuido de tamano") pero las entidades son diferentes (Finding y Measure respectivamente). En ese caso el solapamiento es total, pero también podría ser parcial, como ocurre con las anotaciones T13 y T17: la primera anota "cm" como Abbreviation, mientras que la segunda anota "20 cm aprox" como Measure. Notar que el fragmento anotado por T13 está incluido en el anotado por T17.

Este tipo de entidades complejas son muy comunes en el texto médico, haciendo de *Brat Standoff* un formato muy útil para este dominio. Sin embargo, el problema viene justamente porque la mayoría de los modelos de NER que se utilizan actualmente trabajan bajo la asunción de que no existen ni solapamientos ni discontinuidades [47]. Aunque hay diversas líneas de investigación en este sentido, no nos enfocamos en este problema aquí y optamos por llevar las anotaciones a un formato compatible con la mayoría de los modelos: el formato BIO.

[8] https://brat.nlplab.org/standoff.html

Texto del informe (84966_brat.txt)

18a 2m.
Higado disminuido de tamano, heterogeno.
presenta areas nodulares ecogenicas irregulares en segmento VII y VIII, con area central con contenido liquido y ecos en su interior.
No se logra visualizar vena porta.
Arteria hepatica y VSH vesibles.
No se visualizan vasos mesentericos y vena esplenica por importante interposicion de aire esplenomegalia homogenea de 20 cm aprox con colaterales en hilio.
Liquido tabicado en pelvis.
Ambos rinones de estructura conservada.
Se sugiere correlacionar con TC realizada para mejor visualizacion de las estructuras vasculares .

Anotaciones (84966_brat.ann)

T1	Finding 15 35	disminuido de tamano
T2	Anatomical_Entity 8 14	Higado
T3	Measure 15 35	disminuido de tamano
T4	Anatomical_Entity 237 240	VSH
T5	Anatomical_Entity 268 286	vasos mesentericos
T6	Finding 162 166	ecos
T7	Finding 407 423	Liquido tabicado
T8	Location 427 433	pelvis
T9	Anatomical_Entity 206 216	vena porta
T10	Negation 183 205	No se logra visualizar
T11	Negation 251 267	No se visualizan
T12	Abbreviation 237 240	VSH
T13	Abbreviation 372 374	cm
T14	Anatomical_Entity 218 234	Arteria hepatica
T15	Abbreviation 375 380	aprox
T16	Finding 385 396	colaterales
T17	Measure 369 380	20 cm aprox
T18	Anatomical_Entity 400 405	hilio
T19	Anatomical_Entity 549 571	estructuras vasculares
T21	Anatomical_Entity 289 303	vena esplenica
T24	Finding 341 355;381 405	esplenomegalia con colaterales en hilio

Fig. 4.2: Ejemplo de anotaciones en formato Brat Standoff para el informe con código identificador 84966 del *corpus* de SpRadIE. Siguiendo la especificación del formato, hay dos archivos por cada informe: un archivo con extensión .txt, que contiene el texto original del informe (y se muestra en el cuadro superior) y un archivo con extensión .ann, que contiene las anotaciones para dicho informe (y se muestra en el cuadro inferior). Por claridad, se removieron algunas anotaciones del ejemplo.

Formato BIO

El formato BIO recibe su nombre de las siglas de las palabras en inglés *Beginning-Inside-Outside*, que hacen referencia a la modalidad con la que se realizan las anotaciones de los tokens: cada *token* se etiqueta con una B-NOMBRE-DE-LA-ENTIDAD si es el comienzo de una entidad, con una I-NOMBRE-DE-LA-ENTIDAD si está dentro de una entidad pero no es el comienzo, y O si no pertenece a ninguna entidad. Por ejemplo, para la entidad Location ahora pasamos a tener dos etiquetas posibles: B-LOCATION y I-LOCATION. Y lo mismo con el resto de las entidades.

De esta forma, cada *token* que compone el texto es anotado con exactamente una etiqueta (incluyendo O). Las únicas reglas son que el primer *token* de una entidad se etiquete con una B-etiqueta, siendo "etiqueta" el tipo de la entidad nombrada (por ej. Location), mientras que todos los *tokens* consecutivos, que también sean parte de la misma entidad, deben llevar una I-etiqueta. En cuanto se marca un *token* con una O u otra B-etiqueta, se considera que la anotación de la entidad anterior está terminada. Este mecanismo implica que no sean posibles los solapamientos ni las discontinuidades. En la figura 4.3 se presenta un ejemplo de anotación para dos oraciones de un informe.

```
HIGADO  4   10  B-Anatomical_Entity
:       10  11  O
tamano  12  18  O
y       19  20  O
ecoestructura 21  34  O
normal  35  41  O
.       41  42  O

VIA     43  46  B-Anatomical_Entity
BILIAR  47  53  I-Anatomical_Entity
intra   54  59  I-Anatomical_Entity
y       60  61  I-Anatomical_Entity
extrahepatica  62  75  I-Anatomical_Entity
:       75  76  O
no      77  79  B-Negation
dilatada 80  88  B-Finding
.       88  89  O
```

Fig. 4.3: Fragmento de un archivo en formato BIO. La primer columna incluye el *token* (en este caso se tokenizó por palabras), la segunda columna tiene la posición de inicio en el documento, la tercera la posición de fin, y la cuarta la etiqueta en formato BIO.

Conversión del formato y sus implicancias

Por lo visto anteriormente, se deduce que al realizar la conversión de Brat a BIO se va a perder información siempre y cuando existan solapamientos o discontinuidades. Como el *corpus* de SpRadIE las contiene, esto va a pasar.

A la hora de escoger con qué entidades quedarnos frente a una discontinuidad o solapamiento existen distintas estrategias posibles. Nosotros escogemos las siguientes:

- Para las entidades discontinuas, nos quedamos con el segmento completo que va desde la posición inicial más pequeña hasta la posición final más grande, de entre los fragmentos anotados. Esto provoca que se incluyan fragmentos que originalmente no están anotados como parte de la entidad. Por ejemplo, en el texto "via biliar intra y extrahepatica" se suelen anotar dos entidades anatómicas distintas: "via biliar intrahepatica" y "via biliar extrahepatica". Siguiendo el criterio planteado aquí, en ambos casos la nueva anotación va a ser: 'via biliar intra y extrahepatica" como una sola Anatomical Entity. Este método asegura que los modelos entrenados aprendan a etiquetar todos los *tokens* de la entidad anotada originalmente, pero también implica que pueden ser más propensos a los falsos positivos ya que les estamos enseñando a etiquetar cosas de más.

- Para las entidades solapadas nos quedamos con la que sea más larga. Esto favorece a que el modelo etiquete correctamente mayor cantidad de caracteres, pero también significa que hay entidades que puede no aprender a identificar bien debido a que las ve mucho menos en su entrenamiento. Un ejemplo de esto, sobre el que volveremos más adelante, son las entidades de tipo Abbreviation, que suelen estar anidadas dentro de entidades más grandes como Measure (por ej. "30 cm."), Anatomical Entity (por ej. "riñón izq"), etc.

4.3. Embeddings y Language Models

En esta sección detallamos los modelos utilizados, cómo se realizó su entrenamiento y también documentamos las decisiones importantes que tomamos.

Dada la naturaleza de los informes, que están en español e incluyen un frecuente uso de abreviaturas, errores ortográficos y términos *de nicho* que no son frecuentes en el uso general del idioma, tomamos algunos cuidados al momento de escoger los modelos a utilizar. Nos enfocamos en modelos pre-entrenados específicamente para el idioma español y en general ajustados al dominio bio-médico. También priorizamos que utilicen *tokens* a nivel de caracteres o *sub-palabras* (lo que permite mayor robustez frente a palabras por fuera del vocabulario de entrenamiento). Teniendo en cuenta este último criterio se descarta a los métodos más clásicos como Word2Vec y GloVec que solo aprenden a embeber palabras completas. Los modelos escogidos pueden clasificarse en tres arquitecturas distintas: *Transformers*, BiLSTM y FastText.

Con lo que respecta a los *Transformers* (ver sección 2.6), escogimos el siguiente listado de modelos pre-entrenados para usar como base para los experimentos y *fine-tuning*. Todos los modelos son de acceso público y pueden encontrarse en el *hub* de modelos de HuggingFace[9].

- BETO CASED[10]: Modelo pre-entrenado [11] basado en la arquitectura de BERT pero entrenado con un *corpus* de dominio general en español [10]. Este es un transformer entrenado de cero, razón por la cual el *tokenizer* asociado también fue entrenado específicamente para el *corpus* usado. En este caso el *tokenizer* preserva el uso de mayúsculas y minúsculas.

[9] HuggingFace es una plataforma *open source* que permite publicar y descargar modelos de Machine Learning, siendo particularmente popular en la comunidad de NLP. Para más información puede visitarse su página web: https://huggingface.co/

[10] https://huggingface.co/dccuchile/bert-base-spanish-wwm-cased

- BETO UNCASED[11]: Igual que BETO CASED pero el *tokenizer* se encarga de llevar todas las palabras a minúsculas antes de enviarlas al modelo.
- BETO CLINICAL WL[12]: Modelo previamente adaptado con *fine-tuning* a partir de BETO UNCASED para el *corpus* de ChileanWL [5]. Notar que al ser un caso de *fine-tuning* el *tokenizer* es el mismo usado por BETO UNCASED.
- ROBERTA BNE[13]: Modelo pre-entrenado [29] utilizando el *corpus* en español más grande conocido hasta la fecha, con un total de 570 GB de texto procesado para este trabajo, compilado a partir de las rastreos web realizados por la Biblioteca Nacional de España desde 2009 hasta 2019.
- ROBERTA BIO CLINICAL[14]: Modelo pre-entrenado [12] con un *corpus* compuesto por varios *corpora*, biomédicos en español, recopilados de *corpus* disponibles públicamente y rastreadores, así como un *corpus* clínico del mundo real, recopilado a partir de más de 278,000 documentos y notas clínicas.
- ROBERTA CLINICAL WL[15]: Modelo de RoBERTa basado en el anterior pero adaptado sobre el *corpus* de ChileanWL [55].

También, siguiendo la línea de modelos contextuales y en este caso apuntando a tokenización por caracteres y uso de *Bidirectional Long Short-Term Memory* (BiLSTM), en lugar de Transformers, probamos la arquitectura Flair [2] (ver 2.5.1) y la biblioteca homónima[16]. Particularmente, dado que en este caso también resulta muy intensivo computacionalmente el proceso de entrenamiento, partimos de un modelo público[17] entrenado sobre el *corpus Chilean Waiting List* (ChileanWL) [55]. Para ser más específicos, en realidad se trata de dos modelos distintos, debido a la naturaleza de los *embeddings* generados por BiLSTMs: un modelo genera representaciones a partir de tratar de predecir la siguiente palabra en la oración (`es-clinical-forward`), mientras que el otro modelo intenta predecir la palabra que lo precede (`es-clinical-backward`) en una oración. Típicamente se combinan ambas representaciones (usando concatenación de vectores) para tener información bidireccional del texto. Para referenciarlo más fácilmente, llamamos a esta combinación de modelos CLINICAL FLAIR. Vale aclarar que en este punto solo estamos combinando dos modelos pre-entrenados, pero aún no aplicamos *fine-tuning* sobre nuestro *corpus*.

Adicionalmente para contrastar con métodos no-contextuales pero que tienen la capacidad de embeber palabras fuera del vocabulario creamos dos *embeddings* de FastText (ver sección 2.3.1) sin partir de modelos pre-entrenados.

- FASTTEXT100: Modelo completamente entrenado por nosotros usando el algoritmo de FastText sobre el *corpus* `anonymized-ultrasounds-80k-with-diacritics`. En este caso se generan vectores de 100 dimensiones.

[11] https://huggingface.co/dccuchile/bert-base-spanish-wwm-uncased
[12] https://huggingface.co/plncmm/beto-clinical-wl-es
[13] https://huggingface.co/PlanTL-GOB-ES/roberta-base-bne
[14] https://huggingface.co/PlanTL-GOB-ES/roberta-base-biomedical-clinical-es
[15] https://huggingface.co/plncmm/roberta-clinical-wl-es
[16] Es importante mencionar que la palabra Flair es ambigua: por un lado es el nombre de una arquitectura basada en BiLSTMs descripta en [2] y por el otro lado es una popular biblioteca para desarrollo de modelos de NLP (https://flairnlp.github.io/) desarrollada por el mismo equipo que creo dicha arquitectura. Durante este trabajo haremos uso de ambas herramientas y tratamos de ser explícitos cuando nos referimos a una u otra.
[17] https://flairnlp.github.io/docs/tutorial-embeddings/flair-embeddings

- FASTTEXT200: Idéntico al anterior pero generando *embeddings* de 200 dimensiones.

4.3.1. Proceso de entrenamiento

Para lo que sigue se hizo una división de los informes de `anonymized-ultrasounds-80k`: 4440 se separaron para hacer después análisis de *clusters* sobre eso. Por lo tanto quedaron un poco menos de 70 mil informes (69556 exactamente) para utilizar durante el *fine-tuning* o entrenamiento.

Modelos basados en Transformers

Para el entrenamiento de los modelos basados en Transformers no realizamos en ningún caso entrenamientos desde cero sino que aplicamos la técnica de *domain adaptation* (ver sección 2.7) a partir de modelos pre-entrenados. La forma de hacer esto para modelos basados en BERT (lo que incluye también a modelos basados en RoBERTa) es realizando un entrenamiento del modelo base para la tarea de Masked Language Model (ver sección 2.4.1) sobre el *corpus* específico al que se desea adaptar (en nuestro caso `anonymized-ultrasounds-80k`).

Para realizar el *fine-tuning* se utilizaron los hiperparámetros estándar definidos para BERT y RoBERTa en sus respectivos papers [19] [41]. La razón de no hacer una selección de hiperparámetros más sofisticada es que resulta poco costo-efectivo por lo demandante que son los entrenamientos de estos modelos (incluso los fine-tunings) en términos de recursos computacionales y tiempo. Además en nuestro caso particular donde lo que más nos importa es la utilidad de las representaciones internas de los modelos, no resulta tan simple tener una evaluación efectiva que permita guiar la búsqueda en el espacio de hiperparámetros con seguridad. No necesariamente siempre la métrica usada para evaluar el modelo durante su entrenamiento (por ejemplo, la *perplexity* en el caso de los *Masked Language Models* como BERT) se condice con el *embedding* más útil luego. Sin embargo hay algunas decisiones relacionadas al proceso de tokenización y a la forma de pasarle los datos al modelo que puede ser bueno detallar.

Primero se deben tokenizar todos los informes. Esto se hace siempre con el tokenizador asociado al modelo pre-entrenado. La razón de esto es que el tokenizador determina el vocabulario sobre el que el modelo fue entrenado y por lo tanto también el vocabulario sobre el que puede ser aplicado. Sin embargo en este punto hay que tomar una decisión que puede tener consecuencias luego: tanto BERT como RoBERTa tienen una ventana de contexto de exactamente 512 *tokens*, lo que significa que aquellos informes que tengan menos *tokens* deben agregar *tokens de padding*[18] y los que tengan más deben truncarse[19]. Esto según el caso concreto puede traer problemas. En nuestro caso particular, los informes que tienen más de 512 *tokens* y perderían información al truncarse están en el orden del 1 %, por lo que no es un gran inconveniente por ese lado. En la gran mayoría de los casos se está muy por debajo de este número (el 95 % de los informes tiene 181 *tokens* o menos) sin embargo existe otro problema: la gran diversidad de largos de los documentos. Esto resulta problemático ya que termina generando una ineficiencia computacional considerable al momento de realizar el entrenamiento con *batches* que tienen longitud variable y requieren

[18] Los *tokens* de *padding* son un valor especial que representa un *token* de relleno que solo sirve para asegurar que el tamaño de entrada sea el correcto, sin aportar información adicional.

[19] El proceso de truncado consiste en descartar todos aquellos *tokens* que vengan después del máximo soportado.

un ajuste de *padding* dinámico. La solución que adoptamos es una de uso habitual cuando se cuenta con tanta diversidad de largos entre los textos: unificamos todos los *tokens* en un solo gran texto, adicionando *tokens* especiales que representan separadores de documentos, y luego lo dividimos en chunks de un tamaño fijo. De esta manera no se pierde texto y se mantienen batches de tamaño fijo. Este procedimiento se conoce como *chunking*. En nuestro caso escogimos hacer chunks de 128 *tokens*. El principal factor de esta decisión es la restricción que nos impone la memoria RAM de la GPU que fue utilizada para entrenar: con más *tokens* por *chunk* suceden errores por falta de memoria.

Otra decisión importante a tomar respecto de la entrada del entrenamiento es cómo generar las máscaras que el modelo tratará de completar. Una primera decisión es si las máscaras deberían aplicarse a nivel de los *tokens* o de las palabras (que en general se componen de varios tokens). Si optamos por la primera opción puede suceder que una palabra se enmascare parcialmente (por ejemplo, "Higado" podría quedar como "<MASK>ado"). En el segundo caso, nos aseguraríamos que se ponga un *token* de máscara por cada *token* que componga la palabra (en el ejemplo previo, "Higado" quedaría como "<MASK><MASK>"). Una duda que podría surgir en este segundo caso es por qué no reemplazar completamente la palabra "Higado" por un solo *token* "<MASK>", en lugar de dos. La explicación es que si hiciéramos eso, dado que el modelo de lenguaje solo puede predecir un *token* por cada máscara, jamás sería capaz de predecir la palabra "Higado" completa. Nosotros optamos por esta opción de enmascarar palabras completas.

La otra decisión importante, referente al enmascaramiento, es si las máscaras deberían definirse una única vez antes de entrenar o si por el contrario deberían definirse dinámicamente (y bajo una distribución de probabilidad sobre los *tokens*/palabras) cada vez que se genera un *mini-batch* durante el entrenamiento. Dado que las herramientas lo permiten, optamos por la segunda opción que ofrece una mayor diversidad de escenarios para el modelo lo cual puede ser bueno para evitar *overfitting*. La distribución escogida fue una `Bernoulli(0.15)`, lo que significa que al armar un nuevo *mini-batch* cada palabra tiene un 15 % de probabilidad de enmascararse, independientemente del resto de las palabras. Es bueno notar que como la probabilidad es por palabra y cada palabra puede estar compuesta por múltiples *tokens*, entonces la probabilidad de enmascarar un *token* puede ser de hecho superior al 15 % y depende de cuántos *tokens* más conformen la misma palabra.

En cuanto a la duración del entrenamiento, empíricamente determinamos que a partir de los 15 *epochs* los modelos empezaban a converger en su función de pérdida y, además, se obtenían resultados similares o mejores que con 30 *epochs* al observar los resultados de la tarea de NER con SpRadIE. Por lo tanto optamos por entrenar todos los modelos durante 20 *epochs*, utilizando una funcionalidad de la biblioteca de HuggingFace que permite quedarse siempre con la epoch que minimiza la función de pérdida a optimizar. De esta forma el resultado es más robusto frente a la volatilidad que puede aparecer en el entrenamiento.

Modelos basados en FastText

En el caso de los modelos de FastText sí realizamos entrenamientos completamente de cero, utilizando únicamente los datos de `anonymized-ultrasounds-80k`. La decisión de ir por este camino y no intentar basarnos en otros *embeddings* de FastText pre-entrenados se sustenta en gran medida sobre los resultados presentados en la tesis de licenciatura de Minces Müller [44]. En dicho trabajo, el autor se basa en el mismo conjunto de datos que nosotros y evalúa el uso de distintas variantes de FastText, concluyendo que los mejores

resultados los obtiene con el modelo entrenado de cero. Aunque vale aclarar que el problema que se trata es distinto (detección de relaciones), no deja de ser un resultado relevante y preferimos no profundizar mucho más ahí para enfocarnos en otros métodos.

A diferencia de los modelos basados en Transformers, aquí sí tenemos libertad para escoger el *tokenizer* deseado. Probamos varios enfoques distintos para esto, utilizando desde técnicas simples como separación por espacio o expresiones regulares ligeramente más complejas, hasta herramientas populares en el ámbito de NLP como NLTK[20] y spaCy[21]. Las consideraciones más importantes que tuvimos fueron: que la tokenización no fuera destructiva (esto significa que no se pierden caracteres al tokenizar, por ejemplo los signos de puntuación) y que el resultado de fuera lo más similar posible a la tokenización realizada por el script que convierte los informes de SpRadIE de formato Brat a BIO (ver 4.1). En general las diferencias fueron pocas, siendo una de la más notorias como se separaban las mediciones: spaCy separa "7cm" en "7" y "cm", mientras que el método basado en RegEx conserva "7cm" como un solo *token*. Si bien el comportamiento de spaCy puede parecer el más deseable, el script de conversión de formatos hace lo segundo. Por lo tanto, terminamos optando por el método basado en RegEx (el código Python puede verse en el Apéndice 7.3).

Para entrenar el modelo, utilizamos la implementación de FastText que provee la biblioteca gensim[22]. Los hiperparámetros se mantuvieron mayoritariamente en sus valores por defecto:

- Tamaño de la ventana de contexto: 5
- Cantidad mínima de apariciones de una palabra en el *corpus* para no ser ignorada: 5
- Largo mínimo de n-grams de caracteres: 3
- Largo máximo de n-grams de caracteres: 6

Los únicos hiperparámetros que tocamos fueron la cantidad de *epochs* que configuramos en 30[23] (aproximadamente media hora de entrenamiento) y el tamaño de los vectores: generamos vectores de 100 y 200 dimensiones respectivamente.

Modelos basados en Flair

Como mencionamos anteriormente, para los modelos de lenguaje basados en Flair recurrimos a la técnica del *fine-tuning* (*domain adaptation*) del mismo modo que lo hicimos con los Transformers. En este caso solo tenemos un modelo a tunear que es CLINICAL FLAIR.

En este caso la tokenización se resuelve de forma más sencilla: trabajamos con *tokens* a nivel de caracteres, aunque vale aclarar que internamente la implementación de Flair

[20] `https://www.nltk.org/`
[21] `https://spacy.io/`
[22] `https://radimrehurek.com/gensim/models/fasttext.html`
[23] La cantidad de *epochs* fue determinada empíricamente y basandonos en las mejores prácticas que encontramos en la comunidad. Lo idea sería realizar un monitor de la *loss funcion* para analizar los quiebres de comportamiento y evitar caer en un sobreajuste. Lamentablemente, a día de hoy la implementación de Gensim no soporta hacer un monitoreo de dicha métrica. Ver `https://github.com/piskvorky/gensim/issues/2617`.

segmenta estos *tokens* por oraciones. Sobre los hiperparámetros utilizados, mayoritariamente nos basamos en los valores por defecto nuevamente. En términos de estructura de modelo, lo único que especificamos fue la dimensión del vector al cual se mappean los *inputs* y que luego es consumido por la LSTM que termina generando el embedding: le pusimos un valor de 100. Esta decisión nuevamente se basa es pruebas empíricas, teniendo en cuenta que es un hiperparámetro que tiene un alto impacto en el tiempo y recursos de entrenamiento.

A diferencia de los modelos mencionados anteriormente, aquí la elección de la cantidad de *epochs* puede hacerse con un criterio un poco más metódico. Debido a que la biblioteca de Flair cuenta con la posibilidad de usar *early stopping*[24], pudimos poner una tolerancia de 5 *epochs* sin mejora. Además, Flair lleva trazabilidad de cual fue la mejor epoch en términos de *perplexity*, por lo que siempre nos podemos quedar con la iteración que mejor dió al finalizar el entrenamiento.

Como se menciona al principio de la sección, en realidad no se trata de un modelo sino de dos (uno *forward* y otro *backward*) cuyos resultados se concatenan en un único vector. Los hiperparámetros descriptos aplican al entrenamiento de ambos modelos. Cada modelo produce vectores de 2048 dimensiones, por lo que el resultado del modelo concatenado es de 4096 dimensiones, siendo por bastante diferencia la representación más grande que probamos aquí (los vectores basados en BERT-base tiene 768 dimensiones).

4.4. Modelos de NER

En la sección anterior se explica cómo fue el proceso de selección de modelos de *embeddings* y lenguaje, así como también el proceso de entrenamiento o *fine-tuning* de cada uno según correspondiera. Dichos modelos únicamente tienen la capacidad de generar representaciones vectoriales a partir de un texto dado. En esta sección, cubrimos la siguiente etapa que es la entrenar los modelos finales que serán capaces de convertir estas representaciones vectoriales en predicciones sobre cuál debería ser la etiqueta de cada palabra, basándonos en las entidades de SpRadIE. Para esto se realizarán entrenamientos sobre el *corpus* `spradie-corpus-bio-format` (ver 4.2.2). Para diferenciarlos de los modelos de *embeddings* y lenguaje, nos referiremos a estos modelos como *modelos de NER* o *sequence taggers* (del inglés, "etiquetador de secuencias").

Tanto para Flair como FastTex usaremos el mismo *sequence tagger*, que es uno de los que disponibiliza la biblioteca de Flair. En el caso de los BERT y RoBERTa, el procedimiento es un poco diferente y lo realizamos con la misma biblioteca de HugginFace que usamos para su *fine-tuning* previo.

Una aclaración importante que aplica a todos los entrenamientos que se hicieron de modelos para reconocimiento de entidades nombradas: durante el desarrollo de los métodos y las pruebas preliminares se trabaja siempre con las particiones de train, devSameSample y devHeldOut de SpRadIE. La primera se usa para los entrenamientos y las otras dos para la validación de los resultados durante el desarrollo de los modelos. Por simplicidad a veces hablamos de partición de Validation para hacer referencia a la combinación de devSameSample y devHeldOut. Sin embargo en la última ejecución, se entrena utilizando

[24] *Early stopping* es una técnica de entrenamiento que consiste en monitorear una métrica específica durante el entrenamiento (por ejemplo, *perplexity*) y cortar automáticamente el entrenamiento cuando pasan una cierta cantidad de *epochs* sin mejoras. A esta cantidad de iteraciones sin mejoras se la llama "tolerancia".

todos los datos combinados de train, devSameSample y devHeldOut, y se evalúa usando la partición de test. Los resultados de esta última evaluación son los que se reportan en el capítulo 5.

4.4.1. Entrenamiento para NER basado en Transformers

En este caso tomamos los modelos de BERT y RoBERTa presentados en la sección 4.3, tanto en sus versiones base como en sus versiones con *domain adaptation*, pero esta vez les aplicamos *task adaptation* (ver sección 2.7). Esto básicamente consiste en que tomamos los modelos que fueron entrenados para predecir la palabra más probable dentro de un contexto (*Masked Language Model*) y lo adaptamos para que ahora sea capaz de predecir las entidades de SpRadIE (NER o *sequence tagging*).

La forma de hacer esto consiste en reemplazar la capa de salida (*header*) de los modelos entrenados para la tarea de Masked Language Model por una capa especial de clasificación, con una neurona por cada clase posible a predecir. La etiqueta a predecir será la correspondiente a la unidad que retorne el valor más grande. Esta última capa tiene también pesos entrenables, sin embargo inicialmente se agrega con valores random.

Nuevamente se usan valores de referencia para los hiperparámetros. En cuanto a la forma de pasarle los datos de entrenamiento al modelo, se pasa oración por oración y no informes completos. Esto es una decisión que tomamos por simplicidad, sin embargo es algo que podría profundizarse para ver si realmente es lo mejor. Una de las ventajas de esto es que ninguna oración supera el límite de 512 *tokens* lo que simplifica la ingesta. La generación del padding necesario se realiza de forma automática y dinámica durante el entrenamiento.

Posiblemente el aspecto más complejo de todos, antes de poder empezar a entrenar, es el de alinear correctamente las etiquetas que aparecen en `spradie-corpus-bio-format` con los *tokens* generados por los tokenizadores propios de cada modelo. Recordemos que las etiquetas en el *corpus* en formato BIO están asignadas a nivel palabra, sin embargo los tokenizadores usados por los modelos basados en BERT descomponen las palabras en varios *tokens* de sub-palabras. Esto conlleva a que sea necesario mantener trazabilidad de a que palabra original corresponde cada *token* para luego poder asignarle la etiqueta correspondiente. Fuera de esto, no es necesario realizar mayores acciones al momento de pre-procesar los datos. Dado que los informes son mayoritariamente cortos El padding se resuelve dinámicamente, agregando lo que haga falta para cada informe.

4.4.2. Entrenamiento para NER basado en Flair o FastText

En este caso, utilizaremos nuevamente un modelo de BiLSTM, pero en lugar de que sea para generar *embeddings* será para la tarea de reconocer las entidades, tomando como *input* los *embeddings* generados (por FastText o por Flair). Los resultados de salida de esta BiLSTM son procesados por una capa de *Conditional Random Fields* (CRF) [25] que es lo que termina determinando la probabilidad de cada una de las etiquetas posibles.

[25] Los *Conditional Random Fields* son modelos probabilísticos discriminativos utilizados para etiquetado secuencial, donde se consideran dependencias entre etiquetas adyacentes para optimizar asignaciones globales en lugar de decisiones independientes. Cuando se combinan con BiLSTMs, las representaciones contextuales generadas por las redes neuronales bidireccionales se utilizan como entradas al CRF, mejorando la precisión del etiquetado al capturar tanto el contexto de las palabras como las relaciones entre etiquetas consecutivas.

En este caso no es necesario realizar un alineamiento adicional, ya que la biblioteca se encarga automáticamente de gestionar esto. Aquí también el entrenamiento y predicción se realizan oración por oración.

4.5. Evaluación extrínseca

En esta sección se presenta una descripción detallada de cómo se evalúan los modelos, incluyendo las métricas utilizadas y el proceso que se sigue. La figura 4.4 muestra, en forma simplificada, cuáles son los principales grupos de tareas que se tienen que realizar para poder llegar al punto de analizar los resultados. Mientras que la figura 4.5 expande cada uno de estos grupos y muestra las tareas con mayor detalle, incluyendo su orden y dependencias. En total hay cuatro líneas de evaluación, tres extrínsecas y una intrínseca. En esta sección nos enfocaremos en las extrínsecas mientras que la intrínseca se detalla en la sección 4.6.

Las tres principales líneas que seguimos para la evaluación extrínseca son:

- Comparación entre los modelos entrenados usando intervalos de confianza: buscamos comparar de manera lo más estadísticamente robusta posible los resultados de los distintos modelos aquí entrenados.
- Estudio de ablación: a través de reemplazar componentes de la arquitectura de los modelos, intentamos aislar el efecto que tienen los *embeddings* entrenados sobre el resultado final de la tarea de NER.
- Comparación contra los resultados publicados de la competencia de SpRadIE: buscamos analizar si logramos mejorar el estado del arte en una tarea concreta. Aquí consideramos no solo los modelos de Machine Learning sino también la heurística para identificar abreviaturas que se describe más adelante en esta sección. Como parte de esta sección, realizamos también un análisis manual de los errores para determinar oportunidades de mejora en el entrenamiento de nuestros modelos.

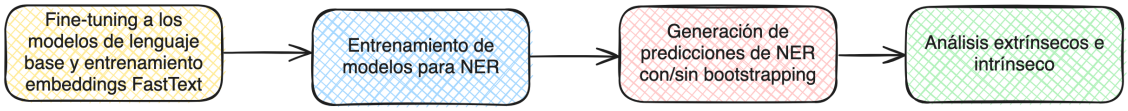

Fig. 4.4: Diagrama simplificado que muestra los principales cuatro conjuntos de tareas a realizar en el trabajo, para poder llegar a los análisis de resultados.

Las métricas utilizadas para evaluar la tarea de NER son exactamente las mismas que se usaron en la competencia de SpRadIE [17]. El contenido presentado en esta subsección está mayormente tomado de esa publicación. En total se computan seis métricas *Precision*, *Recall* y *F1-Score* (o simplemente *F1*) para *match* parcial, y las mismas tres pero evaluadas con *match* exacto. *Match* parcial significa que se da una ponderación relativa a predecir una anotación en forma parcialmente correcta (es decir que se reconocen cosas de menos o de más) mientras que el *match* exacto solo considera predicciones exactas. Para SpRadIE, la métrica que se tomó como referencia para escoger a un ganador fue el F1-Score para *match* parcial, considerando que una anotación aunque no sea perfecta puede ser útil en muchas tareas.

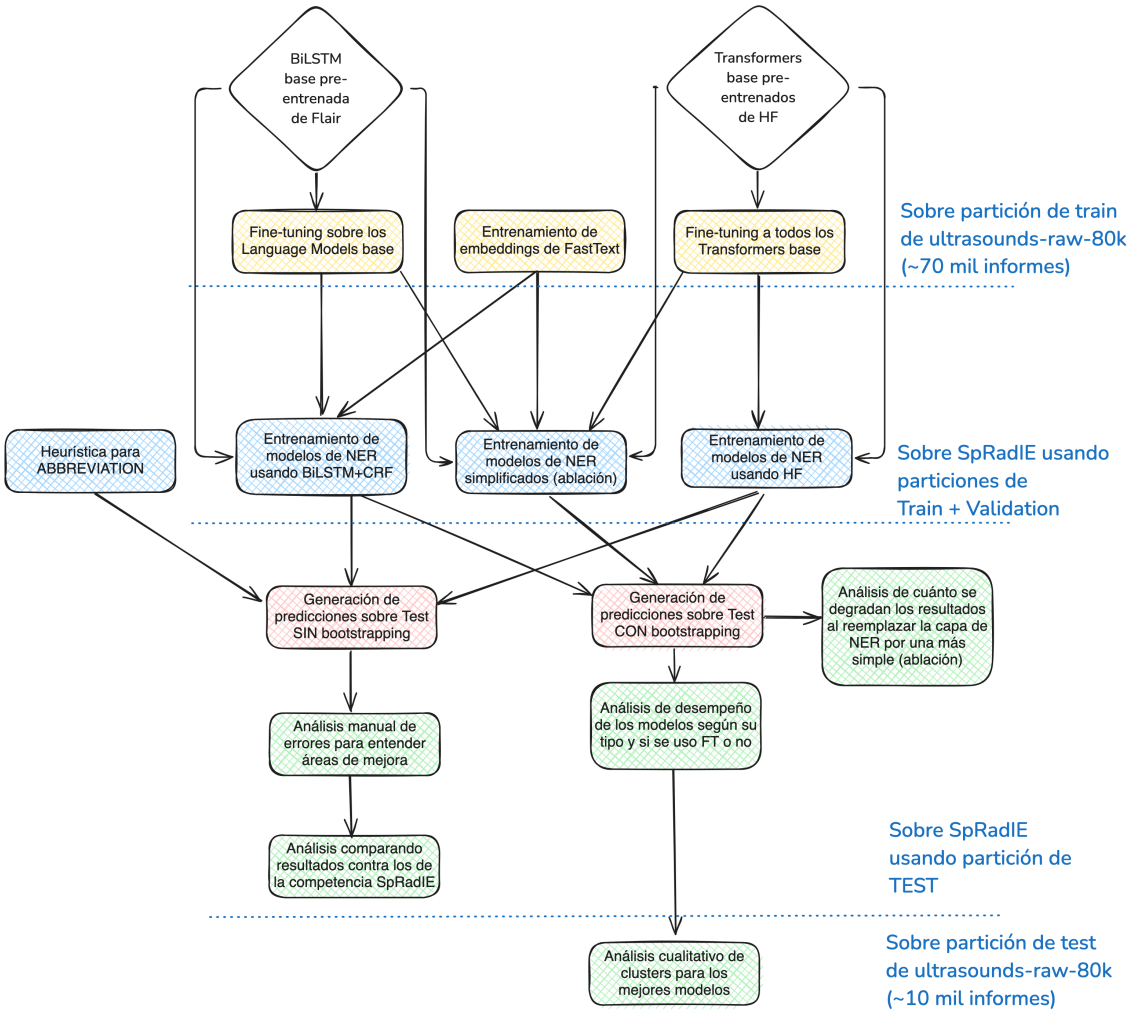

Fig. 4.5: Diagrama muestra el proceso de evaluación de los modelos, desde los entrenamientos realizados hasta llegar a los diferentes análisis realizados. Los pasos se ordenan cronológicamente de arriba hacia abajo, manteniendo en el mismo nivel aquellas tareas que pueden ejecutarse en paralelo. Se incluyen tanto las evaluaciones extrínsecas como las cualitativas. Para representar a los modelos pre-entrenados de los que partimos, utilizamos rombos. Los rectángulos son usados para representar las distintas acciones que a su vez generan resultados para ser consumidos en la siguiente acción. El color de cada rectángulo representa a cuál de los cuatro grupos de tareas, diagramados en la figura 4.4, pertenece. Algunas siglas son utilizadas por una cuestión de espacio: HF (HugginFace), FT (*Fine Tuning*) y WE (*Word Embeddings*).

Una aclaración importante es que todas las métricas reportadas en este trabajo se computan utilizando *micro-average*. Esto implica que todas las instancias de predicciones y referencias tienen el mismo peso al calcular *Precision*, *Recall* y *F1-score*, independientemente de a qué entidad correspondan. Este enfoque se diferencia del *macro-average*, donde las métricas se computan por cada clase de entidad primero y luego se promedian dichos resultados para obtener la métrica general (por ejemplo, primero se computaría *Precision* para cada una de nuestras 10 entidades por separado, y luego se haría el promedio de esas 10 métricas para obtener el *macro-average Precision*). La principal diferencia entre ambos métodos radica en como se comportan frente a problemas con clases desbalanceadas (es decir, que hay clases que concentran más ejemplos que otras). El *micro-average*, al considerar todos los ejemplos con la misma importancia, resulta más afectado por los resultados obtenidos para las clases mayoritarias. Mientras que el *macro-average* le asigna la misma importancia a todas las clases, independientemente de si una representa el 99 % de los casos y otra solo el 1 %. Ambas son útiles y la elección de una u otra depende del caso concreto y lo que se quiera priorizar.

Para el cálculo de todas estas métricas, lo primero que hay que hacer es computar el *índice de Jaccard*, que consiste en una medida de similitud entre la referencia y la predicción. En término de conjuntos, la forma de calcularlo consiste en hacer un ratio de la intersección entre la referencia y la predicción sobre la unión de las mismas. La función J expresa el cálculo del *índice Jaccard* en el contexto de similitud entre fragmentos de texto:

$$J(\mathit{ref}, \mathit{pred}) = \frac{\text{long}(\text{solapamiento}(\mathit{ref}, \mathit{pred}))}{\text{long}(\mathit{ref}) + \text{long}(\mathit{pred}) - \text{long}(\text{solapamiento}(\mathit{ref}, \mathit{pred}))} \tag{4.1}$$

donde *ref* es el texto de referencia anotado en nuestro *ground truth*, *pred* es el texto reconocido por nuestro modelo, *long* es la función que devuelve el largo de una cadena de caracteres y *solapamiento* es una función que devuelve la sub-cadena de caracteres más larga que es compartida por dos cadenas, teniendo en cuenta el orden de los caracteres. Notar que si *ref* y *pred* son iguales, entonces la función vale 1. Mientras que si *ref* y *pred* no comparten nada en común, entonces el índice da 0.

A partir de este índice definimos las métricas a evaluar de la siguiente forma:

$$\text{PREC}_{\text{parcial}} = \frac{\sum_{(\text{ref},\text{pred}) \in M} J(\text{ref}, \text{pred})}{P} \tag{4.2}$$

$$\text{REC}_{\text{parcial}} = \frac{\sum_{(\text{ref},\text{pred}) \in M} J(\text{ref}, \text{pred})}{R} \tag{4.3}$$

$$\text{PREC}_{\text{exacta}} = \frac{|\{(\text{ref}, \text{pred}) \in M : J(\text{ref}, \text{pred}) = 1\}|}{P} \tag{4.4}$$

$$\text{REC}_{\text{exacta}} = \frac{|\{(\text{ref}, \text{pred}) \in M : J(\text{ref}, \text{pred}) = 1\}|}{R} \tag{4.5}$$

donde P es el número total de anotaciones predichas, R es el total de anotaciones de referencia y M es un conjunto de pares de referencias y anotaciones dado. El cómo definir M no es trivial y en SpRadIE se escoge resolver esto con una herurística de tipo *greedy* que itera sobre las anotaciones predichas y las empareja con la mejor anotación del *ground truth*.

Por último, a partir de los valores calculados para Precision y Recall podemos computar el F1 siguiendo la siguiente fórmula:

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.6}$$

Notar que la misma fórmula sirve tanto para *match* parcial como exacto, y el tipo del F1 estará dado por el tipo de Precision y Recall utilizados.

Para asegurar que nuestros resultados sean comparables con los de la competencia, calculamos todas estas métricas utilizando el *script* provisto en la página de SpRadIE[26].

4.5.1. Intervalos de confianza con bootstrapping

Como forma de poder tener una comparación más robusta entre los distintos modelos de NER entrenados, aplicamos la técnica de bootstrapping, introducida en la sección 2.8, para estimar intervalos de confianza del 95 %.

El algoritmo seguido para aplicar el bootstrapping y estimar los intervalos para cada uno de los modelos fue el siguiente:

1. Tomamos los 207 infomes de la partición de test de SpRadIE[27] y generamos los archivos en formato .ann (ver *Brat Format* en 4.2.2) a partir de las predicciones del modelo.

2. A partir de esta muestra base de 207 predicciones del modelo, generamos 1000 nuevas muestras (con 207 informes cada una) a través del muestreo con reposición (bootstrapping).

3. Para cada una de las mil muestras, ejecutamos el script de evaluación de SpRadIE, mencionado en la sección anterior. Esto nos da distintos resultados de *f1-score* (tanto exacto como parcial) para cada una de las muestras.

4. Sobre estas métricas calculadas, tomamos la mediana como estimador del valor esperado del *f1-score* para el modelo. Por otro lado, nos quedamos con el 2.5-percentil como estimador del límite inferior del intervalo, y el 97.5-percentil como estimador del límite superior.

Una ventaja de construir los intervalos de esta manera, a partir de los percentiles, es que no dependemos de asumir hipótesis de normalidad sobre los datos muestreados.

4.5.2. Estudio de ablación

Un estudio de ablación es un enfoque sistemático para evaluar el impacto de los componentes individuales dentro de un modelo de Machine Learning. Al eliminar o reemplazar selectivamente ciertas partes de la arquitectura, es posible cuantificar sus contribuciones al rendimiento general. Esta técnica es particularmente útil para comprender el papel de submódulos o arquitecturas complejas en la obtención de los resultados reportados.

[26] https://github.com/francolq/spradie

[27] Es importante aclarar que la partición de test fue usada únicamente en la ejecución final como forma de generar los datos a reportar. En lo restante del desarrollo, se trabajó siempre usando los conjuntos de devSameSample y devHeldOut.

En nuestro caso, todos los modelos de NER entrenados pueden separarse en dos secciones principales: una "sección de embedding", que se encarga de generar una representación interna a partir del texto analizado; y una "sección de etiquetado" (o *sequence tagger*), que es la encargada de interpretar estas representaciones y generar las etiquetas correspondientes. En el caso de los modelos basados en FastText o Flair esta diferencia es más notoria: ambos métodos se enfocan en generar *embeddings* directamente reutilizables por otras arquitecturas, como por ejemplo las BiLSTM+CRF (*sequence tagger*) que escogimos aquí. Y en general cuando se entrena la sección de etiquetado no se modifican los pesos de la sección de embedding. Por su parte, en los Transformers esta distinción puede ser menos evidente: la capa de salida del Transformer (*head*) podría considerarse el *sequence tagger*, mientras que el resto de capas ocultas serían las encargadas de armar el embedding. Sin embargo, cuando se entrena para la tarea de NER generalmente también se actualizan los pesos de varias o todas las capas internas.

Estas diferencias que se dan en la sección de etiquetado entre los distintos modelos entrenados en este trabajo, hace que una comparación directa del aporte de los *embeddings* en cada caso sea complicada. Aún más, a veces estos *sequence tagger* son tan poderosos en sí mismos que logran aprender nuevas relaciones complejas entre los *inputs* aun cuando estas no hayan sido capturadas en los *embeddings*. Por ejemplo, los BiLSTM son una arquitectura que se caracteriza justamente por su capacidad de retener dependencias de largo plazo en si misma, lo que puede suplir carencias que tienen los *embeddings* estáticos, por ejemplo FastText. En el caso de los Transformers, si bien la capa de salida no es particularmente compleja, el hecho de que se estén reentrenando todas las capas del modelo de lenguaje al momento de aprender a etiquetar, genera un efecto similar.

Debido a que el propósito de esta tesis es evaluar a los *embeddings* y modelos de lenguaje no solo desde el punto de vista de la utilidad para una tarea puntual, sino también desde sus propiedades inherentes, optamos por realizar un estudio de ablación en el que reemplazamos los complejos *sequence taggers* de todos los modelos por una capa de etiquetado lineal más un CRF. Dicha capa puede proyectar los *embeddings* al espacio de salida deseado, pero no tiene posibilidad de capturar propiedades semánticas o retener dependencias de largo plazo adicionales. Para este experiemento nos basamos en un análisis planteado en el artículo fundacional de Flair [2].

4.5.3. Algoritmo para abreviaturas

Un hallazgo que fue importante tras las pruebas iniciales con la partición de Validation, es el bajo desempeño que lograban todos los modelos sobre la categoría *Abbreviations*, especialmente en términos de *recall*. En todos los casos se obtuvieron valores por debajo de 0.25 para *recall* al evaluar sobre la partición de validación. Al analizarlo, encontramos que es un fenómeno consistente con la metodología de resolución de anotaciones solapadas que describimos en la sección 4.2.2: las abreviaturas son la entidad más penalizada por este criterio debido a que la mayoría de las veces son partes de entidades más grandes, como *Measure* o *Anatomical_Entity*, y terminan quedando muy subrepresentadas en el conjunto de entrenamiento en formato BIO. Por ejemplo, algunas de las Abbreviations más comúnes son "cm", "mm", "RD" y "RI". Las dos primeras suelen aparecer dentro de mediciones, mientras que las últimas dos son entidades anatómicas (riñón derecho y riñón izquierdo, respectivamente). Adicionalmente, como puede observarse en la figura 4.1, la entidad *Abbreviation* es la segunda en frecuencia después de *Anatomical_Entity*, representando el 19.4 % del total de las etiquetas. Esto provoca que sus resultados tengan

un alto impacto en el *micro-average* de las métricas, como se explica en la sección 4.5.

Debido a estos motivos y al hecho de que los mejores resultados de la competencia original fueron obtenidos por aquellos que decidían usar una estrategia de segmentar entidades para entrenar modelos especialistas, optamos por excluir la etiqueta de *Abbreviation* del entrenamiento de los modelos. En su lugar, para este tipo de entidad utilizamos un algoritmo simple basado en expresiones regulares (*RegEx*) que se explica a continuación.

Primero analizamos la frecuencia de ocurrencia de las distintas palabras marcadas como Abbreviation en la partición de *Train* de `spradie-corpus`. La figura 4.6 muestra la distribución tipo *long-tail* que presenta esta entidad: es decir que pocas palabras representan la gran mayoría de las ocurrencias, mientras que hay muchas palabras con muy pocas ocurrencias. Esto significa que simplemente quedándonos con un pequeño grupo de las palabras más frecuentes podemos cubrir la mayoría de anotaciones. A partir de esta observación armamos una lista de las 24 palabras más frecuentes para Abbreviation, que puede encontrar en el Apéndice (7.2).

Fig. 4.6: Gráfico de barras mostrando la frecuencia de las veinte palabras más frecuentes etiquetadas como Abbreviation en el conjunto de entrenamiento de `spradie-corpus`

Con dicha lista se aplica un algoritmo simple para anotar las palabras: buscamos en cada informe ocurrencias de palabras en la lista, siempre y cuando no estén precedidas o seguidas directamente por otras letras. Con este último criterio lo que hacemos es prevenir Falsos Positivos por anotar sub-palabras erroneamente. Por ejemplo, la Abbreviation "izq" puede ser encontrada dentro de la palabra "izquierda" que no es una Abbreviation.

Vale la pena destacar que como el principal objetivo de la tesis es identificar la utilidad y aporte de las representaciones vectoriales, al momento de realizar las evaluaciones nos enfocamos en los modelos de Machine Learning, dejando de lado la categoría de Abbreviation para estos análisis. En el único caso donde realmente nos importan y las usamos en combinación con los resultados de los modelos, es en la sección 5.3.2 cuando hacemos el *benchmark* contra el estado del arte.

4.6. Evaluación intrínseca

Como se comenta en el capitulo 3 la escasez de *gold standards* para el dominio de análisis de ecografías o afines en idioma español dificulta la evaluación intrínseca, al menos de

una forma fácil de cuantificar y comparar. Por este motivo decidimos implementar esta evaluación intrínseca a través de un análisis cualitativo que nos permita comprender un poco mejor cuán bien nuestros *embeddings* capturan la semántica del dominio, independientemente de una tarea objetivo específica. Debido a que este sistema requiere un gran esfuerzo de análisis manual, optamos por no hacerlo para todos los modelos sino solo para los que nos dieron los mejores resultados en las pruebas extrínsecas.

El marco general consiste en realizar tres pasos: generar los *embeddings* para cada uno de los informes, aplicar una reducción de dimensionalidad y por último un análisis de los *clusters* formados en este espacio reducido. A continuación presentamos los métodos aplicados para dicho análisis.

Como mencionamos arriba, los modelos usados para este análisis son solo algunos basados en Transformers, y más específicamente basados en la arquitectura de BERT. Para obtener los *embeddings* seguimos el procedimiento explicado en la sección 2.4.2. En nuestro caso, por simplicidad y dado que hasta donde conocemos no hay ningún estudio que generalice la mejor forma de usar los estados internos para *embeddings*, decidimos proceder únicamente con la última capa oculta.

Esta metodología la aplicamos sobre una partición de Test con 8222 informes separados de `anonymized-ultrasounds-80k-with-diacritics` antes de entrenar. Escogemos realizar los *embeddings* sobre los informes completos, promediando los *embeddings* obtenidos para cada *token* que compone un informe. Otro tipo de análisis que podría resultar interesante es el de hacer los *embeddings* a nivel de oraciones.

Debido a que los *embeddings* tienen 768 dimensiones resulta complejo analizarlos. Por tal motivo aplicamos técnicas de reducción de dimensionalidad para llevarlos a un espacio más manejable. Específicamente usamos *Principal Component Analysis* (PCA) y *t-Distributed Stochastic Neighbor Embedding* (t-SNE), que fueron introducidas en 2.9. El foco lo ponemos sobre t-SNE ya que nuestro principal interés está en poder hacer un análisis gráfico en 2-dimensiones. Sin embargo también realizamos algunas pruebas con PCA y PCA + t-SNE (reduciendo las dimensiones de los *embeddings* de 768 a 50 mediante PCA y luego aplicando t-SNE sobre esto).

Una vez reducida la dimensionalidad de los *embeddings* pasamos a aplicar algunas técnicas de clustering para tratar de simplificar el análisis de los *clusters* que se forman. Los métodos usados son K-Means, DBSCAN y HDBSCAN, todos introducidos en 2.10. Vale aclarar que acá no estamos tratando de optimizar nada, solo ayudarnos de estos métodos para poder segmentar de forma semi-automática algunas áreas interesantes para analizar en el espacio reducido.

El objetivo es ver si se forman *clusters* y si los mismos son consistentes con la semántica de los informes representados por los puntos de dichos *clusters*.

5. RESULTADOS DE EVALUACIÓN EXTRÍNSECA

En este capítulo analizamos los resultados obtenidos para la tarea objetivo (SpRadIE) utilizando los distintos modelos de lenguaje y *embeddings* mencionados en el capítulo anterior. Todos los resultados que se presentan a continuación fueron calculados sobre la partición de Test de la tarea de SpRadIE. Para el cómputo de las métricas se ejecuto exactamente el mismo script utilizado en dicha competencia. En todos los caso, las métricas de Precision, Recall y F1-Score fueron calculadas con *micro-average* sobre todas las entidades analizadas.

Este capítulo se divide en cuatro secciones principales: primero analizamos los resultados generales de los modelos entrenados apoyándonos en la técnica de *bootstrapping*; luego hacemos lo propio con el estudio de ablación propuesto; continuamos con una comparación de nuestros resultados contra el estado del arte; y por último, cerramos haciendo un análisis manual de los errores que nos permite ganar un mayor entendimiento sobre el comportamiento del modelo.

5.1. Análisis usando Bootstrapping

En esta sección presentamos los resultados obtenidos para todos los modelos entrenados al evaluarlos sobre la partición de Test. Para poder sacar conclusiones estadísticamente significativas, presentamos las medianas del F1-Score tanto exacto (Exact F1-Score) como parcial (Lenient F1-Score) en conjunto con sus intervalos de *al menos* 95 % de confianza[1]. Dichos resultados se muestran en la tabla 5.1, agrupados según el tipo de modelo de lenguaje pre-entrenado que se usó. También diferenciamos los resultados obtenidos por aquellos clasificadores que se construyeron directamente a partir de estos modelos ("Base") de aquellos que lo hicieron sobre los modelos con *fine-tuning* sobre nuestros datos ("Finetuned"). En el caso particular de los clasificadores basados en FastText, dado que no son modelos de lenguaje ni tampoco se usaron pesos pre-entrenados, nos referimos a estos como "Trained", como forma de destacar que fueron completamente entrenados por nosotros y sobre nuestros datos.

[1] Si bien fueron estimados intervalos de confianza del 95 % usando *bootstrapping*, los mismos no tienen porque ser simétricos respecto de la mediana ya que no asumimos una distribución particular sobre las muestras. Sin embargo en la práctica vemos que en la mayoría de los casos resultan simétricos o casi simétricos, por lo que decidimos escoger siempre tomar un intervalo simétrico respecto de la media, para poder simplificar la notación y usar el símbolo $\pm$. Por ejemplo, si la mediana fuera 80, el límite inferior del intervalo 79,5 ($-0{,}5$) y el superior 81 ($+1$), entonces registramos 80 ± 1. Esta metodología asegura que el intervalo de confianza del 95 % estimado con el método de bootstrapping siempre esté incluido en el intervalo presentado. Y por este motivo hablamos de que los intervalos son de "al menos 95 %".

Modelo	**Lenient F1-Score**	**Exact F1-Score**
BETO CLINICAL WL		
Base	83.8 ± 1.2	78.5 ± 1.6
Finetuned	**85.3 ± 1.2**	**80.1 ± 1.5**
ROBERTA BIO CLINICAL		
Base	83.8 ± 1.3	78.9 ± 1.6
Finetuned	84.9 ± 1.1	79.5 ± 1.5
BETO CASED		
Base	84.2 ± 1.2	79.0 ± 1.6
Finetuned	84.9 ± 1.2	79.9 ± 1.5
BETO UNCASED		
Base	83.8 ± 1.2	78.6 ± 1.4
Finetuned	84.6 ± 1.2	79.7 ± 1.4
ROBERTA CLINICAL WL		
Base	84.4 ± 1.2	78.8 ± 1.6
Finetuned	84.5 ± 1.3	79.1 ± 1.5
ROBERTA BNE		
Base	82.5 ± 1.3	77.0 ± 1.6
Finetuned	84.1 ± 1.3	79.0 ± 1.5
CLINICAL FLAIR		
Base	82.4 ± 1.4	76.6 ± 1.6
Finetuned	83.7 ± 1.2	77.9 ± 1.5
FASTTEXT200		
Trained	81.5 ± 1.4	74.8 ± 1.7
FASTTEXT100		
Trained	80.5 ± 1.4	74.6 ± 1.5

Tab. 5.1: Mediana de los F1-Scores (*micro-averaged*) estimando intervalos de al menos 95 % de confianza mediante *bootstrapping*, tanto para *match* exacto como parcial. Los modelos están ordenados de mayor a menor Lenient F1-score del modelo finetuneado.

Como primer observación viendo los resultados en la tabla 5.1 podemos notar que los modelos que mejor se desempeñaron fueron todos aquellos basados en Transformers, seguidos de los modelos basados en Flair (BiLSTM) y por último aquellos basados en FASTTEXT. Entre el mejor y el peor modelo hay una diferencia de 5 puntos porcentuales (pp) para *match* parcial y de 5.5 pp para *match* exacto.

Todos los modelos presentan mejores resultados para el *match* parcial que para el exacto. Si bien esto no sorprende, es destacable que en todos los casos (independientemente del tipo de modelo y del tratamiento que recibió respecto al *fine-tuning*) la diferencia entre el resultado exacto y parcial está entre 5 y 6 pp. Una hipótesis del porqué esta diferencia se mantiene casi constante entre todos los modelos podría ser que exista un sub-conjunto de informes para el cuál resulta particularmente desafiante lograr un *match* exacto. Esta es una hipótesis que no profundizamos en este trabajo.

Un punto a destacar es que, para los casos de los modelos de lenguaje donde se realizó

fine-tuning sobre el corpus `anonymized-ultrasounds-80k`, los resultados son sostenidamente mejores en comparación a los casos donde se utilizó directamente el modelo base, aunque no por demasiado. En algunos casos la diferencia es incluso casi nula (por ejemplo, en ROBERTA CLINICAL WL y en BETO CASED) sobretodo si se consideran los intervalos de confianza.

Dentro de lo que son los modelos basados en Transformers, no vemos diferencias demasiado grandes entre los modelos finetuneados. No hay en principio una dominancia clara entre los modelos basados en arquitecturas del tipo BERT, de aquellos basados en arquitecturas del tipo RoBERTa.

Si bien en algún punto es alentador ver que por lo menos el proceso de *domain adaptation* sobre el modelo de lenguaje mantiene o mejora los resultados, *a priori* esperábamos ver diferencias más significativas. Recordemos que aquí técnicamente se están haciendo dos *fine-tunings* por cada modelo: el primero es un *domain adaptation* no-supervisado a nivel del modelo de lenguaje (o *embedding*) y el segundo es una *task adaptation* para la tarea de NER específica. Nuestro objetivo principal es medir la ganancia generada por el primer proceso y en ese sentido el segundo proceso puede generar cierta distorsión. Por eso realizar un estudio de ablación resulta de gran ayuda para entender mejor esto.

5.2. Estudio de ablación

En esta sección se presentan los resultados del estudio de ablación, que como se explicó previamente en 4.5.2 consiste en utilizar una capa lineal combinada con *Conditional Random Fields* como forma de clasificar las entidades del NER. Al hacer esta simplificación en la capa de clasificación podemos apreciar mejor la contribución de los *embeddings* entrenados al resultado final.

En la tabla 5.2 podemos ver el resultado de dicho estudio. El formato de esta tabla es similar al de la tabla 5.1 con un agregado: para cada métrica se pone entre paréntesis de cuántos puntos porcentuales fue la caída en el desempeño respecto de los resultados mostrados en la sección anterior. Para simplificar un poco la cantidad de información a mostrar en la tabla se omitieron los resultados basados en los modelos BETO CASED, BETO UNCASED y FASTTEXT100 por no presentar características adicionales a las que mencionamos a continuación.

El primer punto que se observa es que los modelos basados en FASTTEXT (no contextuales) son los que sufren la mayor caída de *performance* al aplicar la ablación: la caída está en torno a los 24 puntos porcentuales (pp) para el F1-Score con *match* parcial y 27 pp para el *match* exacto. Mientras que para las representaciones contextualizadas, la caída máxima está en alrededor de los 10 pp, con la única excepción de ROBERTA BNE, donde la caída es mayor (entre 11 y 14 pp) pero sigue lejos de la de FASTTEXT. Es destacable que estos resultados son muy similares a los que se presentan en el *paper* fundacional de Flair [2] de donde tomamos inspiración para realizar este análisis. Este caso también pone en evidencia el peso que tiene el uso de un algoritmo de NER que se apalanque en la información contextual por sí mismo, mostrando que al menos para el caso de SpRadIE puede compensar el uso de representaciones no tan potentes.

Yendo al caso de los modelos contextualizados hay varios aspectos interesantes para notar. La primera de todas es que al realizar la ablación de los modelos el orden del ranking que se ve en la tabla 5.1 cambia. BETO CLINICAL WL con *finetuning*, que había presentado los mejores resultados, pasa a quedar en un tercer lugar al cambiar su

Modelo	**F1-Score Parcial**	**F1-Score Exacta**
BETO CLINICAL WL		
Base + Ablation	77.8 ± 1.5 (↓ 6 pp)	71.2 ± 1.9 (↓ 7.3 pp)
Finetuned + Ablation	78.6 ± 1.5 (↓ 6.7 pp)	71.7 ± 2.0 (↓ 8.4 pp)
ROBERTA BIO CLINICAL		
Base + Ablation	78.2 ± 1.5 (↓ 5.6 pp)	71.3 ± 1.7 (↓ 7.6 pp)
Finetuned + Ablation	76.1 ± 1.7 (↓ 8.8 pp)	69.1 ± 2.0 (↓ 10.4 pp)
ROBERTA BNE		
Base + Ablation	69.1 ± 1.8 (↓ 13.4 pp)	63.2 ± 2.0 (↓ 13.8 pp)
Finetuned + Ablation	73.2 ± 1.6 (↓ 10.9 pp)	65.8 ± 1.8 (↓ 13.2 pp)
ROBERTA CLINICAL WL		
Base + Ablation	77.8 ± 1.6 (↓ 6.6 pp)	69.7 ± 1.8 (↓ 9.1 pp)
Finetuned + Ablation	79.6 ± 1.4 (↓ 4.9 pp)	73.0 ± 1.7 (↓ 6.1 pp)
CLINICAL FLAIR		
Base + Ablation	73.6 ± 2.1 (↓ 8.8 pp)	68.4 ± 2.1 (↓ 8.2 pp)
Finetuned + Ablation	**82.0 ± 1.3** (↓ 1.7 pp)	**76.1± 1.5** (↓ 1.8 pp)
FASTTEXT200		
Trained + Ablation	57.2 ± 2.6 (↓ 24.3 pp)	47.6 ± 2.8 (↓ 27.2 pp)

Tab. 5.2: Mediana de los *micro-averaged* F1-Scores (tanto para *match* parcial como para *match* exacto) estimando intervalos de al menos 95 % de confianza mediante *bootstrapping*. Además, entre paréntesis se presenta la caída de *performance* medida en puntos porcentuales (pp) con respecto a los resultados presentados en la tabla 5.1 para mismo modelo pero sin aplicar la ablación. Dichas caídas deben considerarse en forma aproximada debido a que los márgenes de error de las mediana no se están teniendo en cuenta para este cálculo. Por simplicidad y claridad los resultados para BETO CASED, BETO UNCASED y FASTTEXT100 fueron omitidos aquí.

última capa, quedando por detrás de CLINICAL FLAIR y ROBERTA CLINICAL WL respectivamente (ambos con *finetuning*). Es de destacar el caso de CLINICAL FLAIR que previamente había mostrado el peor resultado para los modelos contextualizados (si bien fue por una diferencia ajustada) y tras la ablación queda en primer lugar con cierta diferencia (alrededor de 2 pp respecto del segundo mejor y 3 pp respecto del tercero).

Otro punto que resulta relevante analizar es el impacto de la presencia de *fine-tuning* o no en los modelos. El caso de CLINICAL FLAIR es de los más llamativos ya que si comparamos las pérdidas de desempeño entre el modelo base y el finetuneado, vemos que el primero tiene una caída significativa (cercana a 9 pp) mientras que el segundo recibe un impacto mucho más acotado (por debajo de 2 pp). Esto parece evidenciar que en este caso el proceso de *domain adaptation* es exitoso en capturar información interna de la estructura de los informes y generar representaciones que simplifican la tarea de reconocer las entidades aún con un algoritmo lineal. También suma un punto más al argumento de que el uso de modelos complejos (como Redes Neuronales Recurrentes en este caso) para la capa de reconocimiento de entidades nombradas opaca en buena medida la diferencia de calidad de los *embeddings* que se usen como entrada. Al respecto de esto basta notar en la tabla 5.1, de la sección anterior, que la diferencia entre el modelo base y el finetuneado era menor a 2 pp sin aplicar la ablación.

Volviendo al análisis de la relevancia del *fine-tuning* de los modelos de lenguaje, si ahora nos enfocamos en los modelos basados en Transformers observamos un comportamiento mixto. BETO CLINICAL WL, ROBERTA CLINICAL WL y ROBERTA BNE performan un poco mejor o igual en sus versiones finetuneadas respecto de las base. Sin embargo, ROBERTA BIO CLINICAL por algún motivo performa mejor en su versión base (a pesar de que en el modelo sin ablación se desempeñaba mejor la versión finetuneada) por una diferencia aproximada de 2 pp.

A modo de comentario de cierre de esta sección, es importante señalar que este estudio no es suficiente por si solo para determinar que un *embedding* es mejor que otro. Al estar poniendo un clasificador lineal en la capa de salida, básicamente estamos condicionando que las representaciones usadas pertenezcan a un espacio donde las anotaciones de las entidades sean linealmente separables. Si bien la separabilidad lineal es una propiedad generalmente deseable, no necesariamente eso quita que *embeddings* que no la cumplan puedan resultar más propicios para otro tipo de capas de salida. También hay que tener en cuenta que un mismo *embedding* puede ser reaprovechado para distintos problemas donde no necesariamente la representación cumpla separabilidad lineal. Para profundizar en esto se aconseja la lectura de la sección 15.1.1 del libro "Deep Learning" [27].

5.3. Comparación de resultados respecto de trabajos previos

En lo que sigue se realiza un análisis de los resultados obtenidos comparándolos directamente contra los mejores resultados publicados en la competencia de SpRadIE. Un resumen de los resultados alcanzados por cada equipo participante puede encontrarse en "Overview of CLEF eHealth Task 1-SpRadIE: A challenge on information extraction from Spanish Radiology Reports." [17]. Las métricas son calculadas siguiendo exactamente el mismo procedimiento que se uso en la competencia (ver sección 4.5) de forma tal que pueda realizarse efectivamente la comparación. Este es el motivo por el que los resultados presentados en esta sub-sección no incluyen un intervalo de confianza estimado, ya que aquí se considera únicamente la métrica computada sobre la totalidad de la partición de

Test.

Para esto, además es necesario incluir predicciones para las entidades de tipo *Abbreviation* que hasta ahora excluimos de los resultados presentados pero que son parte de los resultados obtenidos por los participantes de la competencia. En la primera parte de esta sección explicamos cómo se realizan las predicciones para esta entidad específica. Luego, presentamos los resultados finales combinando los resultados obtenidos para las 10 entidades etiquetadas en el *corpus*.

5.3.1. Etiqueta Abbreviation

En la tabla 5.3 presentamos los resultados obtenidos a través del algoritmo desarrollado para detectar abreviaturas. Este algoritmo fue introducido en la sección 4.5.3. Una observación interesante es que por el funcionamiento del mismo, que identifica patrones exactos en el texto, las métricas dan prácticamente el mismo resultado para *match* parcial y para *match* exacto . Por ese motivo, informamos únicamente el valor obtenido aplicando *match* exacto.

Métrica	**Valor**
Precision	97.6
Recall	88.1
F1	92.6

Tab. 5.3: Métricas de Precision, Recall y F1-Score (todas *micro-averaged*) con *match* exacto obtenidas para la identificación de entidades de tipo *Abbreviation*, usando el algoritmo basado en Regex.

Estos resultados son mejores que el promedio obtenido para el resto de las entidades, sacando una diferencia favorable en el orden de 10 puntos porcentuales (si comparamos métrica a métrica con los resultados vistos en la tabla 5.1, para cualquiera de los modelos que se tome como referencia). A continuación, en la sección 5.3.2, se reportan los resultados sobre el conjunto de Test combinando tanto este método (aplicado únicamente para detectar Abbreviation) como los basados en modelos de lenguaje (que se aplican para detectar el resto de las entidades descriptas en 4.1.2). De este modo, se logra un resultado comparable con los reportados en la competencia de SpRadIE [17].

5.3.2. Comparación con la competencia

Ahora presentamos los resultados obtenidos al realizar la evaluación de los modelos de reconocimiento de entidades nombradas, sobre el conjunto de Test entero y combinándolos con los resultados presentados en la sección 5.3.1 para las abreviaciones. En las tablas 5.4 y 5.5 se vuelcan los resultados obtenidos para *match* parcial y exacto respectivamente. Además se incluyen los resultados de los modelos que, hasta el momento de la publicación de este trabajo, eran los mejores publicados para la tarea de SpRadIE. Recordemos que el criterio que se escogió para seleccionar al mejor modelo en la competencia fue aquel que alcanzara mayor F1-Score (*micro-averaged*) con *match* parcial sobre todas las entidades descriptas en la sección 4.1.2.

Model	Precision	Recall	F1 Score
FT BETO CLINICAL WL$_{+Regex}$	**91.2**	83.1	**86.9**
FT BETO CASED$_{+Regex}$	91.2	82.5	86.6
FT ROBERTA BIO CLINICAL$_{+Regex}$	91.1	82.6	86.6
FT BETO UNCASED$_{+Regex}$	90.5	82.5	86.3
FT ROBERTA CLINICAL WL$_{+Regex}$	90.6	82.4	86.3
ROBERTA CLINICAL WL$_{+Regex}$	90.6	82.2	86.2
BETO CASED$_{+Regex}$	90.1	82.3	86.0
FT ROBERTA BNE$_{+Regex}$	90.7	81.7	86.0
ROBERTA BIO CLINICAL$_{+Regex}$	89.9	82.0	85.8
BETO UNCASED$_{+Regex}$	89.7	82.1	85.8
BETO CLINICAL WL$_{+Regex}$	89.2	82.5	85.7
EdIE (UK) – run2	87.2	**83.9**	85.5
ROBERTA BNE$_{+Regex}$	89.0	81.0	84.8
LSI (Spain) – run1	90.3	78.3	83.9
FT FASTTEXT200$_{+Regex}$	89.1	79.3	83.9
FT CLINICAL FLAIR$_{+Regex}$	88.9	79.1	83.7
FT FASTTEXT100$_{+Regex}$	89.0	78.2	83.2
BASE CLINICAL FLAIR$_{+Regex}$	87.2	78.1	82.4

Tab. 5.4: Métricas de Precision, Recall y F1-Score (todas *micro-averaged*) con *match* parcial, calculadas bajo el mismo procedimiento que se utilizó en la competencia de SpRadIE. En esta tabla se combinan los resultados obtenidos para Abbreviations (usando Regex) con los resultados para las restantes 9 entidades (usando Deep Learning). A fines de comparación se incluyen también los dos mejores resultados de la competencia (resaltados en negrita). Los modelos están ordenados por *micro-averaged* F1-Score de forma descendente y en negrita se resaltan los mejores resultados para cada métrica. La sigla FT (por *fine-tuned*) al comienzo del nombre indica que dicho modelo de NER se basa en un modelo de lenguaje o *embedding* fine-tuneado por nosotros. Caso contrario, parte de uno de los modelos base señalados en la sección 4.3.

Model	Precision	Recall	F1 Score
FT BETO CLINICAL WL$_{+Regex}$	86.9	**79.2**	**82.9**
FT BETO CASED$_{+Regex}$	**87.1**	78.8	82.7
FT BETO UNCASED$_{+Regex}$	86.5	78.9	82.5
FT ROBERTA BIO CLINICAL$_{+Regex}$	86.6	78.5	82.3
FT ROBERTA CLINICAL WL$_{+Regex}$	86.2	78.4	82.1
BETO CASED$_{+Regex}$	85.8	78.4	82.0
FT ROBERTA BNE$_{+Regex}$	86.5	77.9	82.0
ROBERTA BIO CLINICAL$_{+Regex}$	85.8	78.3	81.9
ROBERTA CLINICAL WL$_{+Regex}$	86.0	78.0	81.8
BETO UNCASED$_{+Regex}$	85.4	78.2	81.6
BETO CLINICAL WL$_{+Regex}$	84.9	78.5	81.6
ROBERTA BNE$_{+Regex}$	84.4	76.8	80.4
EdIE (UK) – run2	81.9	78.7	80.3
LSI (Spain) – run1	86.2	74.8	80.1
FT FASTTEXT200$_{+Regex}$	83.5	74.3	78.7
FT CLINICAL FLAIR$_{+Regex}$	82.8	73.7	78.0
FT FASTTEXT100$_{+Regex}$	84.0	73.8	78.6
BASE CLINICAL FLAIR$_{+Regex}$	81.1	72.6	76.6

Tab. 5.5: Métricas de Precision, Recall y F1-Score (todas *micro-averaged*) con *match* exacto, calculadas bajo el mismo procedimiento que se utilizó en la competencia de SpRadIE. En esta tabla se combinan los resultados obtenidos para Abbreviations (usando Regex) con los resultados para las restantes 9 entidades (usando Deep Learning). A fines de comparación se incluyen también los dos mejores resultados de la competencia (resaltados en negrita). Los modelos están ordenados por *micro-averaged* F1-Score de forma descendente y en negrita se resaltan los mejores resultados para cada métrica. La sigla FT (por *fine-tuned*) al comienzo del nombre indica que dicho modelo de NER se basa en un modelo de lenguaje o *embedding* fine-tuneado por nosotros. Caso contrario, parte de uno de los modelos base señalados en la sección 4.3.

Observamos que no solo obtuvimos uno sino varios modelos que performaron mejor que los mejores resultados publicados hasta ahora, tanto para *match* parcial como exacto. Es justo mencionar que aún así los resultados son muy ajustados para *match* parcial: 1.4 pp separan a nuestro mejor modelo del de EdIE, lo que representa una mejora de un 1.6 %. La diferencia es un poco más significativa en el caso de *match* exacto: el nuevo modelo saca 2.6 pp de ventaja, representando una mejora del 3.2 %.

Quizás lo más destacable es que se logró obtener resultados comparables e incluso mejores con una solución significativamente menos compleja que la implementada por EdIE (una descripción de la misma se encuentra en la sección 3.2). Ellos realizan algunos pre-procesamientos al *dataset* y algunas correcciones de etiquetas, además de usar algunos métodos más sofisticados para tratar de representar mejor las entidades anidadas. También hacen uso de un BETO separado por cada tipo de entidad para resolver palabras con anotaciones de distintos tipos y lidiar mejor con el desbalanceo, además de un método adicional basado en análisis de frecuencia de patrones. Probablemente fue gracias a estos últimos esfuerzos que consiguieron obtener el mejor Recall para *match* parcial. Por nuestro lado ningún preprocesamiento de los datos fue realizado más allá de la conversión de formato, y la resolución de solapamientos y discontinuidades se hizo en forma *naïve*. Se

entrenó un único modelo para todas las entidades con excepción de Abbreviation, y para esta última se realizó un método simple, basado en *RegEx* para encontrar los patrones más frecuentes.

En la tabla 5.6 presentamos con mayor detalle las métricas obtenidas por nuestro mejor modelo, FT BETO CLINICAL WL, para cada tipo de entidad nombrada por separado. Además mostramos el soporte de cada entidad, que es la cantidad total de etiquetas que existen en el corpus para dicha entidad. Esto nos permite dimensionar mejor el impacto que tiene el desempeño del modelo por entidad en el resultado conjunto de cada métrica (que al ser *micro-averaged* le da más peso a las entidades con mayor soporte, como se explica en la sección 4.5). Una apreciación que podemos hacer es que la diferencia favorable conseguida en los resultados globales, respecto de los mejores de la competencia, no se explica por la forma particular en la que tratamos a las Abbreviations. El equipo ganador de SpRadIE tuvo incluso mejores resultados en esta categoría: nuestro F1-Score fue de 92.8 para Abbreviations mientras que el del equipo EdIE fue de 95.4. Esto es destacable ya que muestra que el diferencial positivo que conseguimos se genera en el resto de las categorías, que fue donde centramos nuestros esfuerzos con los modelos de *Deep Learning* . Además es la segunda categoría con más peso (observando el soporte) y en la que menos esfuerzo invertimos, por lo que es razonable pensar que podemos mejorarla (quizás simplemente agregando más patrones a nuestro sistema de expresiones regulares) e incrementar aún más nuestro F1-Score.

Tipo de entidad	**Parcial**			**Exacto**			**Soporte**
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**	
Anatomical Entity	92.9	83.4	87.9	86.9	78.0	82.2	1772
Abbreviation	97.8	88.3	92.8	97.6	88.1	92.6	1652
Measure	93.5	82.2	87.5	90.1	79.3	84.3	1258
Finding	80.4	79.5	80.0	70.7	69.9	70.3	1000
Location	77.4	65.2	70.8	72.2	60.8	66.1	646
Type of measure	96.0	89.1	92.4	91.6	85.0	88.2	558
Negation	94.7	95.9	95.3	93.5	94.7	94.1	453
Degree	85.2	67.5	75.4	83.1	65.9	73.5	82
Uncertainty	86.4	67.5	75.8	84.2	65.8	73.8	73
Conditional Temporal	45.5	62.5	52.6	45.5	62.5	52.6	8

Tab. 5.6: Métricas de Precision, Recall y F1-Score (todas *micro-averaged* y computadas tanto con *match* exacto como parcial) para el modelo FT BETO CLINICAL WL, desagregadas por tipo de entidad nombrada. Para cada entidad, además, se muestra el soporte.

Otra nota interesante, al compararnos con el equipo de EdIE, es que en su trabajo ([61]) ellos mencionan que no lograron predecir ninguno de las ocho fragmentos anotados como `Conditional Temporal`, obteniendo un F1-Score de 0 para esta categoría. Mientras que nosotros logramos reconocer el 62.5 % con una precisión del 45.5 % (promediando un F1 de 52.6 %). Si bien este resultado es anecdótico en el contexto de la competencia (ya que al ser solo 8 casos no representan ni siquiera el 0.1 % de la métrica) se puede considerar como un resultado alentador en cuanto a la capacidad de nuestro modelo y un indicio de que el proceso de *transfer learning* favorece el aprendizaje de categorias altamente sub-representadas.

5.3.3. Análisis manual de los errores

Para entender un poco mejor los resultados de FT BETO CLINICAL WL (nuestro mejor modelo) y analizar sus oportunidades de mejora, procedimos con un análisis manual de algunos de los errores que cometió el modelo al realizar predicciones sobre el conjunto de Test. Escogemos dicho modelo dado que fue el que mejores resultados presentó para la métrica de F1-Score (tanto en esta sección como en el análisis usando bootstrapping de la sección 5.1). Para este análisis excluimos la entidad Abbreviation, ya que únicamente nos interesa buscar oportunidades para mejorar el modelo de lenguaje o el modelo de NER.

Primero repasamos algunos detalles de notación. Usamos el término "predicción" para referirnos a las etiquetas sugeridas por el modelo y "anotación" para referirnos a las etiquetas provistas en SpRadIE. Recordemos que cuando hablamos de error aquí nos referimos a los casos donde el índice de Jaccard (explicado en 4.5) entre la predicción y la anotación es estrictamente menor a 1 (es decir que no son exactamente iguales). En particular, hablaremos de "error parcial" si el índice es distinto de 0 y de "error completo" si el índice es exactamente 0. Ahora sí, pasamos a repasar los principales hallazgos.

En la tabla 5.7 se muestran la cantidad y tipo de errores cometidos por el modelo FT BETO CLINICAL WL para cada uno de los tipos de entidad nombrada. Podemos observar que las dos entidades con mayor cantidad de errores son Location (30) y Degree (23). En cambio Anatomical Entity es la entidad que menos errores tiene (solo un error parcial). Analizamos un poco más a fondo los casos de Location y Degree para entender a que se deben estos errores.

Entidad	Error Parcial	Error Completo	**Total**
Location	8	22	**30**
Degree	2	21	**23**
Uncertainty	3	10	**13**
Type of Measure	3	10	*13*
Measure	7	5	**12**
Finding	6	5	**11**
Negation	1	5	**6**
Conditional Temporal	0	5	**5**
Anatomical Entity	1	0	**1**
Total	**31**	**83**	**114**

Tab. 5.7: Cantidad y tipo de errores por entidad para el modelo FT BETO CLINICAL WL evaluado sobre la partición de Test de SpRadIE. Llamamos "error parcial" a aquellos errores donde el índice de Jaccard entre la predicción y la anotación es estrictamente mayor a 0 y estrictamente menor a 1. Los "errores completos" son aquellos donde el índice de Jaccard es exactamente 0. La columna "Total" refleja la suma de los errores parciales y completos para cada tipo de entidad. Las entidades están ordenadas por esta columna, de mayor a menor. Por último, la fila "Total" resume la cantidad de errores según su tipo (parciales y completos) y en total (para todos los tipos y entidades).

En el caso de Location (LOC) analizamos 18 de los 30 errores totales. En la muestra, observamos que el tipo de error predominante son los Falsos Negativos (16 de los 18), es decir ya sea total o parcialmente hay cosas que el modelo no está prediciendo como Location cuando debería. Vemos en varios casos (al menos 5) que el problema se debe a una limitación intrínseca de nuestro modelado: el modelo solo permite asignar una etiqueta

a cada palabra por lo que en los casos donde dos entidades distintas se solapan sobre una palabra, solo una de las dos puede ser asignada. Veamos los siguiente ejemplos:

Ejemplo 5.3.1. Nodulo isoecoco conocido en lobulo izquierdo hepatico de 41x338x30 mm

Ejemplo 5.3.2. Pequenas adenopatias cervicolaterales dispuestas en cadena de ecoestructura conservada.

En el caso del ejemplo 5.3.1 "hepatico" (resaltado en amarillo) fue anotado manualmente como Location aunque también forma parte de una Anatomical Entity (resaltada en rojo). Por otro lado en 5.3.2 "cervicolaterales" (en amarillo) fue marcado también como Location en la anotación, pero a su vez pertenece a un Finding (en rojo). En ambos casos el modelo falla en reconocer las Location pero identifica con éxito la Anatomical Entity y el Finding respectivamente. En este sentido podemos decir que el modelo está haciendo "lo mejor que puede", dado que si identificara adecuadamente la LOC entonces reduciría su desempeño en las otras entidades. Notar que el motivo por el que en estos casos se priorizan otras entidades por sobre las Location viene de la forma en que se resolvieron las entidades solapadas durante el preprocesamiento de la partición de Train: recordemos que si una entidad está anidada dentro de otra más grande entonces solo conservamos la entidad más grande al convertir de Brat a BIO (ver sección 4.2.2), perdiendo la más corta. Por lo tanto es consistente que en estos casos Location, al ser la la entidad más corta, sea la que se falle en reconocer.

Para los casos de arriba es fácil identificar el impacto de la estrategia *naïve* que seguimos para resolver los anidamientos. Sin embargo tenemos la hipótesis de que esto también afecta de una forma indirecta a otros Falsos Negativos de Location. Veamos nuevamente un ejemplo para ilustrar esto:

Ejemplo 5.3.3. Quiste ovarico derecho

En el ejemplo 5.3.3 la anotación indica que "Quiste ovarico derecho" es un Finding (FI) y que "ovarico derecho" es una LOC. Sin embargo el modelo falla en predecir ambas cosas: predice en forma parcialmente correcta "quiste ovarico" como FI, pero "derecho" se anota como Anatomical Entity (AE). Claramente esta última anotación no tiene sentido y sin embargo vemos que este comportamiento se repite en otros casos también: "derecho", "izq", "ambas", "inferiores" son palabras que vemos que el modelo anota en forma individual como AE o FI según el caso. Aquí la hipótesis: si bien estas palabras anotadas como AE o FI por sí solas no tienen sentido, ciertamente son palabras que suelen ser *parte de* AE o FI, y por cómo funciona el mecanismo que se explicó anteriormente para resolver anidamientos, resulta entendible que el modelo asocie estas palabras con AE o FI antes que con LOC. El modelo nunca pudo aprender a reconocer estos *tokens* como LOC durante su entrenamiento ya que, en casos similares, se descartaron las anotaciones de LOC en favor de entidades generalmente más grandes como AE o FI.

Para estos casos donde el problema nace de la existencia de entidades solapadas posiblemente la mejor estrategia sea hacer una segmentación de modelos por tipo de entidad similar a la que hizo el equipo de EdIE. De esta forma se pueden identificar independientemente los distintos tipos de entidades y evitamos estos efectos cruzados que se generan.

El caso de Degree (DE) es prácticamente igual al de LOC en cuanto a los anidamientos se refiere, y posiblemente se ve agravado porque esta entidad representa solo el 1 % de todas

las anotaciones del corpus (ver figura 4.1) por lo que es aún más complejo para el modelo poder aprender sobre esta categoría. Sorprendentemente, a pesar de estos factores el F1-Score es mejor para Degree (75.4) que para LOC (70.8) si vemos la tabla 5.6. *A priori* una teoría de por qué puede pasar esto viene dada por la composición de la categoría: dentro de DE pueden distinguirse principalmente adjetivos (por ejemplo, "mínimo") y adverbios (por ejemplo, "mínimamente"). Al analizar los errores vemos que en casi todos los casos se trata de adjetivos, lo cual tiene sentido ya que es extraño que un adverbio sea parte de una Anatomical Entity o de un Finding, a diferencia de los adjetivos. De hecho, los dos casos de errores que involucran adverbios se tratan de Falsos Positivos, como consecuencia de anotaciones manuales incompletas. Veamos un ejemplo de solapamiento y otro de falso positivo:

Ejemplo 5.3.4. Ambos rinones hidronefroticos, con espesor parenquimatoso minimo .

Ejemplo 5.3.5. BAZO: minimamente aumentado de tamano.

En el ejemplo 5.3.4 la palabra "minimo" (resaltada en amarillo) es anotada como DE, mientras que lo resaltado en rojo es un FI. De esta forma, el solapamiento dificulta la potencial predicción de la entidad Degree. En cambio, en el caso de 5.3.5 lo que sucede es distinto: la predicción indica que "minimamente" es un DE (lo cual tiene sentido) pero no fue incluida en las anotaciones manuales. En este caso se trata de un error de anotación, hablaremos un poco de los mismos a continuación.

Algo que vimos en varias oportunidades tanto en este análisis sobre el conjunto de Test de SpRadIE como en algunos anteriores sobre las otras particiones, es que hay varios errores en la anotación. En algunos casos los errores son evidentes y en otros hay cierta ambigüedad que puede requerir una revisión adicional de los criterios de anotación. Los errores más frecuentes que hemos visto son por omisión: en algunos casos se omite una palabra o entidad completa; en otros la anotación está bien pero falta o sobra algún caracter. Para la muestra de errores analizados del conjunto de Test, se hallaron al menos 8 errores de anotación y 7 más en las particiones de validación. En ambos casos no se consideraron las anotaciones de tipo Abbreviation.

Algunos otros tipos de errores fueron vistos con menor frecuencia. Uno de los más destacables es el caso de las entidades discontinuas.

Ejemplo 5.3.6. Vasos iliacos, femorales,popliteos y tibiales permeables en ambos miembros inferiores.

El caso de 5.3.6 es un buen ejemplo de lo que sucede frente a una discontinuidad en las anotaciones. Recordemos que, cuando se hace la conversión a *BIO format*, este caso es resuelto con una estrategia que implica hacer una sola gran anotación que incluya los fragmentos intermedios, que originalmente no fueron anotados. En este caso, se anotaron manualmente 4 entidades de tipo Anatomical Entity: "Vasos iliacos", "Vasos femorales", "Vasos popliteos" y "Vasos tibiales". Sin embargo, el modelo predice una sola anotación: "Vasos iliacos, femorales, popliteos y tibiales". Es un caso imposible de resolver con nuestra forma de modelado actual y, a diferencia del anidamiento o el desbalanceo, no se resuelve haciendo una segmentación de modelos.

En general los errores vistos parecen provenir de decisiones de modelado o de errores en el criterio de anotación, y no de una falta de entrenamiento del modelo, sobreajuste, inconvenientes en la convergencia u otros problemas que pueden surgir al entrenar modelos

de *Machine Learning*. El principal aspecto que identificamos para mejorar el desempeño en la tarea es la segmentación de modelos por tipo de entidad: en lugar de tener un solo modelo para todas las entidades, tener algunos modelos dedicados a entidades específicas podría aliviar el problema de los solapamientos. Sigue quedando el caso de las anotaciones discontinuas como un problema más difícil de resolver y para el que aún no tenemos una propuesta clara.

5.4. Conclusiones del capítulo

En este capítulo hemos analizado los resultados obtenidos en la tarea de reconocimiento de entidades nombradas en informes de ecografías en español (SpRadIE), evaluando distintos modelos de lenguaje y representaciones de texto. A través del análisis de métricas obtenidas mediante *bootstrapping*, observamos que los modelos basados en Transformers superaron a los enfoques más tradicionales, con un margen de mejora de hasta 5.5 puntos porcentuales en F1-Score. También evidenciamos que el *fine-tuning* sobre nuestro corpus específico aporta mejoras, aunque en general la ganancia fue menor a la esperada. Mediante un estudio de ablación, confirmamos la relevancia del uso de representaciones contextuales, destacando que el modelo basado en Clinical Flair mostró una caída mínima en desempeño al ser evaluado con una arquitectura simplificada. En la comparación con el estado del arte, nuestros modelos superaron los mejores resultados publicados hasta la fecha para la tarea de SpRadIE, alcanzando un F1-Score superior en 1.4 puntos porcentuales para *match* parcial y 2.6 puntos porcentuales para *match* exacto. Finalmente, el análisis manual de errores permitió identificar limitaciones en la anotación del corpus y en el modelado del problema, señalando que una posible vía de mejora radica en segmentar modelos según el tipo de entidad y en revisar la estrategia de resolución de anotaciones solapadas y discontinuas.

6. RESULTADOS DE EVALUACIÓN INTRÍNSECA

En este capítulo presentamos algunos de los resultados obtenidos para el análisis cualitativo planteado en 4.6. Debido al alto esfuerzo manual que requiere este tipo de análisis, solo profundizamos en detalle para dos de los modelos de lenguaje entrenados: FT BETO CLINICAL WL (por considerarlo el mejor modelo en términos generales) y FT CLINICAL FLAIR (por ser el modelo que dio los mejores resultados en el test de ablación).

En la sección 4.6 mencionamos varias combinaciones posibles de técnicas de reducción de dimensionalidad y clustering. Sin embargo, al probarlas notamos que PCA (para reducción) y K-Means (para clustering) no aportaban significativamente al análisis. En el caso de PCA se probaron dos cosas:

- Reducir las dimensiones de las 768 originales a las 50 que mayor varianza describen (un 94 % aproximadamente) y luego aplicar técnicas de clustering sobre este espacio. Sin embargo, los resultados son más difíciles de interpretar que al utilizar solamente t-SNE, y al no ser factible hacer una visualización gráfica de las 50 dimensiones, también es difícil evaluar la calidad de los *clusters* hallados[1] y la búsqueda de los mejores hiperparámetros excede el objetivo de esta tesis.

- Aplicar PCA antes de aplicar t-SNE (una de las estrategias mencionadas en la sección 2.9). Con este procedimiento terminamos obteniendo un gráfico prácticamente igual al obtenido usando solo t-SNE. Además por la cantidad de datos usados, aplicar directamente t-SNE resulta factible en tiempos de cómputo por lo que no hay una ganancia aquí por usar PCA antes.

Descartada entonces la utilización de PCA y K-Means, en lo que sigue se trabaja sobre los *embeddings* con dimensionalidad reducida mediante la técnica de t-SNE y el *clustering* se hace utilizando DBSCAN. Recordar que cada uno de los *embeddings* originales, sobre los que se aplica t-SNE, representan un informe completo de la partición de test de `anonymized-ultrasounds-80k`.

En las figuras 6.1 y 6.2 pueden observarse los *clusters* correspondientes a los modelos FT BETO CLINICAL WL y FT CLINICAL FLAIR respectivamente. Ambos muestran la representación de los puntos en un espacio reducido de 2-dimensiones generado mediante la técnica de t-SNE. Es importante tener en cuenta que estos puntos fueron obtenidos con la siguiente configuración[2]: `perplexity=30` e `iteraciones del algoritmo=1000`. Esto es importante ya que estudios empíricos han mostrado que la representación generada por t-SNE puede ser muy sensible a estos hiperparámetros[3], por lo que cambios en sus valores pueden conducir a representaciones diferentes. Adicionalmente, el coloreo de los puntos

[1] Si bien no lo hemos mencionado hasta ahora dado que no es el foco que queremos darle a este análisis, existen una serie de métricas que intentan evaluar la calidad con la que se segmentan los clusters: el Coeficiente de Silhouette, el índice Rand, el índice Davies-Bouldin, entre otros. El siguiente artículo hace una breve descripción de los métodos más usados https://medium.com/@surajsutar37/evaluation-metrics-for-clustering-algorithms-c9baee50e328.

[2] Según la documentación de sklearn, cuya implementación de t-SNE utilizamos aquí, se sugiere utilizar un valor de *perplexity* entre 5 y 50. Además, a mayor cantidad de datos se recomiendan valores más grandes y que el valor escogido siempre sea menor al número de muestras. Fuente: https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html

[3] Ver https://distill.pub/2016/misread-tsne/

está basado en la técnica de *clustering* de DBSCAN. Aquí nuevamente la elección de los hiperparámetros juega un rol clave sobre la forma y tamaño que tendrán los *clusters*. Es conveniente notar que, sin embargo, nuestro objetivo no es obtener una clusterización que sea útil en sí misma para alguna tarea, sino que simplemente queremos tener una orientación sobre como analizar la similitud de informes cercanos en el espacio, de acuerdo a la representación bidimensional arrojada por t-SNE. Esto implica que podemos ser flexibles en cuanto a la forma de clusterizar siempre y cuando sea útil para nuestro análisis.

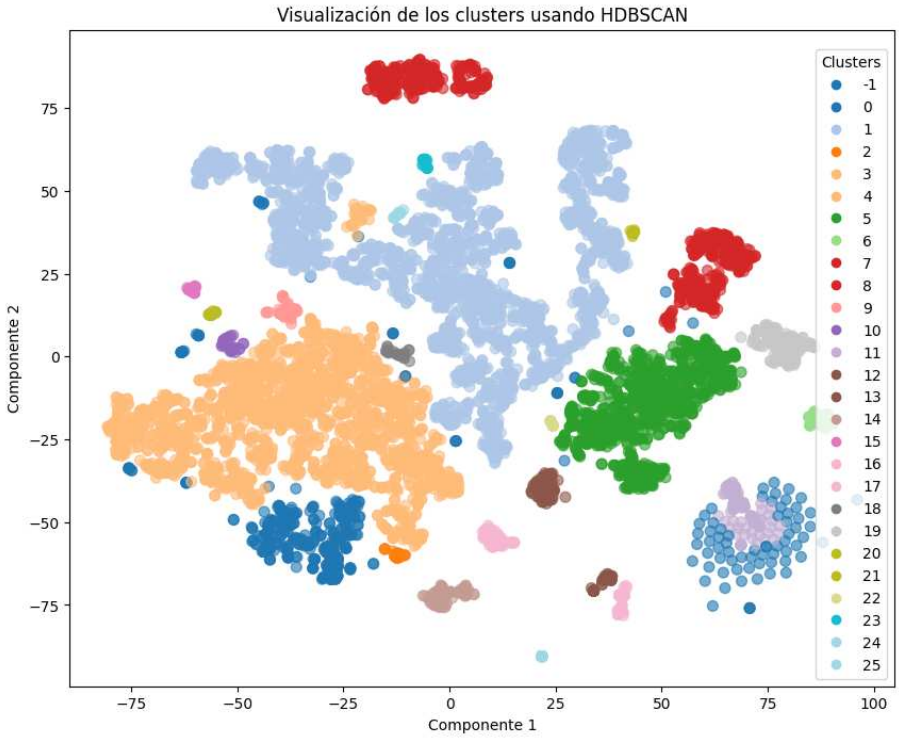

Fig. 6.1: Visualización en 2-dimensiones de los embeddings generados por el modelo FT BETO CLINICAL WL para los informes de la partición de Test de `anonymized-ultrasounds-80k`, usando el método de t-SNE. Los parámetros utilizados de t-SNE para generar esta representación fueron: `perplexity=30` e `iteraciones=1000`. El coloreo fue obtenido tras aplicar el algoritmo de DBSCAN con los hiperparámetros `eps=3.3`, `min_samples=10`, `leaf_size=30`. A la derecha de la figura se asignan números a cada *cluster*.

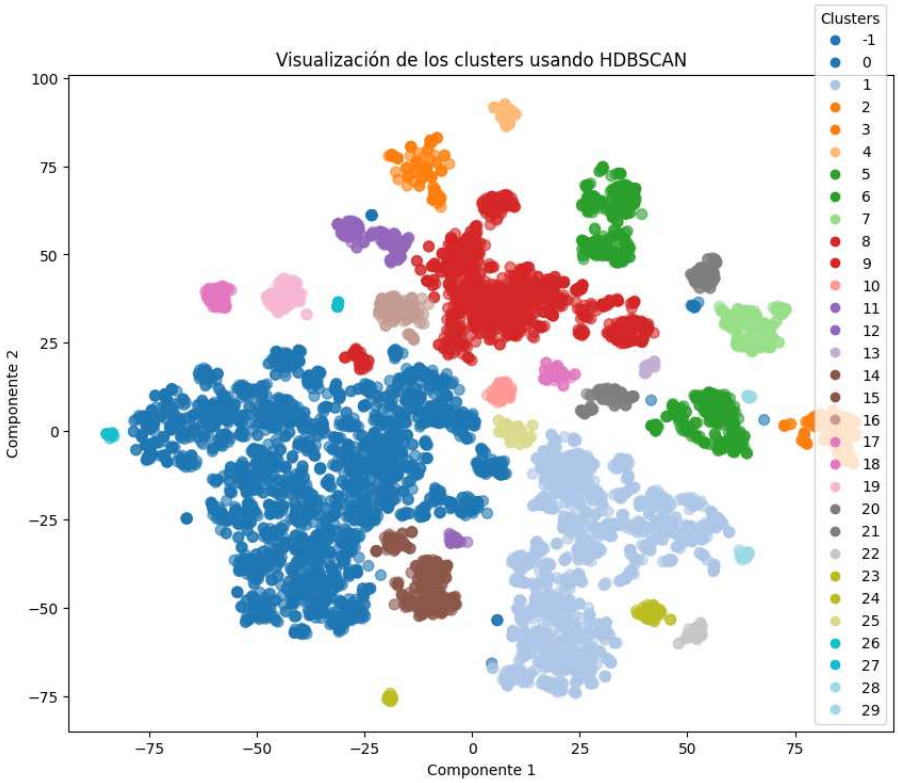

Fig. 6.2: Visualización en 2-dimensiones de los embeddings generados por el modelo FT CLINICAL FLAIR para los informes de la partición de Test de `anonymized-ultrasounds-80k`, usando el método de t-SNE. Los parámetros utilizados de t-SNE para generar esta representación fueron: `perplexity=30` e `iteraciones=1000`. El coloreo fue obtenido tras aplicar el algoritmo de DBSCAN con los hiperparámetros `eps=3.3`, `min_samples=10`, `leaf_size=30`. A la derecha de la figura se asignan números a cada *cluster*.

A continuación comentamos algunos fenómenos interesantes que hallamos al profundizar en el análisis de cada uno de los *clusters* generados. Antes de eso es importante tener en cuenta algunas consideraciones: por la naturaleza semi-estructurada de los informes, muchos tienen sintaxis y vocabularios altamente similares entre sí. Esto naturalmente hace que estén muy cercanos en el espacio vectorial de forma más o menos independiente del algoritmo usado. Dicho de otra forma: incluso un muy mal algoritmo de embeddings va a proyectar dos textos idénticos al mismo vector siempre. Por lo tanto, podemos considerar como un caso trivial aquellos *clusters* donde los informes tienen variación casi nula, y enfocarnos más en aquellos ejemplos que se salen de esta norma. También es bueno notar que, aunque puede resultar tentador hacer una comparación directa entre las representaciones correspondientes a los modelos FT BETO CLINICAL WL y FT CLINICAL FLAIR, no es lo más recomendable debido a que t-SNE es un algoritmo estocástico y la visualización generada puede variar arbitrariamente según el conjunto de vectores que se le pasen como entrada. Igualmente, como observamos luego, puede destacarse que aunque la representación espacial varíe un poco, el tipo de *clusters* que se forman en cada caso no es tan diferente entre sí (es decir, las agrupaciones suelen darse alrededor de los mismos tópicos, o al menos muy similares).

Empecemos analizando particularidades que hallamos en algunos clusters del modelo

FT BETO CLINICAL WL. Por ejemplo, haciendo una rápida inspección manual de los informes pertenecientes al *cluster* número 5 (según la numeración presentada a la derecha en la figura 6.1) nos encontramos con que los mismos parecen pertenecer a dos subtipos de ecografías: ecografías de partes blandas[4] y Doppler venoso. En la figura 6.3 se resalta dicho *cluster* en el espacio vectorial reducido de tSNE y puede notarse que en la practica en realidad se aprecian dos sub-clusters: uno más grande en la parte superior y otro más pequeño en la parte inferior, aunque ambos se encuentran muy próximos entre sí. En la tabla 6.1 listamos apenas algunos ejemplos de informes pertenecientes a dicho *cluster*. El *cluster* tiene casi 1000 informes.

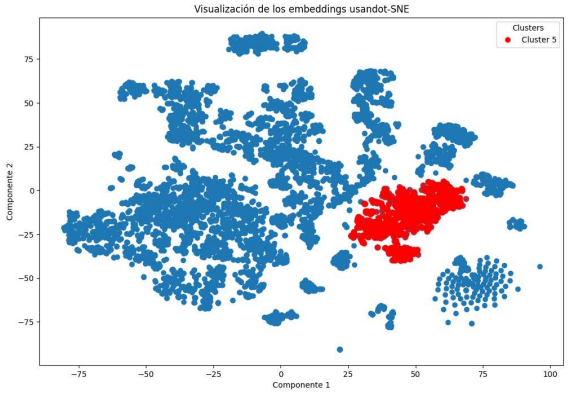

Fig. 6.3: Visualización en 2-dimensiones de los embeddings generados por el modelo FT BETO CLINICAL WL para los informes de la partición de Test de `anonymized-ultrasounds-80k`, usando el método de t-SNE. En rojo se señalan únicamente aquellos puntos que pertenecen al *cluster* 5, según la clusterización presentada en la Figura 6.1.

Adicionalmente a esta muestra aleatoria, también analizamos algunos casos en forma dirigida a analizar la diferencia entre los dos sub-clusters mencionados anteriormente y ver si había una relación posible con los dos tipos de ecografías identificadas. De esta forma hallamos que efectivamente el *cluster* inferior y más pequeño está orientado a estudios del tipo Doppler, mientras que el superior se enfoca más en los estudios articulares y subcutáneos (es decir, partes blandas). Sin embargo, ninguno de los dos es completamente "puro" y esta mezcla particular parece tener sentido al analizar que hay algunos informes donde se involucran tanto estudios Doppler como de partes blandas, sugiriendo que son cosas que en muchas ocasiones se realizan juntas. Por ejemplo, pueden verse los siguientes casos:

> **Informe 58313:** Paciente con IRC, realizacion de fistula 24 hs previas. Se observa flujo arterial (braquial) y venoso (antecubital) conservado , union de anastomosis sin imagenes sugerentes de estenosis. Anastomosis con flujo conservado, trazado Doppler normal. Alteracion del TCS (aumento de espesor y ecogenicidad), sin imagenes sugerentes de coleccion.
>
> **Informe 93099:** Se exploro pantorrilla izquierda. No se observan alteraciones

[4] Citado de `https://www.dra-martinezmiravete.com/ecografia-partes-blandas/` : "El objetivo de las ecografías de partes blandas es evaluar tanto la anatomía como la funcionalidad de los diferentes componente osteoarticulares que nos permite valorar estructuras musculares, tendinosas y tejido subcutáneo. Esta ecografía nos permite realizar el diagnóstico, tratamiento y seguimiento de las lesiones."

En realcion a lo palapable en region subamaxilar derecha se observa imagen heterogenea, hipecoica, de bordes irregulares, que mide aproximadamente 2 x 1 x 2 cm. La misma impresiona interrumpir la cortical del hueso maxilar en su rama horizontal. Aumento de la ecogenicidad del tejido celular subcutaneo. Presencia de adenopatias laterocervicales el mayor de los cuales mide 2 cm. Glandulas Tlroides, parotidas y submaxilares sin particularidades.
ECOGRAFIA DE PARTES BLANDAS En rodilla izquierda se observa minimo aumento de espesor de partes blandas ,no se observan colecciones intrarticulares ni extrarticulares, ni modificaciones en la ecogenicidad de partes blandas, ni alteraciones en el periostio al mamento del examen.
Parotida derecha aumentada de tamaño, de ecoestructura heterogenea con multiples imagenes hipoecoicas redondeadas en su interior y ganglios intraparotideos. Mide: 47 x 30 x 23 mm. Parotida derecha de ecoestructura conservada. Mide. 44 x 26 x 19 mm. Tiroides de ecoestructura conservada. Dg presuntivo: parotiditis recidivante.
Se realiza ecografia a nivel de linea media sobre lesion palpable y visible. Se observa imagen hipoecoica , ovalada, circunscripta a nivel de TCS de 1.6 x 0.46 x 1.5 cm, sin vascularizacion. Se encuentra a 5.8 mm de la piel al centro de la lesion. No se observa compromiso de estructuras profundas. Dicha formacion podria corrsponder a lipoma en primer termino.
En mama derecha por debajo de areola a nivel del TCS se visualiza imagen heterogenea a predominio hipoecoico de 1,5 x 1 x 1,8 cm, que podria corresponder a coleccion.
Se observa por debajo de herida quirurgica en fosa iliaca izquierda, en Douglas y fosa iliaca derecha imagen heterogenea de predominio hipoecoica no vascularizada compatible con coleccion. Dimensiones aproximadas: 7,6 x 11,2 x 3,9 cm
Se evaluó flujo de vasos penianos observandose sefial Doppler color arterial conservada bilateral y simétrica, sin evidancia de imagenes que sugieran fistula o malformación vascular. Se identificó trayecto lineal ecogénico en cuerpo cavernoso derecho de aprox. 1.3 x 0.3 x 0.2cm
Leve engrosamiento difuso de partes blandas en el musio izquierdo sin evidencia de colecciones. Se observo permeabilidad con sefial doppler color conservada para arteria y vena en trayecto femoral comun y superficial izquierdo. Psoas y cadera izquierda libres.
ECOGRAFIA DE PARTES BLANDAS En dorso de pie derecho, entre el plano muscular y los huesos del tetarso se observa pequefia coleccion de 1.1 cm x 1.4 cm x 0.3 cm. Se observa liquido en la vaina del extensor comun de los dedos y aumento de espesor y de ecogenicidad de partes blandas de dorso de pie.
Ambas mamas presentan ecoestructura heterogenea difusa, sin evidencia de imagenes nodulares solidas ni quisticas. Areas axilares libres.
En cavidad abdominal se observa importante ascitis no tabicada. Se realiza marcacion en piel a nivel de Fosa iliaca derecha con distancia de 6.5 cm desde piel hasata punto medio de mayor volumen de liquido.
Se exploraron arteria carotida, vena yugular y art y vena subclavia bilateral. Se observa disminucion de flujo en vena yugular derecha con onda demenor amplitud. Adecuada sefial Doppler color y espectral en el resto sde los vasos.

Tab. 6.1: Muestra aleatoria de 12 informes pertenecientes al *cluster* número 5 (según la clusterización presentada en la Figura 6.1).

en el TCS ni plano muscular. Señal Doppler color y espectral conservada para arteria y vena poplitea y tibial posterior izquierda.

Si bien en ambos casos los informes están más centrados en el estudio de los vasos sanguíneos ("Doppler", "flujo arterial y venoso", "anastomosis") también hacen alusiones al tejido subcutáneo ("TCS"). Es interesante ver como los embeddings logran capturar la relación entre estos dos tipos de estudios, especialmente en un subconjunto de los informes donde el estilo de escritura es particularmente heterogéneo.

Vemos ahora un caso del modelo FT CLINICAL FLAIR pero en lugar de analizar un único *cluster* tomamos un conjunto de 6 *clusters* que se encuentran cercanos en el espacio. Para simplificar la visualización, en la imagen 6.4 dejamos coloreados únicamente los puntos correspondientes a los clusters que vamos a analizar en detalle y el resto quedan en gris. Además en este caso agregamos, junto al número del *cluster*, una breve referencia al tipo de contenido principal de cada uno (ginecológicos, lóbulos tiroideos y testículos). Simplificando un poco, lo que queda en gris puede dividirse en tres grandes segmentos: sistema urinario (vejiga, riñones, uréteres), sistema digestivo (páncreas, hígado, vesícula, etc) y partes blandas (como ya se menciono anteriormente). Estos segmentos también pueden asociarse a distintos *clusters* aunque la división es un poco más difícil de visualizar debido a la cantidad de puntos y su cercanía en el espacio (probablemente, porque son comunes los informes que integran dos o más segmentos). Más adelante hablamos un poco de los *clusters* de sistema urinario, pero no profundizamos en el análisis del resto. Es algo que podría analizarse para futuros trabajos.

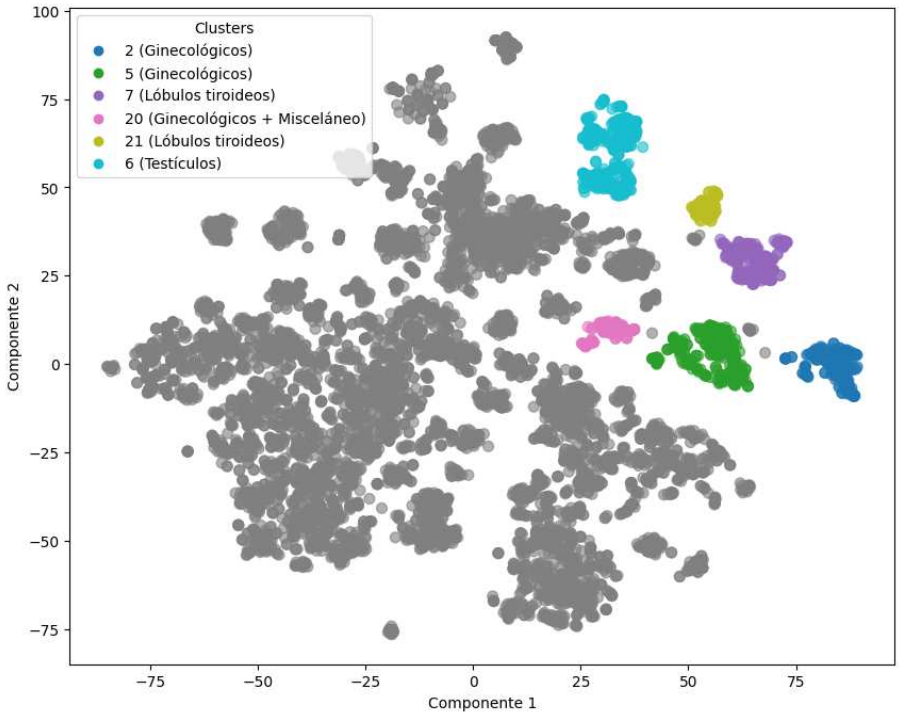

Fig. 6.4: Visualización en 2-dimensiones de los embeddings generados por el modelo FT CLINICAL FLAIR para los informes de la partición de Test de `anonymized-ultrasounds-80k`, usando el método de t-SNE. Se colorean, con diferentes colores, los clusters 2, 5, 6, 7, 20 y 21 (siguiendo la numeración presentada en la Figura 6.2). El resto de los puntos aparecen pintados de gris. Adicionalmente, el cuadro de referencias (en la parte superior izquierda) complementa cada número de *cluster* con un rótulo que resume el contenido de los informes de dicho *cluster*.

La elección de estos cúmulos se debe a que se encuentran en una misma región del espacio pero a su vez están marcadamente separados entre sí, lo que nos lleva a hipotetizar que debería haber factores similares entre ellos pero a su vez claras diferencias. En efecto, si por ejemplo empezamos observando los clusters 2, 5 y 20 hallamos que los tres se centran en el estudio de entidades ginecológicas: principalmente útero y ovarios. Sin embargo, la forma en que los informes están escritos es notoriamente diferente entre ellos. En la tabla 6.2 se muestran algunos ejemplos, a modo ilustrativo, de cada uno de estos tres *clusters*.

Cluster	Informe de ejemplo
2	Ambos ovarios y utero de caracteristicas ecograficas normales. Dimensiones: Ovario derecho: 3.4(cm) x 3.8(cm) x 1.8(cm) Volumen: 12(cc) Ovario izquierdo: 5,4(cm) x 3.4(cm) x 2(cm) Volumen: 17(cc) Utero: diametro longitudinal 8.3(cm), anteroposterior 2.2(cm), transversal 2.7(cm) Endometrio 0.5 cm.
2	Quiste ovarico derecho de 2.8(cm) de diametro. Resto del parenquima sin alteraciones. Ovario izquierdo de caracteristicas normales. Utero en RVF, homogeneo. Endometrio homogeneo, mide 0.7 cm. Ovario derecho: 5(cm) x 2.7(cm) x 3.5(cm) Volumen: 24(cc) Ovario izquierdo: 3.5(cm) x 2.3(cm) x 1.5(cm) Volumen: 6.3(cc) Utero: diametro longitudinal 7.4(cm), anteroposterior 3.6(cm), transversal 4.5(cm) Douglas libre.
5	UTERO: En AVF, de forma,tamaño y estructura conservada. Endometrio desdoblado con pequeña coleccion liquida laminar. (espesor endometrial 6.3 mm) diametro longitudinal 10(cm), transversal 5.5(cm), anteroposterior 3(cm), OVARIO DERECHO: de forma,tamaño y estructura conservada 3(cm) x 2.6(cm) x 2.2(cm) Volumen: 9.4(cc) OVARIO IZQUIERDO: de forma,tamaño y estructura conservada 3.7(cm) x 2.6(cm) x 2.3(cm) Volumen: 12(cc)
5	Paciente con antecedentes de ooforectomia izquierda. UTERO: En AVF, de forma, tamaño y estructura conservada. Endometrio: 4 mm. diametro longitudinal 7(cm), transversal 3(cm) anteroposterior 1.8(cm) OVARIO DERECHO: de forma, tamaño y estructura conservada 4.6(cm) x 2.3(cm) x 2.3(cm) Volumen: 13.3(cc)
20	Paciente con antecedente de reseccion de ovario derecho. Presenta utero tubular de caracteristicas infantiles. Ovario unico izquierdo de tamaño, forma y ecoestructura conservada, con multiples imagenes foliculares . La mayor de 0.60 cm. Tamaño: 2.4 cm x 0.88 cm x 1.1 cm. Vol.: 1.2 cc. No se observa liquido libre.
20	Se visualiza utero de 1.5cm x 1 cm de caracteristicas infantiles. No puedo visualizar ovarios. Rinon derecho muy pequeño de 2.8cm Rinon izquierdo de caracteristicas normales sin dilataciones.Longitudinal 8.8cm.

Tab. 6.2: Ejemplos de informes tomados de los *clusters* 2, 5 y 20 (basándonos en la numeración que aparece en la figura 6.2). La primer columna muestra el número del *cluster*, mientras que la segunda incluye un ejemplo tomado de dicho *cluster*.

Algo llamativo del *cluster* 20 es que, a diferencia de los otros dos, pueden encontrarse referencias a otras entidades anatómicas no relacionadas a la ginecología (por ejemplo vejiga, riñones o hígado) aunque siempre hay al menos una mención al útero. Esto es coherente con la posición espacial que tiene el cluster: es el más "central" de todos, estando cerca de *clusters* enfocados en entidades anatómicas de los sistemas urinario y digestivo.

Por otra parte, los *clusters* 7 y 21 se caracterizan por ser informes sobre ecografías de los lóbulos tiroideos. Si bien ambos *clusters* suelen mantener estructuras muy similares, hay una característica bastante interesante que los diferencia: los informes del *cluster* 7 tratan de "lóbulos tiroideos (e istmo) con estructura homogénea (o finamente heterogénea)", mientras que los del *cluster* 21 presentan "lóbulos tiroideos con eco-estructura heterogénea difusa, no no-

dular". Es interesante como esta diferencia (generalmente presente al comienzo de los informes) logra diferenciar marcadamente a los informes, aun cuando el resto de su estructura puede ser muy similar. Esto probablemente se deba a la capacidad de los modelos de lenguaje de dar mayor peso a ciertas partes del texto, de acuerdo al contexto específico. Por último, el *cluster* 6 está enfocado en ecografías testiculares de pacientes varones. Es destacable como en este *cluster* el estilo de escritura es bastante heterogéneo así como el uso de abreviaciones, por ejemplo: "testiculo izquierdo" puede encontrarse también como "testiculo izq.", "TI", "TIz", entre otras. Esto resalta la capacidad del modelo de lenguaje de poder capturar similaridad semántica más allá de la similaridad sintáctica.

Volviendo al motivo original por el que escogimos estos seis clusters, nos preguntamos entonces si hay un motivo particular para su cercanía relativa. Creemos que hay dos motivos principales. Uno es la estructura de los informes que comparte ciertos razgos en común, al menos en los casos más estructurados: "Ambos ... presentan ... Dimensiones: Entidad izquierda: ... Entidad derecha: ... Otras entidades: ...". Este ejemplo de estructura muestra el otro motivo que, consideramos, es compartido y puede influenciar la distribución: en todas estas entidades prevalece el uso de la palabra "ambos" debido a la naturaleza simétrica de los órganos involucrados: ovarios, testículos y lóbulos (tiroideos en este caso). De hecho esta prevalencia de la palabra "ambos" podemos verla en todo el semieje positivo del Componente 1 de la figura 6.2. Los *clusters* 1, 22, 24 y 29 (que se encuentran ahí y pueden visualizarse con mayor claridad en la figura 6.5) se centran en sistema urinario, con una fuerte presencia de menciones a los riñones y (en mucho menor medida) los uréteres, que suelen presentarse también con la palabra "ambos" ("ambos riñones" o "ambos uréteres"). En oposición, en el semieje negativo del Componente 1 se encuentran clusters centrados en entidades anatómicas como hígado, páncreas, vía biliar, bazo, arterias, venas y otras que no comparten esta propiedad de duplicidad/simetría. Aún si aparecen algunas menciones a "ambos riñones", distan de ser el tópico central de los informes, a diferencia de los casos antes mencionados.

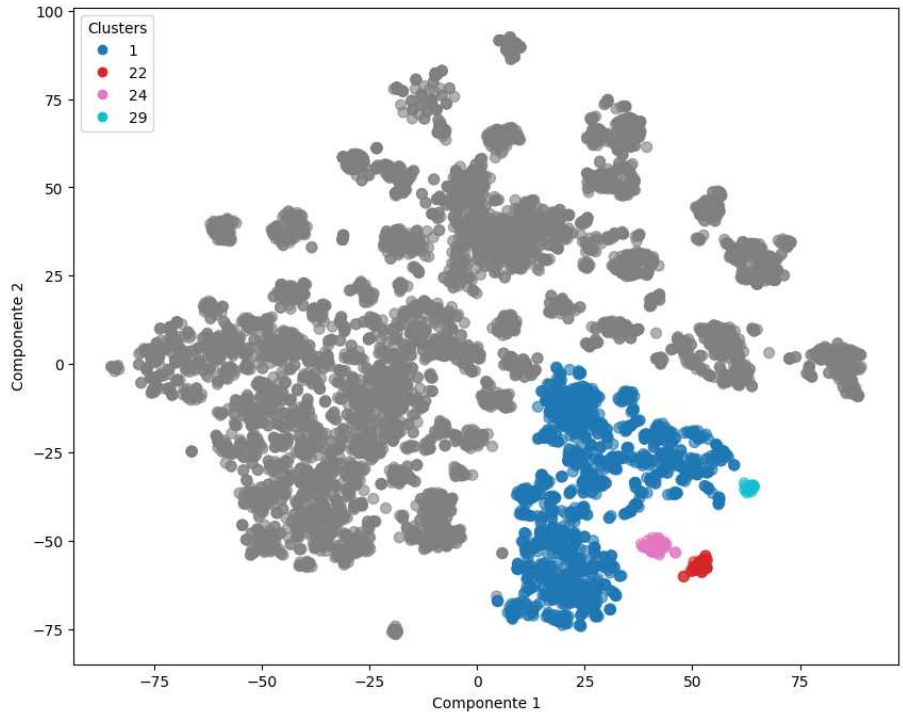

Fig. 6.5: Visualización en 2-dimensiones de los embeddings generados por el modelo FT CLINICAL FLAIR para los informes de la partición de Test de `anonymized-ultrasounds-80k`, usando el método de t-SNE. Se colorean, con diferentes colores, los clusters 1, 22, 24 y 29 (siguiendo la numeración presentada en la Figura 6.2). Estos *clusters* se centran principalmente en el sistema urinario. El resto de los puntos aparecen pintados de gris.

6.1. Conclusiones del capítulo

En este capítulo realizamos un análisis cualitativo de los *embeddings* generados por los modelos FT BETO CLINICAL WL y FT CLINICAL FLAIR para evaluar su capacidad de estructuración semántica sobre informes ecográficos. Tras probar diversas combinaciones de técnicas de reducción de dimensionalidad, optamos por la combinación de t-SNE para reducción y DBSCAN para *clustering*. A partir de la representación en dos dimensiones, identificamos patrones de agrupación que reflejan la similitud semántica de los informes, destacando la formación de *clusters* coherentes en torno a tipos de estudios específicos, como ecografías ginecológicas, de partes blandas, testiculares y del sistema urinario. El análisis manual reveló que la segmentación se ve influenciada tanto por el contenido anatómico de los estudios como por características estilísticas y de redacción, evidenciando la capacidad de los modelos de lenguaje para capturar estas relaciones. Además, observamos que la proximidad entre ciertos *clusters* sugiere que los modelos reconocen relaciones semánticas entre estudios complementarios, como la coexistencia de evaluaciones Doppler con estudios musculoesqueléticos. Finalmente, la estabilidad de las agrupaciones entre ambos modelos indica que, si bien las visualizaciones pueden variar por la naturaleza estocástica de t-SNE, las estructuras semánticas subyacentes se

preservan, lo que refuerza la validez de los *embeddings* como representaciones útiles para la organización de información en informes médicos.

7. CONCLUSIONES Y TRABAJO FUTURO

En este trabajo nos propusimos estudiar el impacto de contar con *embeddings* específicamente entrenados para análisis de texto médico, particularmente en un sub-dominio poco explorado hasta ahora: informes de ecografías escritos en español. Se analizaron distintos tipos de *embeddings*, tanto estáticos (FastText) como contextuales (BiLSTM y Transformers). Para su entrenamiento fue necesario realizar algunos procesamientos sobre nuestro *corpus*, siendo el más complejo de todos la anonimización. Para la evaluación extrínseca, usamos la tarea de reconocimiento de entidades nombradas de la competencia SpRadIE. Mientras que para la evaluación intrínseca, planteamos un análisis cualitativo usando reducción de dimensionalidad y *clustering*.

Los resultados expuestos en el capítulo 5 muestran que, al menos para la tarea de reconocimiento de entidades nombradas sobre el *corpus* de SpRadIE, efectivamente se logró tener un impacto positivo en los resultados, superando a los mejores resultados publicados hasta ahora. Además se observó que la utilización de *domain adaptation* permitió alcanzar mejores resultados, comparando con las variantes que no lo usaban. Esto último resulta un incentivo importante para seguir buscando generar representaciones más específicas de los distintos dominios clínicos.

Se vio que los *embeddings* contextuales (basados en modelos de lenguaje) superaron en todos los casos a nuestros *embeddings* estáticos, que implementamos usando FastText.

Observamos que los métodos basados en *Transformers* superaron con un margen pequeño pero significativo a aquellos basados en BiLSTMs (Flair) en la tarea de NER. Sin embargo, como se vio en el estudio de ablación, si simplificamos la capa de clasificación y partimos de los *embeddings* generados por cada método, esta relación se revierte. Esto sugiere que en términos de capacidad para capturar relaciones semánticas, el *embedding* generado por Flair puede resultar más eficiente. Al hacer el análisis cualitativo de las representaciones notamos que ambas comparten características similares, no pudiendo sacar conclusiones significativas sobre las diferencias. Sí pudimos observar propiedades deseables en cuanto a la proximidad de las representaciones y su *relatedness*.

Como un resultado de este trabajo, se obtuvieron varios modelos de lenguaje basados en BERT y Flair finetuneados para informes de ecografías escritos en español. Se podrá evaluar a futuro su publicación, con la debida aprobación de los dueños de los datos utilizados, para que la comunidad pueda probarlos en otras tareas. Además, como se comentó, fue ejecutado un proceso exhaustivo de anonimización sobre los informes involucrados. Nuevamente, con aprobación de las instituciones correspondientes, dicho *corpus* anonimizado también podría ser publicado, siendo un recurso de gran valor dada la escasez de *corpora* de dominio clínico que se ha mencionado al comienzo de este trabajo.

7.1. Trabajo futuro

En línea con tratar de mejorar aún más los resultados obtenidos para la tarea de SpRadIE, se señala en el capítulo 5 que sería provechoso aplicar un marco de trabajo más adecuado para tratar las entidades disjuntas y anidadas. Para estas últimas, un enfoque basado en segmentar modelos para reconocer entidades específicas (por ejemplo, un modelo que solo identifique Location, otro que solo identifique Degree, etc) debería conducir a una mejora de desempeño. Una prueba que quedó por fuera de esta tesis es la de probar los más recientes *Large Language Models*, que tienen la particularidad de destacarse en escenario de *zero-shot* y *few-shot learning*, por lo que podrían aportar un valor interesante en este escenario en el que contamos con un número relativamente pequeño de datos. También, al tratarse de modelos generativos, podrían ser una forma interesante de lidiar con los casos de entidades disjuntas, dado que no están limitados en el formato de su respuesta, a diferencia de los modelos de lenguaje expuestos en esta tesis.

Un aspecto completamente diferente que apareció al hacer los análisis, es que se identificaron oportunidades (algunas concretas y otras que requieren revisarse junto con un médico) para hacer mejoras en el *corpus*usado en la tarea de SpRadIE.

En este trabajo se probó con un *challenge* específico de BioNLP (SpRadIE) pero sería interesante a futuro poder evaluar la utilidad de los *embeddings* y modelos de lenguaje aquí presentados en otros conjuntos de datos, incluso saliendo de la tarea de NER (por ejemplo, para detección de relaciones entre entidades nombradas). Por otro lado, para el entrenamiento de los modelos de *embeddings* nos limitamos a utilizar informes de ecografías ya que eran afines al conjunto de informes de SpRadIE e incluían algunos preprocesamientos ya realizados en trabajos pasados. Sin embargo, contamos con otros tipos de informes provistos por la misma institución que incluyen resonancias magnéticas y tomografías computarizadas. Sin dudas es de interés incluirlos como parte del *corpus*de entrenamiento de los modelos (posterior a realizar un proceso de revisión y anonimización) con la hipótesis de obtener *embeddings* útiles para más sub-dominios relacionados. También nos permitiría validar si "diversificar" los sub-dominios sobre los que se entrenan los *embeddings* tiene un impacto (positivo o negativo) sobre las tareas específicas.

Por último, si bien en este trabajo propusimos un análisis cualitativo como forma de intentar tener una noción de la calidad de los *embeddings* sin necesidad de depender de una tarea externa, este análisis no es fácil de escalar (debido a que es un proceso mayoritariamente manual y subjetivo) ni permite realizar comparaciones concluyentes entre modelos. Por lo tanto, vemos un área de oportunidad en construir un conjunto de *benchmark* que incluya casos de palabras similares o relacionadas (del mismo modo que lo hacen UMNSRS-sim y UMNSRS-rel) para el sub-dominio de informes de ecografías en español.

BIBLIOGRAFíA

[1] Hervé Abdi y Lynne J Williams. "Principal component analysis". En: *Wiley interdisciplinary reviews: computational statistics* 2.4 (2010), págs. 433-459.

[2] Alan Akbik, Duncan Blythe y Roland Vollgraf. "Contextual string embeddings for sequence labeling". En: *Proceedings of the 27th international conference on computational linguistics*. 2018, págs. 1638-1649.

[3] Liliya Akhtyamova et al. "testing contextualized word embeddings to improve NER in Spanish clinical case narratives". En: *IEEE Access* 8 (2020), págs. 164717-164726.

[4] Emily Alsentzer et al. "Publicly available clinical BERT embeddings". En: *arXiv preprint arXiv:1904.03323* (2019).

[5] Pablo Báez et al. "The Chilean Waiting List Corpus: a new resource for clinical named entity recognition in Spanish". En: *Proceedings of the 3rd clinical natural language processing workshop*. 2020, págs. 291-300.

[6] Yoshua Bengio, Patrice Simard y Paolo Frasconi. "Learning long-term dependencies with gradient descent is difficult". En: *IEEE transactions on neural networks* 5.2 (1994), págs. 157-166.

[7] Piotr Bojanowski et al. "Enriching word vectors with subword information". En: *Transactions of the association for computational linguistics* 5 (2017), págs. 135-146.

[8] Tom Brown et al. "Language Models are Few-Shot Learners". En: *Advances in Neural Information Processing Systems*. Ed. por H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, págs. 1877-1901. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[9] Aurelia Bustos et al. "Padchest: A large chest x-ray image dataset with multi-label annotated reports". En: *Medical image analysis* 66 (2020), pág. 101797.

[10] José Cañete. "Compilation of large spanish unannotated corpora". En: *Zenodo, mayo de* (2019).

[11] José Cañete et al. "Spanish Pre-Trained BERT Model and Evaluation Data". En: *PML4DC at ICLR 2020*. 2020.

[12] Casimiro Pio Carrino et al. *Biomedical and Clinical Language Models for Spanish: On the Benefits of Domain-Specific Pretraining in a Mid-Resource Scenario*. 2021. arXiv: 2109.03570 [cs.CL].

[13] Jacob Cohen. "A coefficient of agreement for nominal scales". En: *Educational and psychological measurement* 20.1 (1960), págs. 37-46.

[14] Ronan Collobert et al. "Natural language processing (almost) from scratch". En: *Journal of machine learning research* 12 (2011), págs. 2493-2537.

[15] Viviana Cotik et al. "Annotation of Entities and Relations in Spanish Radiology Reports". En: *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*. Ed. por Ruslan Mitkov y Galia Angelova. Varna, Bulgaria: INCOMA Ltd., sep. de 2017, págs. 177-184. DOI: 10.26615/978-954-452-049-6_025. URL: https://aclanthology.org/R17-1025/.

[16] Viviana Cotik et al. "Overview of CLEF eHealth Task 1-SpRadIE: A challenge on information extraction from Spanish Radiology Reports". En: *CLEF 2021 Evaluation Labs and Workshop: Online Working Notes. CEUR-WS*. 2021.

[17] Viviana Cotik et al. "Overview of CLEF eHealth Task 1-SpRadIE: A challenge on information extraction from Spanish Radiology Reports." En: *CLEF (Working Notes)*. 2021, págs. 732-750.

[18] Viviana Erica Cotik. "Extracción de información en informes radiológicos escritos en español". Tesis doct. Universidad de Buenos Aires. Facultad de Ciencias Exactas y Naturales, 2018.

[19] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". En: *arXiv preprint arXiv:1810.04805* (2018).

[20] Giorgio Maria Di Nunzio et al. "IMS-UNIPD@ CLEF eHealth task 1: a memory based reproducible baseline". En: *CEUR WORKSHOP PROCEEDINGS*. Vol. 2936. CEUR-WS. 2021, págs. 770-774.

[21] David M Eberhard, Gary F Simons y Charles D Fennig. "What are the top 200 most spoken languages". En: *Ethnologue: Languages of the world* (2021).

[22] Hermenegildo Fabregat et al. "LSI_UNED at CLEF eHealth2021: Exploring the effects of transfer learning in negation detection and entity recognition in clinical texts." En: *CLEF (Working Notes)*. 2021, págs. 780-793.

[23] Hermenegildo Fabregat et al. "LSI_UNED at CLEF eHealth2021: Exploring the effects of transfer learning in negation detection and entity recognition in clinical texts." En: *CLEF (Working Notes)*. 2021, págs. 780-793.

[24] Lev Finkelstein et al. "Placing search in context: The concept revisited". En: *Proceedings of the 10th international conference on World Wide Web*. 2001, págs. 406-414.

[25] Miguel Ángel Martın-Caro Garcıa-Largo e Isabel Segura-Bedmar. "Extracting information from radiology reports by Natural Language Processing and Deep Learning." En: *CLEF (Working Notes)*. 2021, págs. 804-817.

[26] Aitor Gonzalez-Agirre et al. "PharmaCoNER: Pharmacological Substances, Compounds and proteins Named Entity Recognition track". En: *Proceedings of The 5th Workshop on BioNLP Open Shared Tasks*. Hong Kong, China: Association for Computational Linguistics, nov. de 2019, págs. 1-10. DOI: 10.18653/v1/D19-5701. URL: https://aclanthology.org/D19-5701.

[27] Ian Goodfellow, Yoshua Bengio y Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[28] Suchin Gururangan et al. "Don't stop pretraining: Adapt language models to domains and tasks". En: *arXiv preprint arXiv:2004.10964* (2020).

[29] Asier Gutiérrez-Fandiño et al. "Maria: Spanish language models". En: *arXiv preprint arXiv:2107.07253* (2021).

[30] Maryam Habibi et al. "Deep learning with word embeddings improves biomedical named entity recognition". En: *Bioinformatics* 33.14 (2017), págs. i37-i48.

[31] T. Hastie, R. Tibshirani y J.H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer series in statistics. Springer, 2009. ISBN: 9780387848846. URL: https://books.google.com.uy/books?id=eBSgoAEACAAJ.

[32] Felix Hill, Roi Reichart y Anna Korhonen. "Simlex-999: Evaluating semantic models with (genuine) similarity estimation". En: *Computational Linguistics* 41.4 (2015), págs. 665-695.

[33] Sepp Hochreiter y Jürgen Schmidhuber. "Long short-term memory". En: *Neural computation* 9.8 (1997), págs. 1735-1780.

[34] Alistair EW Johnson et al. "MIMIC-III, a freely accessible critical care database". En: *Scientific data* 3.1 (2016), págs. 1-9.

[35] Dan Jurafsky. *Speech & language processing*. Pearson Education India, 2000.

[36] Katikapalli Subramanyam Kalyan y S. Sangeetha. "SECNLP: A survey of embeddings in clinical natural language processing". En: *Journal of Biomedical Informatics* 101 (2020), pág. 103323. ISSN: 1532-0464. DOI: https://doi.org/10.1016/j.jbi.2019.103323. URL: https://www.sciencedirect.com/science/article/pii/S1532046419302436.

[37] Mirza S Khan et al. "Intrinsic Evaluation of Contextual and Non-contextual Word Embeddings using Radiology Reports". En: *AMIA Annual Symposium Proceedings*. Vol. 2021. 2022, pág. 631.

[38] Faiza Khan Khattak et al. "A survey of word embeddings for clinical text". En: *Journal of Biomedical Informatics* 100 (2019). Articles initially published in Journal of Biomedical Informatics: X 1-4, 2019, pág. 100057. ISSN: 1532-0464. DOI: https://doi.org/10.1016/j.yjbinx.2019.100057. URL: https://www.sciencedirect.com/science/article/pii/S2590177X19300563.

[39] Jinhyuk Lee et al. "BioBERT: a pre-trained biomedical language representation model for biomedical text mining". En: *Bioinformatics* 36.4 (2020), págs. 1234-1240.

[40] Xiaoya Li et al. "Is word segmentation necessary for deep learning of Chinese representations?" En: *arXiv preprint arXiv:1905.05526* (2019).

[41] Yinhan Liu et al. "Roberta: A robustly optimized bert pretraining approach". En: *arXiv preprint arXiv:1907.11692* (2019).

[42] Pilar López-Úbeda et al. "Pre-trained language models to extract information from radiological reports." En: *CLEF (Working Notes)*. 2021, págs. 794-803.

[43] Tomas Mikolov et al. "Efficient estimation of word representations in vector space". En: *arXiv preprint arXiv:1301.3781* (2013).

[44] Javier Minces Müller. "Detección de relaciones en informes médicos escritos en español". Universidad de Buenos Aires, dic. de 2020.

[45] A Miranda-Escalada, E Farré y M Krallinger. "Named entity recognition, concept normalization and clinical coding: Overview of the cantemist track for cancer text mining in spanish, corpus, guidelines, methods and results". En: *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020), CEUR Workshop Proceedings*. 2020.

[46] Antonio Miranda-Escalada et al. "Overview of automatic clinical coding: annotations, guidelines, and solutions for non-english clinical cases at codiesp track of CLEF eHealth 2020". En: *Working Notes of Conference and Labs of the Evaluation (CLEF) Forum. CEUR Workshop Proceedings*. 2020.

[47] Aldrian Obaja Muis y Wei Lu. "Learning to recognize discontiguous entities". En: *arXiv preprint arXiv:1810.08579* (2018).

[48] Aurélie Névéol et al. "Clinical natural language processing in languages other than English: opportunities and challenges". En: *Journal of biomedical semantics* 9 (2018), págs. 1-13.

[49] Mariana Neves y Ulf Leser. "A survey on annotation tools for the biomedical literature". En: *Briefings in bioinformatics* 15.2 (2014), págs. 327-340.

[50] Serguei Pakhomov et al. "Semantic similarity and relatedness between clinical terms: an experimental study". En: *AMIA annual symposium proceedings*. Vol. 2010. American Medical Informatics Association. 2010, pág. 572.

[51] Serguei VS Pakhomov et al. "Towards a framework for developing semantic relatedness reference standards". En: *Journal of biomedical informatics* 44.2 (2011), págs. 251-265.

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

[53] Marco Polignano, Marco de Gemmis, Giovanni Semeraro et al. "Comparing Transformer-based NER approaches for analysing textual medical diagnoses." En: *CLEF (Working Notes)*. 2021, págs. 818-833.

[54] James Pustejovsky y Amber Stubbs. *Natural Language Annotation for Machine Learning: A guide to corpus-building for applications*. .ºReilly Media, Inc.", 2012.

[55] Matıas Rojas, Jocelyn Dunstan y Fabián Villena. "Clinical flair: A pretrained language model for Spanish clinical natural language processing". En: *Proceedings of the 4th Clinical Natural Language Processing Workshop*. 2022, págs. 87-92.

[56] Roland Roller et al. "Detecting named entities and relations in German clinical reports". En: *Language Technologies for the Challenges of the Digital Age: 27th International Conference, GSCL 2017, Berlin, Germany, September 13-14, 2017, Proceedings 27*. Springer International Publishing. 2018, págs. 146-154.

[57] Ronald Rosenfeld. "Two decades of statistical language modeling: Where do we go from here?" En: *Proceedings of the IEEE* 88.8 (2000), págs. 1270-1278.

[58] Rico Sennrich, Barry Haddow y Alexandra Birch. "Neural machine translation of rare words with subword units". En: *arXiv preprint arXiv:1508.07909* (2015).

[59] Felipe Soares et al. "Medical word embeddings for Spanish: Development and evaluation". En: *Proceedings of the 2nd clinical natural language processing workshop*. 2019, págs. 124-133.

[60] Oswaldo Solarte-Pabón et al. "Information extraction from Spanish radiology reports using multilingual BERT". En: *CLEF eHealth* (2021).

[61] Vıctor Suárez-Paniagua, Hang Dong y Arlene Casey. "A multi-BERT hybrid system for Named Entity Recognition in Spanish radiology reports". En: *CEUR Workshop Proceedings*. Vol. 2936. CEUR Workshop Proceedings. 2021.

[62] Vıctor Suárez-Paniagua, Hang Dong y Arlene Casey. "A multi-BERT hybrid system for Named Entity Recognition in Spanish radiology reports". En: *CEUR Workshop Proceedings*. Vol. 2936. CEUR Workshop Proceedings. 2021.

[63] Laurens Van der Maaten y Geoffrey Hinton. "Visualizing data using t-SNE." En: *Journal of machine learning research* 9.11 (2008).

[64] Ashish Vaswani et al. "Attention is all you need". En: *Advances in neural information processing systems* 30 (2017).

[65] Changhan Wang, Kyunghyun Cho y Jiatao Gu. "Neural machine translation with byte-level subwords". En: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 05. 2020, págs. 9154-9160.

[66] Yanshan Wang et al. "A comparison of word embeddings for the biomedical natural language processing". En: *Journal of biomedical informatics* 87 (2018), págs. 12-20.

[67] Yonghui Wu et al. "Google's neural machine translation system: Bridging the gap between human and machine translation". En: *arXiv preprint arXiv:1609.08144* (2016).

7. APÉNDICE

7.2. Lista de palabras usadas para el reconocimiento de abreviaturas con RegEx

- cm
- Diam
- RD
- RI
- mm
- long
- Long
- cc
- FID
- EP
- VN
- RX
- AP
- OV
- ECO
- VSH
- izq
- Izq
- der
- Der
- vol
- aprox
- riz
- rder

7.3. Tokenización para FastText basada en regex

```
import re

DELIMITERS = "([,.:;\s\(\)])"

def regex_tokenize_report(report):
```

```
tokenized_report = re.split(DELIMITERS, report)
tokenized_report = [token for token in tokenized_report
                    if token and not token.isspace()]
return tokenized_report
```