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

Detección de relaciones en informes médicos escritos en español

Tesis de Licenciatura en Ciencias de la Computación

Javier Minces Müller

Directora: Viviana Cotik

Buenos Aires, diciembre de 2020

Índice general

RESUMEN

La detección automática de relaciones entre entidades es una tarea importante del procesamiento del lenguaje natural. En informes médicos, en particular, la extracción de relaciones es de suma utilidad. Permite, entre otras cosas, asociar de manera automática los hallazgos clínicos descriptos en el informe con el área del cuerpo en donde ocurrieron. Esto hace posible descubrir información, que puede asistir en la toma de decisiones, de manera mucho más rápida de lo que se haría mediante un análisis manual.

Casi todos los métodos desarrollados para esta tarea están implementados para textos en idioma inglés. Estos incluyen métodos basados en reglas y en técnicas de aprendizaje automático.

En este trabajo se realizó extracción de relaciones entre entidades nombradas sobre informes de ecografías escritos en español. Estos tienen la dificultad adicional de ser de naturaleza informal. Para esto se propusieron tres métodos: uno basado en co-ocurrencia de entidades, otro basado en reglas y finalmente uno basado en redes neuronales convolucionales. Para este último se entrenaron *word embeddings* en español para textos médicos.

Se obtuvieron resultados alentadores para los últimos dos métodos, siendo mejores aquellos basados en reglas (F1 0.880 y 0.867 respectivamente). Se observó que la distancia entre las entidades relacionadas influye en los resultados.

Palabras clave: detección de relaciones, NLP, español, BioNLP, informes radiológicos, aprendizaje profundo

ABSTRACT

The detection of relationships between words is an important task of natural language processing. When dealing specifically with medical reports, relation identification is important. It allows, among other things, finding clinical conditions in a report, associated with the body part where they occur. This makes it possible to discover information, which can assist in decision making, much faster than through manual analysis.

Almost all methods developed for this task are for English texts. These include mainly rule-based methods and machine learning techniques.

In this work, we focus on relation extraction for ultrasound reports in Spanish. These texts have the additional difficulty of being informal in nature.

To this end, three methods are proposed: one based on co-occurrence, one based in rules and one based in convolutional neural networks. For the latter, word embeddings are trained in Spanish for medical texts.

Encouraging results are obtained for the last two methods, the best of these with the rule-based method (an F1 score of 0.880 and 0.867). The results are better the closer the entities are.

Keywords: relation detection, NLP, Spanish, BioNLP, radiology reports, deep learning

AGRADECIMIENTOS

A mi directora, por su infinita paciencia en este proceso que sin su empuje no hubiera terminado. También por impulsarme a difundirlo.

A los jurados, por tomarse el tiempo para leer este trabajo y darme su devolución.

A mis amigos. A Gastón y Martín por invitarme a trabajar con ellos, y por leer algunas secciones. A Nicolás por terminar antes, mostrándome así que se podía. A Julián por darme el primer empujón, y no por primera vez. A Christian por llevarme en auto infinitas veces. A Gustavo e Ignacio por mostrarme cómo la carrera te puede llevar por el mundo. A todos ellos y también a Sebastián y Christian, por todo lo que compartimos que hizo la carrera fuera mucho más que estudiar.

A mis padres, que siempre hicieron lo que estuvo a su alcance para que pudiera concentrarme en el estudio.

A la UBA por ser pública y gratuita, por darme la oportunidad de tener una formación muy completa aún trabajando y, además, por permitirme aprender enseñando.

ÍNDICE DE TABLAS

Índice de figuras

1. INTRODUCCIÓN

Motivación

En los últimos años ha tenido lugar un crecimiento constante 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 distintos tipos de textos. Podemos mencionar informes científicos, historias clínicas 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 inmediata.

El formato informal y no estructurado de estos textos hace que la extracción de información no sea trivial. Los textos suelen presentar dificultades como faltas de ortografía o falta de signos de puntuación. Además, usan un vocabulario muy específico, que varía con el dialecto de cada país, o dentro del mismo país. Distintos términos pueden referir al mismo concepto y el mismo término puede tener varios significados. Presentan también abundancia de acrónimos y abreviaturas. Es necesario tener en cuenta que la formalidad no es la principal prioridad en estos textos; suele ser más importante que la escritura sea rápida.

Este punto se puede ilustrar mejor con una oración tomada de un informe:

```
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 idioma en el que fueron escritos. Dado que el español es uno de los idiomas más hablados del mundo [2], resulta necesario tener herramientas para trabajar con recursos en ese idioma.

[1] https://www.argentina.gob.ar/salud/digital, consultado en agosto de 2020.

[2] Ver Névéol et al. [1] para una comparación detallada de la producción de artículos científicos sobre el tema en diferentes idiomas.

Si bien muchos métodos desarrollados en inglés pueden ser utilizados en otros idiomas [3], resulta necesario adaptarlos a las características de cada uno. Por ejemplo, si se usan *word embeddings* (ver Subsección 2.3.1), es necesario tener o entrenar vectores para las palabras del idioma.

Trabajar en español, además, presenta el desafío adicional de conseguir datos. No hay una gran disponibilidad de corpora.[3] Conseguir 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** 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 es del año 2017 y se describe en la Subsección 4.1.1.

Definiciones

Se llama *lenguaje natural* al lenguaje hablado y escrito por seres humanos. El **procesamiento del lenguaje natural** (NLP) es un área de estudio multidisciplinaria que se encarga de procesarlo para analizarlo o generarlo [4].

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. Este es un campo muy activo, con grandes proyectos a nivel mundial, *workshops* en los congresos más importantes del área [5] y una gran producción de trabajos de investigación y recursos en los últimos años [1, 6, 7]. Entre estos recursos podemos mencionar registros de terminologías, diccionarios de abreviaturas, entre otros.

La extracción de información como campo de estudio es la búsqueda de material que satisface una necesidad de información en grandes colecciones de textos de naturaleza no estructurada o semiestructurada [8].

Se denominan entidades nombradas a palabras o conjuntos de palabras de un tipo particular que sean relevantes para analizar. Si dos entidades están asociadas entre sí, decimos que hay una relación; podemos clasificar las relaciones en varios tipos. En la Subsección 4.1.1 detallamos tanto los tipos de entidades como los tipos de relaciones con los que se trabajó.

En la Figura 1.1 se puede ver una oración tomada de un informe ecográfico con sus entidades y relaciones etiquetadas manualmente. Cuando se habla de **anotación** en este trabajo nos referimos a este tipo de etiquetado. *Pelvis renales*, *dobles o bífidas*, *via urinaria* son ejemplos de entidades correspondientes a los tipos elegidos para este etiquetado. *Pelvis renales* es una entidad de tipo anatómico, mientras que *dobles o bífidas* hace referencia a un $finding$ (hallazgo).

[3] Un corpus (pl. corpora) es una colección de textos o discursos utilizados para un propósito específico, que puede estar anotado.

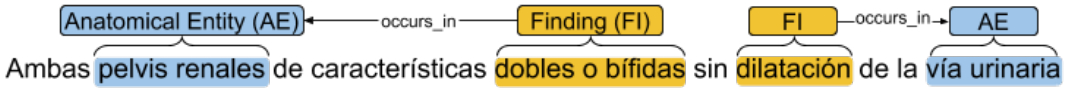

Fig. 1.1: Ejemplo de una oración tomada de un informe médico, con sus entidades y relaciones etiquetadas manualmente.

Decimos que *dobles o bífidas* está relacionada con *pelvis renales* porque en la oración habla de una característica de las pelvis renales. La relación es de tipo *occurs_in*. En cambio, *dilatación* hace referencia a otra entidad, y por lo tanto no está relacionada con las pelvis renales.

El problema que se ocupa de la detección y clasificación automática de distintos tipos predefinidos de entidades en textos no estructurados o semiestructurados es el reconocimiento de entidades nombradas (NER).

En este trabajo no nos ocuparemos de ese problema, sino que nos centraremos en el **reconocimiento de relaciones**: dado un texto con entidades ya etiquetadas, se busca encontrar cuáles de ellas están relacionadas.

Descripción del problema

La extracción de relaciones se puede enfocar como parte de un *pipeline* de procesamiento como el que se muestra en la Figura 1.2. A continuación se explican algunas etapas del *pipeline*.

La segmentación de oraciones consiste en delimitar donde empieza y termina cada una. La tokenización consiste en dividir cada texto en unidades básicas, que suelen ser palabras. La normalización consiste en aplicar una transformación al texto para facilitar su procesamiento; por ejemplo, eliminando símbolos. Estas etapas se describen con más detalle en la Subsección 4.1.3.

El etiquetado gramatical (*part-of-speech tagging* en inglés, abreviado como *PoS tagging*) consiste en asignarle a cada palabra una categoría según la función que cumple en cada oración. Una palabra se puede categorizar como sujeto, verbo, adjetivo, entre otros; y también puede tener asignadas subcategorías de estos. No debe confundirse con el etiquetado de entidades y relaciones.

El análisis sintáctico de un texto consiste en analizar la estructura de cada oración. La forma más común es con árboles de dependencia, en los que cada nodo es una palabra y cada arista indica que una palabra depende de otra.

Sobre un texto preprocesado se puede aplicar un algoritmo de reconocimiento de entidades nombradas y, una vez encontradas, buscar relaciones entre estas entidades; o se puede tomar un texto ya etiquetado para hacer extracción de relaciones.

En Bundschus et al. [9] los autores trabajan con ambos enfoques y se comparan los resultados obtenidos, que son siempre mejores cuando se parte de un texto etiquetado manualmente, que es lo que se hizo en este trabajo.

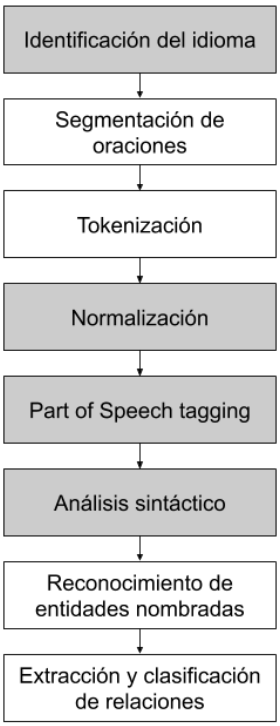

Fig. 1.2: Ejemplo de un *pipeline* de extracción de información. Las etapas en gris son opcionales. Imagen traducida de Cotik [10].

En la literatura se pueden encontrar varios métodos para reconocimiento de relaciones. Se pueden dividir en dos grandes grupos:

- Métodos basados en reglas: a partir de un análisis manual de un subconjunto del *dataset*, se crea manualmente un conjunto de reglas. Estas se usan para determinar qué frases contienen entidades relacionadas y cuáles son. Un ejemplo se puede encontrar en Ben Abacha et al. [11].
- Métodos basados en técnicas de aprendizaje automático: a partir de un conjunto de datos de entrenamiento, se entrena un modelo que permite clasificar las frases según sus relaciones. Entre las técnicas de aprendizaje automático utilizadas se encuentran las *support vector machines* [12], las redes neuronales recurrentes [13] y las redes neuronales convolucionales [14, 15]. Estas últimas son las que han obtenido los mejores resultados y las que utilizaremos para uno de los métodos presentados, por lo que las discutiremos con más detalle en la Sección 2.2.

Contribución

En este trabajo nos centraremos en la detección de relaciones en informes de imágenes radiológicas escritos en español, específicamente ecografías. Buscaremos relaciones de tipo $occurs\_in$ entre entidades anatómicas y $findings$. Es decir, dadas las partes del

cuerpo y las posibles patologías y diagnósticos que se encuentran en un informe, queremos encontrar qué patología *ocurre* en qué parte del cuerpo.

Lo novedoso de este trabajo consiste en que se utilizan textos en español y de naturaleza informal, cuando la mayor parte de la investigación se hace en otros idiomas y sobre textos más formales.

Para esto desarrollamos tres métodos. El primer método, que se presenta en la Sección 4.4, está basado en co-ocurrencia de entidades en una misma oración. Detectamos relaciones a partir de la cantidad de entidades de cada tipo. El segundo método, que se presenta en la Sección 4.5, está basado en reglas: elaboramos manualmente a partir del dataset un conjunto de reglas en forma de expresiones regulares que nos permiten detectar relaciones. Finalmente, el tercer método, que se puede ver en la Sección 4.6, está basado en redes neuronales convolucionales.

Es necesario remarcar también que para desarrollar este último método, se entrenaron *word embeddings* con el corpus disponible de textos biomédicos en español. Estos *word embeddings* pueden quedar disponibles para trabajos futuros.

Organización del trabajo

El presente documento está dividido en seis capítulos. En el Capítulo 2, se desarrollan los conceptos necesarios para profundizar en la comprensión del problema y de los métodos propuestos.

En el Capítulo 3, se desarrolla un análisis de las técnicas utilizadas actualmente para resolver problemas relacionados con el que nos ocupa, especialmente los enfoques utilizados para detección de relaciones en español y en otros idiomas.

En el Capítulo 4, se presentan los tres métodos utilizados. Además, se desarrollan los detalles de su implementación y de la experimentación realizada para evaluarlos.

En el Capítulo 5, se presentan los resultados obtenidos en la experimentación por cada uno de los métodos utilizados y se realiza un análisis de los mismos.

Por último, en el Capítulo 6, se desarrollan las conclusiones obtenidas en este trabajo y se plantean trabajos futuros a ser desarrollados en la misma dirección.

2. MARCO TEÓRICO

En este capítulo se desarrollan los conceptos teóricos necesarios para explicar los métodos utilizados.

2.1. Redes Neuronales

Una red neuronal es un modelo computacional inspirado en el funcionamiento del cerebro. Tiene unidades básicas de procesamiento, análogas a las neuronas, y permite codificar información en los "pesos sinápticos" que las unen, como el cerebro lo hace en las conexiones entre las neuronas [16].

Un modelo basado en redes neuronales artificiales *aprende* a aproximar una función a partir de analizar múltiples ejemplos o instancias. El entrenamiento consiste en presentarle al sistema una gran cantidad de instancias, y este varía los pesos de las conexiones entre las distintas capas de modo que se prediga correctamente el comportamiento de dichas instancias. Este proceso permite crear, eventualmente, un modelo que automatice el procesamiento de la tarea en cuestión.

En general es necesario cierto preprocesamiento sobre los datos de entrada que facilite el aprendizaje.

La ventaja principal de las redes neuronales es su capacidad de generalización, que permite encontrar soluciones a problemas sin la necesidad de desarrollar una técnica manual.

Además, el costo computacional se concentra en el entrenamiento. Una vez entrenada, una red ofrece buena velocidad al momento de usar datos nuevos. Es posible reentrenar con datos nuevos una red ya entrenada, con el objetivo de mejorar la generalización o de adaptarla a otro dominio.

Perceptrón simple

La forma más simple de red neuronal es el perceptrón simple. En una iteración este recibe una entrada, dándole un peso a cada valor que la compone, suma los valores ponderados de la entrada más un *bias*, que actúa como la constante de una función lineal, y devuelve un output correspondiente al resultado de una función de activación (ver Sección 2.1). Luego se ajustan los pesos según una función de pérdida que compara la salida obtenida y la esperada. Un ejemplo se puede ver en la Figura 2.1.

Un perceptrón simple solamente puede aproximar funciones linealmente separables.

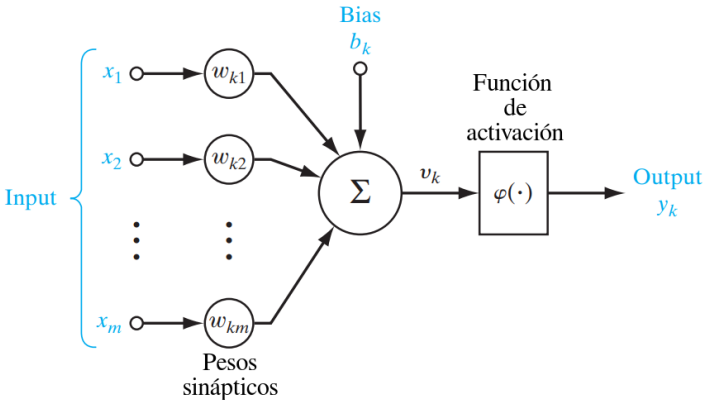

Fig. 2.1: Ejemplo de perceptrón simple, al que llamaremos k. x_1, x_2 ... x_m son los valores de entrada. Los w_{ki}, son los pesos asociados a la entrada x_i. b_k es el *bias*. La salida y_k es el resultado de aplicar la función de activación φ a la suma (Σ) ponderada de los pesos y el *bias*. Imagen tomada de Haykin [16].

Función de activación

La función de activación $\varphi(v)$ determina la salida de un nodo a partir de los valores ponderados de la entrada:

$\varphi(v) = \varphi(\sum_{i=1}^{m} x_i w_{ki})$

Algunas funciones de activación que se utilizan son:

Escalón: $\varphi(v) = \begin{cases} 0 & \text{si} \quad v < 0 \\ 1 & \text{si} \quad v \geq 0 \end{cases}$

Sigmoidea: $\varphi(v) = \dfrac{1}{1 + e^{-av}}$ con a una constante real positiva.

ReLU (*Rectified Linear Unit*): $\varphi(v) = \begin{cases} 0 & \text{si} \quad v < 0 \\ v & \text{si} \quad v \geq 0 \end{cases}$

Es importante que no sean lineales, ya que eso implicaría que la red solo podría aproximar funciones linealmente separables.

Perceptrón multicapa

Los perceptrones multicapa incluyen una o más capas "ocultas" entre la entrada y la salida. Cuando cada nodo de una capa toma como entrada toda la salida de la capa anterior, se habla de una **capa densa**.

Las capas ocultas pueden tener cualquier dimensión. Se conoce como arquitectura de una red a la cantidad y las dimensiones de las capas de entrada y salida y las capas intermedias, así como las funciones de activación utilizadas.

El uso de capas ocultas permite en la teoría aproximar cualquier función continua.

En la práctica, el poder encontrar los hiperparámetros[1] adecuados para poder hacerlo no es un problema trivial.

Descenso por gradiente

En un modelo basado en redes neuronales se define una función de costo o pérdida a partir de la salida obtenida y la deseada. El modelo busca minimizar el valor de esa función.

El gradiente es una generalización a funciones $R^n \to \mathrm{R}$ del concepto de derivada. El descenso por gradiente es un método para encontrar el mínimo de una función siguiendo la dirección en la que la pendiente dada por este indica un mayor decrecimiento. Este método se utiliza para encontrar los parámetros que minimicen la función de costo.

El método de *backpropagation* [17] utiliza la regla de la cadena para adaptar los pesos de cada capa según el resultado de la capa siguiente, de forma que se avance en la dirección en la que la función de error se reduce más rápido.

La magnitud del ajuste que se haga a los pesos depende de la *tasa de aprendizaje*. Para definir un valor adecuado y adaptarlo en cada iteración existen diversos optimizadores. En nuestra implementación se utiliza el optimizador Adam [18], que tiene un valor distinto para cada parámetro y se basa en ponderar el gradiente actual con los obtenidos en iteraciones anteriores.

Dropout

El $overfitting$ es un problema recurrente en en las técnicas de aprendizaje automático, en particular en las redes neuronales. Esto ocurre cuando un modelo se ajusta mucho a los datos de entrenamiento que recibe como entrada, y el resto de ellos no tiene características similares. Como consecuencia, el modelo obtiene buenos resultados para los datos de entrenamiento pero falla al intentar aplicarlo a datos nuevos.

Una de las causas posibles del $overfitting$ es que el aprendizaje se concentre en pocas neuronas. Por ejemplo, puede pasar que solo una parte de la entrada se tome en cuenta y el resto termine con pesos muy bajos. Estos modelos son muy susceptibles a cambios en los datos de entrada. Otra posible causa es que se haga una cantidad excesiva de iteraciones.

Para solucionar este problema se puede agregar una capa de *dropout*, que desactiva (asignando cero a la salida) a un conjunto aleatorio de los nodos de la capa anterior. Este conjunto varía en cada iteración del aprendizaje. Esto se puede ver en la Figura 2.2.

Cada capa de *dropout* tiene un hiperparámetro que representa la proporción de nodos a desactivar en cada iteración. Esto solo se hace durante el entrenamiento; durante la evaluación no se desactivan nodos.

[1] Se llama **hiperparámetros** a los valores de la red que se definen independientemente del proceso de entrenamiento.

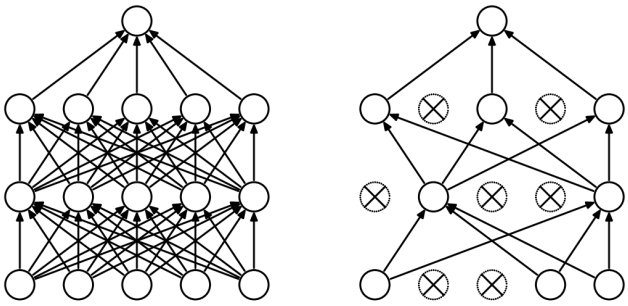

Fig. 2.2: Izquierda: red neuronal sin *dropout*. Derecha: Resultado de aplicar *dropout*. Imagen tomada de Srivastava et al. [19].

2.2. Redes Neuronales Convolucionales

Una red neuronal convolucional (CNN) es una red neuronal que tiene al menos una capa convolucional. Las redes neuronales convolucionales resultan particularmente útiles para el procesamiento de imágenes, debido a que las capas convolucionales aprovechan la distribución espacial de la entrada [20]. Sin embargo, también pueden ser utilizadas para el procesamiento de texto, ya que ambas son entradas secuenciales.

Convolución

La convolución es una operación matemática definida como la integral del producto de dos funciones, donde una es desplazada y reflejada. Una entrada bidimensional puede considerarse como una función discreta de dos variables. En ese caso la convolución puede implementarse como la multiplicación matricial entre distintas submatrices de la entrada y una matriz fija denominada *filtro* o *kernel*.

Formalmente, el resultado de la convolución entre una entrada I bidimensional y un *kernel* K de $n \times m$ es S, donde

$S(x, y) = (I * K)(x, y) = \sum_{i=0}^{n} \sum_{j=0}^{m} I(i, j)K(x - i + n/2, y - j + m/2)$

Esto se puede ver gráficamente en la Figura 2.3.

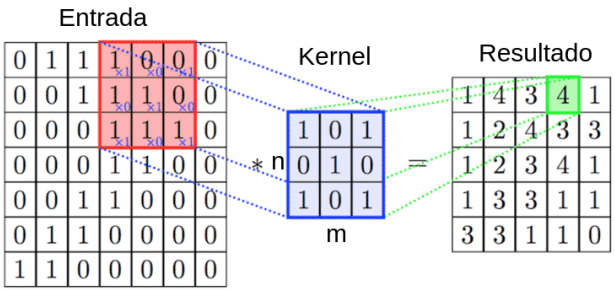

Fig. 2.3: Ejemplo de convolución sobre una entrada bidimensional.

Al tamaño del *kernel* también se lo llama tamaño de ventana. Si es de $n \times n$, la primera posición en la que se puede centrar el *kernel* es la $((n-1)/2, (n-1)/2)$. Es decir que la salida perderá $(n-1)/2$ columnas y filas a cada lado. Para evitar esto se puede incrementar el tamaño de la entrada en $(n-1)/2$ a cada lado, resultando en una salida del mismo tamaño de la entrada. Esto se llama *same padding*, mientras que *valid padding* es usar la entrada sin *padding*.

El *stride* es la cantidad de "pasos" que avanza el *kernel* en cada paso. Es decir que si el *stride* es s y la primera posición del *kernel* es la $((n-1)/2, (n-1)/2)$, la segunda deberá ser $((n-1)/2 + s, (n-1)/2)$. Si no se aclara se considera que es 1.

Las convoluciones tienen muchos usos. Por ejemplo, la aplicación de filtros gaussianos para suavizar una imagen con ruido o la detección de bordes mediante la búsqueda de saltos grandes en el gradiente de la imagen.

Capa convolucional

En una capa convolucional de una red neuronal, los valores del *kernel* no están fijos sino que se obtienen del entrenamiento. Un aprendizaje exitoso resulta en una capa convolucional capaz de detectar $features$ útiles para el objetivo de la red. Los hiperparámetros son el tamaño de la ventana, el *padding* y el *stride*.

En las capas convolucionales, a diferencia de las capas densas, los nodos comparten los mismos pesos, lo cual disminuye significativamente la cantidad de parámetros a aprender. De esta manera, se reduce la complejidad del modelo, así como el tiempo necesario para el entrenamiento. Una comparación entre una capa densa y una capa convolucional se puede ver en la Figura 2.4.

Otra característica de las capas convolucionales es la equivariancia a la traslación: como la convolución es realizada sobre toda la entrada utilizando el mismo filtro, las características resaltadas por dicho filtro son detectadas independientemente de su posición, a diferencia de lo que sucede en las capas densas.

Por lo general en una capa convolucional no se busca entrenar un único filtro. Lo que se hace es aprender una serie de filtros distintos sobre la misma entrada. Si la entrada es bidimensional, se pueden concatenar las salidas de cada filtro en una nueva dimensión, resultando en una salida tridimensional.

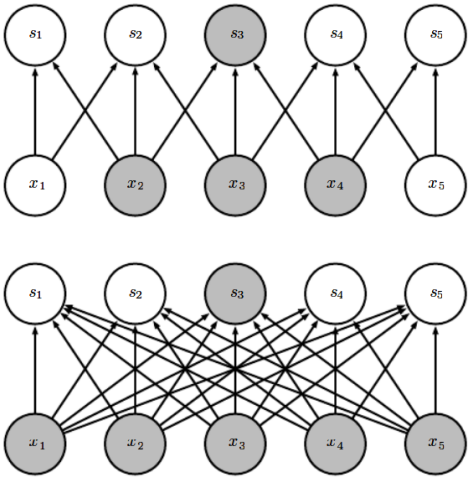

Fig. 2.4: En la imagen de arriba se ve una capa convolucional. El nodo s_3 de la capa s toma como entrada los valores de x_2, x_3 y x_4. El vector de los pesos que asigna a cada valor x_i será el mismo que el de todos los s_i. En la imagen de abajo se ve una capa densa. s_3 recibe información de toda la capa x y tiene pesos distintos a los del resto de los nodos de s. Imagen tomada de Goodfellow y Bengio [21].

Capa de *pooling*

Una capa de agregación o *pooling* toma en cada posición una función estadística de la salida de múltiples neuronas de la capa anterior. En general las capas de *pooling* también hacen *downsampling*, es decir reducen el tamaño de la entrada, tomando la salida de múltiples neuronas para la entrada de una sola neurona de la siguiente. Un ejemplo se puede ver en la Figura 2.5.

El objetivo de una capa de *pooling* es eliminar el ruido en las capas previas tomando un único valor representativo de todo el campo receptivo[2]. Este valor suele ser el máximo o el promedio de la entrada de cada nodo. Luego, las combinaciones más comunes de la capa de pooling son $MaxPooling$ (el máximo) y $AvgPooling$ (el promedio).

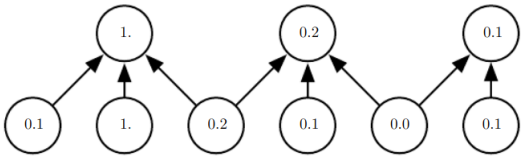

Fig. 2.5: Ejemplo de $MaxPooling$ (con *downsampling*). Los nodos de la capa superior toman el máximo de los valores de su entrada. Imagen tomada de Goodfellow y Bengio [21].

[2] El *campo receptivo* es la región de la entrada que procesa cada neurona. En la Figura 2.4 se puede ver un ejemplo de campo receptivo, marcado en color

Arquitectura de una CNN

Las CNN suelen utilizar capas convolucionales alternadas con capas de *pooling*. Con cada capa aumenta el campo receptivo.

Las primeras capas funcionan como herramientas básicas (por ejemplo, en una imagen pueden detectar bordes y esquinas). Las intermedias detectan características más complejas (formas, figuras), y las finales detectan características particulares del dominio en grandes regiones de la imagen (orejas de gatos y perros en clasificación de animales, por ejemplo)

En general, luego de una serie de capas convolucionales y de *pooling*, la red finaliza con una serie de capas densas, una vez reducida la cantidad de parámetros a un volumen manejable. Sin embargo, existen también redes puramente convolucionales, sin capas densas.

2.3. Word embeddings

Una red neuronal tiene una entrada numérica. En nuestro problema, la entrada son palabras. Se busca entonces una representación para estas palabras que se pueda utilizar como entrada.

Una opción es lo que se llama *one-hot vectors*. Estos consisten en tener un vector que tenga un largo igual a la cantidad total de palabras aceptadas. Para representar la i-ésima palabra, el vector valdrá 1 en la posición i y 0 en todas las demás posiciones. Esta representación tiene la desventaja de que genera vectores demasiado largos y esparsos. Además, no tienen ninguna información sobre el significado o sobre las relaciones entre palabras.

Se denomina *word embedding* a la representación de una palabra como un punto en el espacio, de forma que este punto represente el significado de una palabra. El término *embedding* hace referencia a que la palabra queda *embebida* en un espacio vectorial. Para esto hay varios aspectos a tener en cuenta [4]:

- El **sentido** específico, de todas las acepciones posibles, con el que se usa una palabra en una oración. Por ejemplo, `blanco` puede ser un color o un objetivo.
- La **similaridad** entre palabras, que va desde la ausencia de relación hasta la sinonimia.
- La afinidad (***relatedness***) entre palabras: `vaso` y `agua` son palabras afines, aunque no sean similares.
- **Campos léxicos**: el dominio al que se asocia la palabra. Por ejemplo, `cirujano`, `médico` y `anestesia` son todas palabras relacionadas con hospitales.
- **Connotación**: la relación que tiene el significado con las emociones de quien la escribe.

El modelo computacional actual se basa en teorías lingüísticas que afirman que el significado de una palabra está dado por el **contexto** [22]. De esta forma, se espera que dos palabras similares tengan alrededor de cada aparición contextos parecidos; es decir, con alta probabilidad de tener palabras en común. Como contexto se toma una ventana de n palabras ($n$grama) a cada lado de la palabra original.

En un modelo de *word embeddings*, la similaridad entre dos palabras w_1 y w_2 queda representada por la distancia coseno entre sus vectores. Ésta se calcula como su producto interno dividido por el producto de sus normas:

$$\frac{w_1 \cdot w_2}{||w_1|| \cdot ||w_2||}$$

Además, el desplazamiento que existe entre los vectores de dos palabras contiene información sobre la relación entre dichas palabras. Así, por ejemplo, si $v(w)$ es el vector correspondiente a la palabra w, se espera que se cumpla que $v(rey) - v(hombre) + v(mujer) \approx v(reina)$.

Los algoritmos más populares para generar vectores cortos y densos, aptos para ser utilizados en técnicas de aprendizaje automático, son Skip-Gram y GloVe. En este trabajo utilizaremos el primero.

Skip-Gram, implementado por primera vez por Mikolov et al. [23], se basa en entrenar para cada palabra w un clasificador mediante regresión logística que prediga qué otras palabras son probablemente parte del contexto de w. Los pesos finales del clasificador son la salida del algoritmo.

Para entrenar el clasificador se toman las palabras del contexto cercano como refuerzos positivos y palabras aleatorias del vocabulario como refuerzos negativos.

Este entrenamiento se hace con corpus tan grandes como sea posible, en la escala de millones de palabras. Si el corpus es de un dominio específico, se espera que los vectores resultantes sean útiles para problemas de ese dominio.

También es posible reentrenar vectores ya creados con más textos. Por ejemplo, se puede tomar *word embeddings* de dominio general y reentrenarlos con textos de un dominio específico. Esto puede permitir obtener vectores que conservan la información obtenida en los textos con los que primero se los entrenó, pero adaptados a un dominio particular.

Los *word embeddings* son específicos para cada idioma. Si bien hay palabras con significados similares en español y en inglés, los vectores obtenidos al entrenar con textos en inglés no sirven para problemas sobre textos en español; y la disponibilidad de *word embeddings* en español es mucho menor que en inglés.

2.3.1. Evaluación

Existen varias formas de evaluar *word embeddings*. Se pueden dividir en evaluaciones intrínsecas y extrínsecas. Las evaluaciones **extrínsecas** consisten en comparar contra un estándar los resultados obtenidos por dichos vectores en una tarea de NLP.

Por otro lado, las evaluaciones **intrínsecas** buscan medir directamente en los *word embeddings* la información que contienen sobre las relaciones entre palabras. Algunos

ejemplos de evaluaciones intrínsecas son los test de similaridad y de analogías. En un test de similaridad se calculan los valores obtenidos por un conjunto de pares de palabras con el obtenido promediando las respuestas de anotadores humanos. En un test de analogías, en cambio, se construye una serie de 4-uplas de palabras de la forma

`Brasil` (es a) `Brasilia` (como) `Uruguay` (es a) `Montevideo`.

Para cada 4-upla (A, B, C, D), se calcula $x = v(A) - v(B) + v(C)$ y se busca qué vector es el más cercano en distancia coseno a x. Se considera que esa analogía se resolvió correctamente si y solo si este vector es $v(D)$. Sobre esto se pueden calcular métricas.

2.4. Expresiones regulares

Una expresión regular r define un lenguaje $L(r)$[24]. Para un alfabeto Σ, un lenguaje es un subconjunto de las cadenas de caracteres que se pueden construir utilizando una o más veces sus símbolos. Las expresiones regulares se definen inductivamente de la siguiente forma:

- λ es una expresión regular que representa al conjunto que solo tiene la cadena vacía: $L(\lambda) = \{\lambda\}$
- Para todo $a \in \Sigma$, a es una expresión regular que representa al conjunto que solo tiene una cadena de un símbolo: $L(a) = \{a\}$
- $L(r|s) = L(r) \cup L(s)$
- $L(rs) = L(r)L(s)$
- $L((r)) = L(r)$
- $L(r)^* = (L(r))^*$

$L(r)L(s)$ denota la concatenación de dos lenguajes. El resultado de esto es el lenguaje donde cada palabra está formada por la concatenación de una palabra de $L(r)$ y una palabra de $L(s)$.

El símbolo * en este contexto representa la clausura de Kleene. Esta operación se define para un lenguaje L como $\bigcup_{i=0}^{\infty} L_i$, donde L_i es el resultado de concatenar L consigo mismo i veces. Es decir, $L_0 = \{\lambda\}$ y $L_i = L^{i-1}L$ si $i > 0$. Por ejemplo, para $\Sigma = \{abc\}$ la expresión $(a|b)c$ incluye únicamente las cadenas ac y bc; mientras que $(a|b)^*$ representa un conjunto infinito con todas las cadenas formadas por los caracteres a o b.

Las expresiones regulares se utilizan ampliamente en la actualidad para buscar cadenas de texto o para validar entradas. En este trabajo las utilizaremos para construir reglas que se correspondan con los caracteres que se encuentran entre dos entidades relacionadas.

3. TRABAJOS PREVIOS

Se han realizado muchos trabajos sobre detección y clasificación de relaciones. En esta sección se describen algunos de ellos. Salvo los casos en los que se indica lo contrario, trabajan con textos en inglés; además, las tareas específicas que resuelven no son la misma que la de este trabajo, pero algunas técnicas que utilizan siguen resultando útiles.

Una tarea sobre la que hay varios trabajos realizados es el análisis de la relación entre dos medicamentos (*Drug-drug interaction*, DDI). En este caso hay cuatro tipos de relación predefinidos ($advice$, $mechanism$, $effect$, int), más un tipo "negativo" para indicar que no hay relación.

Otras tareas comunes son la búsqueda de relaciones entre proteínas (*Protein-Protein interaction*, PPI), entre genes y proteínas, entre enfermedades y tratamientos y entre medicamentos y efectos adversos.

A continuación se presentan las principales técnicas usadas por los trabajos de detección de relaciones y se mencionan algunos trabajos que las utilizan.

3.1. Métodos basados en co-ocurrencia

Los métodos basados en co-ocurrencia se focalizan en si dos entidades aparecen juntas en una misma oración. No incluyen información sobre el contenido del texto, utilizando principalmente métodos estadísticos para saber si una co-ocurrencia implica una relación o no.

El método planteado por Duque et al. [25] consiste en hacer un análisis estadístico sobre la proporción en la que aparece cada par de entidades en la misma oración sobre las apariciones totales de cada una. Si pasa cierto umbral, se asume que las dos entidades estarán relacionadas. Este método es el único de la tabla que tiene un F1[1] menor a 0.6.

3.2. Métodos basados en reglas

En los métodos basados en reglas, se realiza un análisis manual de un conjunto de oraciones, con el objetivo de identificar cómo son aquellas que incluyen una relación. Esto se plasma en una serie de reglas; se asume que aquellas oraciones que las cumplen contienen una relación.

Kabiljo et al. trabajan en la interacción entre genes y proteínas (GPI) [26]. El método consiste en considerar a un gen relacionado con una proteína si y solo si son mencionados en la misma oración y entre ambos hay una palabra clave que indique interacción. La lista de palabras clave la obtienen de tres proyectos anteriores. Como *baseline*, consideran a un gen relacionado a una proteína si y solo sí están en la misma oración.

[1] Ver Sección 4.2 para una explicación de las métricas utilizadas

Ben Abacha y Zweigenbaum desarrollan un método basado en expresiones regulares [11]. En primer lugar, para cada tipo de relación, buscan en el UMLS Metathesaurus [27] ejemplos de pares de entidades que se relacionan de esa forma.

Para cada uno de estos pares de entidades, buscan en el corpus todas las oraciones en las que ambos ocurran. Con estas oraciones, construyen manualmente expresiones regulares que acepten las palabras que estén entre ambas entidades, que son las que indican que hay relación.

El algoritmo entonces se reduce a chequear, en cada oración que tenga dos entidades que pueden estar relacionadas, si las expresiones regulares generadas aceptan o no las palabras que las unen.

Casillas et. al. [28] trabajan con textos en español, lo que hace que sus métodos sean especialmente relevantes. Buscan relaciones entre medicamentos y efectos adversos. El método basado en reglas que proponen se inicia buscando las distintas formas de la palabra "alergia" y generando patrones con las palabras que aparecen en un contexto inmediato. Esto se generaliza a reglas basadas tipos de palabras.

3.3. Métodos basados en redes neuronales recurrentes (RNN)

Estos métodos se basan en entrenar una red neuronal recurrente para la clasificación de relaciones a partir de instancias generadas con las oraciones del corpus. Para esto es necesario decidir qué características de dichas oraciones y de las palabras que las componen son útiles para este entrenamiento.

Sahu y Anand trabajan con *Long Short Term Memory Networks* (LSTM), un tipo de RNN [13] para hacer DDI. Como entrada toman los *word embeddings* de cada palabra de la frase, concatenados con *embeddings* que representan la distancia en palabras a las dos drogas cuya relación buscan clasificar. Esto es similar a los *positional embeddings* que se describen en la Subsección 4.6.2. La arquitectura de la red consiste en una capa de LSTM, seguida de una capa de pooling, y por último una salida $softmax$.

También se han utilizado LSTM en trabajos en español. En Goenaga [29] utilizan una arquitectura muy similar a la de Sahu y Anand para hacer DDI.

3.4. Métodos basados en redes neuronales convolucionales

Estos métodos son similares a los basados en RNN, pero la red neuronal que se entrena es convolucional.

Nguyen y Grishman usan la misma entrada que en el método de Sahu y Anand para su método basado en CNN [30]. Su modelo tiene una capa convolucional de tamaño de ventana variable, una capa de *pooling*, una capa densa y una salida $softmax$. Utilizan *dropout* antes de la salida.

Este trabajo no fue realizado sobre textos médicos sino sobre un corpus general. Sin embargo, otros autores se basaron en este modelo para tareas con un mayor grado de relación con el dominio biomédico. Wang et. al recurrieron a un modelo similar, aunque prescindiendo de la capa densa, para hacer DDI [15].

Roller et. al. tomaron el modelo de Nguyen y Grishman para clasificación de relaciones en textos médicos en alemán. En su trabajo destacan, además de los resultados, el haberlos obtenido con un corpus pequeño (626 informes de altas y de un departamento de transplante de riñón) [3].

En Asada et. al. [31] no se utilizan *positional embeddings*. En su lugar se recurre a un mecanismo de atención sobre la capa convolucional. Además, se experimenta con una función de *ranking* como alternativa a la salida $softmax$.

El modelo de Peng y Lu [14] utiliza la misma arquitectura básica que el de Nguyen y Grishman, pero tiene un nivel de complejidad mucho mayor en la entrada. En primer lugar, construyen un árbol de dependencia para cada oración. Para cada palabra, además de los *word embeddings*, incluyen información de su *PoS_tagging*, *chunking*,[2] cuál es su padre en el árbol de dependencias, y la distancia a otras palabras. Además de todo esto, buscan al padre en el árbol de dependencias y agregan un vector a la matriz de entrada con las mismas características que para el hijo. Es decir que si una palabra tiene muchos hijos, se repetirá muchas veces en la entrada.

Este método también se utiliza para textos en español. En Suárez-Paniagua [32] se utilizan redes neuronales convolucionales para hacer DDI.

3.5. Otros métodos

Muzaffar et. al. [12] utilizan dos algoritmos de clasficación: *Support Vector Machines* y *Naïve Bayes*. Como entrada utilizan un conjunto de características de cada palabra (lema, *PoS tagging*), junto a información del *chunk*. Además, para cada *chunk* incluye una lista de los conceptos más comunes con los que suele estar relacionado, obtenida de MetaMap [33].

En el trabajo ya mencionado de Casillas et al. [28] se propone otro método, basado en un clasificador *Random Forest*. Las $features$ utilizadas incluyen las palabras en un contexto de cada entidad, su forma lematizada, su $PoS\_Tagging$, si está negada y la distancia entre las entidades.

3.6. Resumen

Para los trabajos mencionados se muestra en la Tabla 3.1 el título, los autores, el año de publicación, la tarea que resuelven, los resultados obtenidos y el corpus que utilizaron.

[2] El *chunking* es la búsqueda de frases cortas dentro de una oración, con más información que las palabras sueltas pero sin llegar a construir un árbol de dependencias completo.

Título	Autores	Año	Tarea	Método	Resultados			Corpus
					P	R	F1	
A realistic assessment of methods for extracting gene/protein interactions from free text [26]	Kabiljo, Clegg, Shepherd	2009	Interacción entre genes y proteinas (GPI)	Coocurrencia con *keywords*	0.529	0.720	0.716	5 corpus, hasta 1100 oraciones cada uno
Automatic extraction of semantic relations between medical entities: a rule based approach [11]	Ben Abacha, Zweigenbaum	2011	RE en textos médicos	Expresiones regulares	0.757	0.604	0.672	580 oraciones de PubMed
Unsupervised extraction of adverse drug reaction relationships [25]	Duque et al.	2015	Medicamentos vs efectos adversos	Análisis estadístico de co-ocurrencia	0.451	0.603	0.516	Corpus ADE: 6800 relaciones a partir de 2972 abstracts de MEDLINE
Relation extraction: perspective from convolutional neural networks. [30]	Nguyen, Grishman	2015	RE en textos de dominio general	CNN	0.713	0.539	0.613	ACE 2005, SemEval 2010
A Relation Extraction Framework for Biomedical Text Using Hybrid Feature Set [12]	Muzafar et al.	2015	Enfermedad-tratamiento	Support Vector Machine + Naïve Bayes	-	-	0.935	3495 oraciones de MEDLINE 2001
Drug-Drug Interaction Extraction via Convolutional Neural Networks [15]	Wang et al.	2016	DDI	CNN	0.757	0.647	0.696	730 documentos de DrugBank y 175 abstracts de MEDLINE sobre DDI
Extracting Drug-Drug Interactions with Attention CNNs [31]	Asada et al.	2017	DDI	CNN con atención	0.763	0.633	0.691	900 documentos de MEDLINE + DrugBank
Deep learning for extracting protein-protein interactions from biomedical literature [14]	Peng, Lu	2017	Interacción entre proteínas (PPI)	CNN	0.627	0.682	0.653	3000 oraciones de AIMed y BioInfer
Drug-drug interaction extraction from biomedical texts using long short-term memory network [13]	Sahu, Anand	2017	DDI	LSTM	0.678	0.668	0.673	905 documentos de MEDLINE y DrugBank
Detecting Named Entities and Relations in German Clinical Reports [3]	Roller et al.	2018	NER y RE en informes médicos en alemán	CNN (y SVN para comparar)	0.796	0.845	0.806	626 informes clínicos
Clinical text mining for efficient extraction of drug-allergy reactions [28]	Casillas et al.	2016	Medicamentos vs efectos adversos en español	Expresiones Regulares y Random Forest	0.900	0.870	0.880	370 informes de hospital

Tab. 3.1: Performance de trabajos previos sobre detección y clasificación de relaciones en textos biomédicos. En caso de que haya varios resultados reportados, se muestra en esta tabla el que corresponde a la configuración con mejor F1 (ver Sección 4.2 para la definición de Precisión (P), *Recall* (R) y F1).

4. METODOLOGÍA

En este capítulo se describirán los métodos desarrollados en este trabajo. Además se incluyen los detalles de implementación y de experimentación.

En primer lugar se describen varios aspectos del corpus utilizado; se incluyen datos estadísticos y de formato, así como el preprocesamiento necesario. Luego se describen las métricas utilizadas y los recursos externos en los que se basó la experimentación. A continuación se desarrollan los tres métodos que se presentan en este trabajo: un método basado en co-ocurrencia, un método basado en reglas y un método basado en redes neuronales convolucionales.

4.1. Datos

Para este trabajo se cuenta con un corpus de informes anonimizados de ecografías, realizados en el Hospital de Pediatría Dr. Juan P. Garrahan.[1] En esta sección se describen distintos aspectos del conjunto de datos utilizado.

4.1.1. *Dataset*

El corpus disponible para este trabajo consiste en alrededor de 80000 informes. De estos, se dispone de 513 que fueron anotados manualmente. Estos fueron utilizados para el entrenamiento y *testing* de los métodos propuestos.

Los detalles del proceso de anotación se encuentran en Cotik et al. [34]. Este conjunto de datos fue utilizado en trabajos anteriores para detección de negaciones [35] y para el desarrollo de parte de un pipeline de extracción de información como el que se describe en el Capítulo 1 [10].

El formato de los datos genera una serie de desafíos adicionales para extraer información.

En primer lugar, la segmentación de oraciones (ver Subsección 4.1.3) había sido realizada en forma automática. La falta de signos de puntuación lleva a que oraciones que deberían estar separadas se reconozcan como una sola. Por ejemplo, tenemos el siguiente fragmento, que la segmentación reconoció como una sola oración:

```
Ambos ovarios y utero de caracteristicas ecograficas normales. Utero
AVF, tipo postpuberal, con endometrio engrosado ( 2.5cm ) Dimensiones:
Ovario derecho: 4.3 ( cm ) x 2.8 ( cm ) x 5.3 ( cm ) Volumen: 35 ( cc )
Ovario izquierdo: 4.8 ( cm ) x 2.6 ( cm ) x 5.5 ( cm ) Volumen: 37 ( cc )
Utero: diametro longitudinal 7.6 ( cm ) , anteroposterior 5.3 ( cm ) ,
transversal 5.2 ( cm ) Ambos rinones de ecoestructura conservada, con
leve ectasia pielica izquierda 0.7 ( cm ) .
```

[1] http://www.garrahan.gov.ar/, consultado en agosto de 2020.

En segundo lugar tenemos casos donde una sola palabra es dividida y recibe dos etiquetas. El ejemplo más común es `alitiasica`, en donde el prefijo `a` está etiquetado como una negación, y el resto de la palabra como una entidad anatómica. Esto es un problema cuando los algoritmos desarrollados buscan relaciones entre palabras completas.

Por otro lado, tenemos casos donde una entidad está relacionada con varias entidades de otro tipo en una misma oración, pero en los que se podría argumentar que en realidad se trata de una sola entidad compuesta. Por ejemplo, una frase que aparece varias veces es `Arteria hepatica, venas porta y suprahepaticas sin alteraciones`.

El conjunto de datos fue dividido aleatoriamente en dos partes: el conjunto de desarrollo, con 410 informes, y el conjunto de test, con 103. El conjunto de test fue definido antes de iniciar el trabajo y solamente se utilizó para evaluar los algoritmos una vez que estos estuvieron en su versión final.

Un subconjunto de 103 informes del conjunto de desarrollo se tomó como conjunto de análisis. Este se usó inicialmente para buscar patrones que permitieran construir reglas en los primeros dos algoritmos, si bien para el segundo se terminó utilizando todo el conjunto de desarrollo.

La partición descripta se puede ver en la Figura 4.1.

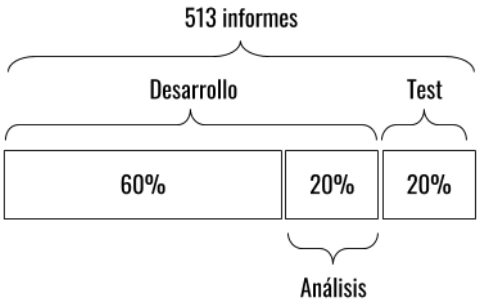

Fig. 4.1: Partición del conjunto de datos.

4.1.2. Formato de los datos

La herramienta utilizada para anotar y visualizar entidades y relaciones es `brat`.[2] Esta herramienta no modifica archivos de texto, sino que genera para cada uno un archivo con el mismo nombre y extensión `.ann`. El formato del archivo consiste en una línea para cada anotación registrada, que puede ser de varios tipos. Para mostrarlo se utilizará el siguiente informe, de una sola oración:[3]

```
se visualiza timo de estructura normal, con lobulo izquiedo prominente
```

[2] `brat` es un acrónimo recursivo para *brat rapid annotation tool*. Fue presentado en Stenetorp et al. [36]. Ver https://brat.nlplab.org/standoff.html (último acceso en julio de 2020) para más detalles del formato de anotación.

[3] El error de tipeo está en el texto original.

La anotación de este informe contiene la siguiente línea, que indica que entre los caracteres 20 (inclusive) y 24 (no inclusive) del texto hay una entidad que se va a etiquetar como "Anatomical_Entity". A esta entidad se le asigna el identificador T1 para referenciarla en otras anotaciones; por ejemplo, relaciones.

```
T1 Anatomical_Entity 20 24 timo
```

También hay relaciones. Por ejemplo, esta línea indica que entre las entidades T2 y T1 hay una relación de tipo `occurs_in`.

```
R1 occurs_in arg1:T2 arg2:T1
```

Por otro lado, las líneas que empiezan con # son comentarios.

Un ejemplo de oración anotada puede verse en la Figura 4.2. Dicha anotación queda representada de la siguiente forma:

```
T1  Anatomical_Entity   0 37    Ecoestructura parenquimatosa
                                cerebral
T2  Anatomical_Entity   43 61   nucleos de la base
T3  Neg   62 65   sin
T4  Finding 66 78   alteraciones
R1  occurs_in   Arg1:T4 Arg2:T1
R2  occurs_in   Arg1:T4 Arg2:T2
R3  negates Arg1:T3 Arg2:T4
```

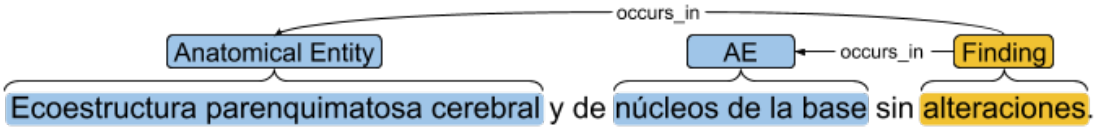

Fig. 4.2: Ejemplo de oración anotada con la herramienta `brat`.

Al procesar archivos con este formato hay información relevante que es necesario obtener para cada palabra. Interesa saber cómo está etiquetada: si es una entidad (o parte de una entidad), y de qué tipo.

También es necesario saber, para cada relación, qué entidades se relacionan, de qué tipo son y qué palabras las componen. Esta información además debe estar separada por oraciones.

4.1.3. Preprocesamiento

Los informes utilizados ya tenían un cierto grado de preprocesamiento:

- **Normalización**: las tildes fueron eliminadas y las letras *ñ* fueron reemplazadas por *n*. Esto puede resultar en pérdida de información. Por ejemplo, una palabra puede cambiar tipo al cambiar su acentuación: *médico* es un sustantivo, mientras que *medico* es la primera persona del presente del verbo *medicar*. Esto se hizo porque ya habia casos de reemplazo de tildes y de *ñ* por *n* en los informes originales.
- **Segmentación de oraciones**: Consiste en dividir cada informe en oraciones. Esto no es trivial, ya que el punto, además de ser el símbolo que se utiliza para indicar el final de una oración, tiene otras funciones, como indicar abreviaturas. Además, como se mencionó previamente, en textos informales como los que forman el corpus de este trabajo pueden faltar signos de puntuación. Los informes del corpus utilizado habían sido segmentados automáticamente.

Por otra parte, para utilizar estos informes resultó necesario pasar el texto a minúscula, ya que dos de los algoritmos utilizados (co-ocurrencia y regex) consideraban dos expresiones como diferentes si sólo se diferenciaban en el uso de mayúsculas.

Finalmente, el último método requiere que la oración que se utiliza como entrada tenga delimitadas las palabras, proceso que se conoce como ***tokenización***. Al igual que la segmentación de oraciones, no es un proceso trivial. Entre otras cosas, es necesario separar las palabras de los signos de puntuación. Para realizarlo se utilizó la función *word_tokenize* de la librería NLTK, mencionada en la Sección 4.3.

4.1.4. Entidades y relaciones de interés

El esquema de anotación del corpus contempla las entidades nombradas que se muestran en la Tabla 4.1:

Tipo de etiqueta	**Etiqueta**	**Abv.**
Entidad	Anatómica	AE
	Hallazgo	FI
	Localización	LO
	Medida	ME
	Textura	TE
	Tipo de medida	TM
Modificador	Negación	NG
	Incertidumbre	UT
Otro	Abreviatura	
	Expresión temporal	
	Término de varios nombres	

Tab. 4.1: Etiquetas utilizadas en el conjunto de datos anotado, junto con las abreviaturas que utilizaremos en este trabajo para referirnos a ellas.

Además, se encuentran anotadas las relaciones indicadas en la Tabla 4.2:

Relación	Entidades relacionadas
occurs_in	FI-AE
	FI-LO
located_in	LO-AE
areaof	AE-LO
measure_of	TM-AE
	TM-LO
	TM-FI
	ME-AE
	ME-LO
	ME-FI
has_measure_type	ME-TM
texture	TE-AE
	TE-LO
	TE-FI
negates	NG-FI
speculates	UT-FI
not_present	CT-FI

Tab. 4.2: Relaciones anotadas. En la primera columna se encuentra el nombre de la relación y en la segunda las entidades nombradas unidas por dicha relación.

En este conjunto de datos, cada par de entidades puede estar relacionado entre sí de con una sola forma.[4] Es decir: dadas dos entidades, si están relacionadas, hay un solo tipo de relación posible. Es por esto que decimos que este trabajo se ocupa de **reconocimiento** de relaciones y no de **clasificación** de relaciones.

Estas relaciones son unidireccionales, en cuanto a que, por ejemplo, no es posible decir que una entidad anatómica *ocurre* en un hallazgo. Sin embargo, para los algoritmos que utilizamos puede ser útil saber el orden en la oración de las entidades del par. En el resto del trabajo, si se distingue, por ejemplo, AE-FI y FI-AE, será por el orden relativo de las entidades en la oración.

Este trabajo se centrará en un solo tipo de relación: *occurs_in*; y en tres tipos de entidades, que son los que pueden participar en una relación de este tipo:

- *anatomical entity* (entidad anatómica). Entidades que hacen referencia a partes del cuerpo. Por ejemplo: *hígado*, *páncreas*, *bazo*.
- *finding* (hallazgo). Corresponde a un diagnóstico o a una posible patología encontrada. Por ejemplo: *absceso*, *quiste*.
- *location* (localización). Corresponde a una ubicación dentro del cuerpo o dentro de una entidad anatómica. Por ejemplo: *periférico*, *adyacente*.

[4] Con excepción de la combinación AE-LO, donde hay dos relaciones posibles, que se diferencian en la dirección.

En la Tabla 4.3 se muestra un recuento de estas entidades y de la relación *occurs_in* en el conjunto de datos de análisis.

Tipo de entidad o relación	Cantidad	(distintos)
AE	999	(104)
FI	565	(171)
LO	166	(42)
occurs_in (FI-AE)	86	
occurs_in (AE-FI)	348	
occurs_in (FI-LO)	26	
occurs_in (LO-FI)	27	

Tab. 4.3: Cantidad de entidades y relaciones encontradas en el conjunto de datos de análisis. Se muestra entre paréntesis la cantidad de valores distintos.

En el conjunto de datos hay relaciones etiquetadas entre entidades que se encuentran en oraciones distintas. Sin embargo, como se puede ver en la tabla Tabla 4.4, la mayor parte de las relaciones es entre entidades dentro de la misma oración. En los trabajos previos estudiados los algoritmos de reconocimiento de relaciones se aplican únicamente a aquellas que están dentro de la misma oración. Por lo tanto en este trabajo se tomará la misma decisión.

	FI-AE	FI-LO
Relaciones en la misma oración	406	50
Relaciones entre distintas oraciones	28	3

Tab. 4.4: Clasificación de las relaciones del conjunto de análisis de acuerdo a si se encuentran en la misma oración o no.

4.1.5. Oraciones repetidas

Las frases utilizadas en el conjunto de datos para indicar ausencia de anomalías suelen repetirse mucho. Así se tiene, en el conjunto de datos de desarrollo:

- 166 casos de `via biliar intra y extrahepatica: no dilatada`.
- 153 casos de `vesicula biliar: alitiasica`.
- 150 casos de `no se observo liquido libre en cavidad`.
- 137 casos de `no se detectaron adenomegalias`.

Se decidió hacer dos ejecuciones de los algoritmos: una en que se consideran todas las oraciones y la otra en que sólo se consideran una vez las repetidas. En el contexto de este trabajo, "sin repetidos", implica contar una sola vez cada **oración** distinta. Se considera a dos oraciones como iguales si todas las palabras que la componen lo son y están en el mismo orden.

En la Tabla 4.5 se muestra qué proporción hay de pares relacionados y cómo influye en ese número la decisión de filtrar o no repetidos.

En la Tabla A.1 del apéndice se muestra cuántas veces aparece cada entidad, y en qué proporción como parte de una relación; mientras que en la Tabla B.1 se muestra lo mismo pero considerando palabras en vez de entidades.

Conjunto de datos	Relación	Repetidos	cant. pares relacionados	cant. pares no relacionados	porc. relacionados	porc. no relacionados
Desarrollo \ Análisis			1090	359	75.22 %	24.78 %
Análisis	AE-FI	Sí	350	146	70.56 %	29.44 %
Test			387	167	69.86 %	30.14 %
Total	AE-FI	Sí	1827	672	73.11 %	26.89 %
Desarrollo \ Análisis			78	64	54.93 %	45.07 %
Análisis	LO-FI	Sí	28	28	50.00 %	50.00 %
Test			28	37	43.08 %	56.92 %
Total	LO-FI	Sí	134	129	50.95 %	49.05 %
Desarrollo \ Análisis			463	262	63.86 %	36.14 %
Análisis	AE-FI	No	170	117	59.23 %	40.77 %
Test			173	132	56.72 %	43.28 %
Total	AE-FI	No	806	511	61.20 %	38.80 %
Desarrollo \ Análisis			76	58	56.72 %	43.28 %
Análisis	LO-FI	No	28	27	50.91 %	49.09 %
Test			28	37	43.08 %	56.92 %
Total	LO-FI	No	132	122	51.97 %	48.03 %

Tab. 4.5: Se muestra para cada partición del conjunto de datos y para cada tipo de relación cuántas relaciones y cuantos pares no relacionados hay, y cuál es la proporción entre unos y otros. Se considera un par por cada posible combinación AE-FI o LO-FI dentro de una oración.

4.2. Métricas utilizadas

Para medir la calidad de los algoritmos implementados es necesario comparar la predicción que se obtiene con la realidad. Como en esta área la realidad tiene un componente subjetivo, se utiliza un *standard*, es decir, una anotación que consideramos como el valor real. Se llama *gold standard* si esta anotación fue hecha por expertos, como en este caso, y *silver standard* si fue obtenida automáticamente.

Para cada par de entidades de interés, entonces, se obtiene por medio de la ejecución de los distintos algoritmos una predicción (entidades relacionadas o no) que se compara

con la anotación previa. Según el valor obtenido y el esperado el resultado puede clasificarse como verdadero positivo, verdadero negativo, falso positivo o falso negativo de acuerdo a la Tabla 4.6.

		Valor real	
		Positivo (hay relación)	Negativo (no hay relación)
Predicción	Positivo (hay relación)	TP	FP
	Negativo (no hay relación)	FN	TN

Tab. 4.6: Matriz de confusión.

Es decir que los **Verdaderos positivos (TP)** son los pares de entidades que estaban anotados como relacionados, y que el algoritmo también detectó como relacionados. Los **Falsos positivos (FP)** son los pares de entidades que el algoritmo detectó como relacionados, pero que **no** estaban anotados como relacionados. Los **Falsos negativos (FN)** son los pares de entidades que estaba anotados como relacionados, pero el algoritmo **no** detectó como relacionados. Por lo indicado en la sección anterior, se excluyen las relaciones entre palabras en diferentes oraciones.

A partir de esto se definen las métricas a utilizar para evaluar los algoritmos:

Precisión: de todos los pares clasificados por el algoritmo como relacionados, qué proporción efectivamente lo está.

$$Precision = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Recall: de todos los pares de entidades relacionados, qué proporción el algoritmo clasificó correctamente como relacionados.

$$Recall = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

F1-score: la media armónica entre precisión y *recall*.

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

En distintos contextos puede resultar más importante la precisión o el *recall*. En este trabajo consideramos que ambos lo son por igual, por lo que se informarán las dos métricas y se utilizará el $F1$ para comparar los resultados de distintos algoritmos o del mismo algoritmo con distintos parámetros.

4.3. Recursos utilizados

Todos los modelos, experimentos y herramientas utilizados en este trabajo fueron desarrollados en Python 3. Para los métodos basados en redes neuronales se utilizó Keras sobre TensorFlow 2.2.0.

Para trabajar con *word embeddings* se utilizó GenSim [37]. Para entrenarlos, fastText [38]. Para *tokenizar* se utilizó la librería Natural Language Toolkit (NLTK) [39].

4.4. Método basado en co-ocurrencia

En esta y las siguientes secciones se desarrollan los tres métodos desarrollados para este trabajo. En esta sección se desarrolla un método basado en co-ocurrencia, en la Sección 4.5, un método basado en reglas y en la Sección 4.6, un método basado en redes neuronales convolucionales.

Como primera aproximación a un algoritmo de detección de relaciones se puede asumir que dos entidades están relacionadas si y solo si están en la misma oración. Sobre esto se puede hacer un pequeño ajuste; este método consistió en buscar inferir las relaciones a partir de la cantidad de entidades de cada tipo en cada oración.

Este análisis fue realizado por separado para relaciones entre *Finding* y *Anatomical Entity* (Subsección 4.4.1) y para relaciones entre *Finding* y *Location* (Subsección 4.4.2).

Este método fue desarrollado para usar como *baseline* contra el que comparar los otros dos.

4.4.1. Análisis de relaciones FI-AE

En primer lugar se contó, en el conjunto de datos de análisis, para cada oración en la que hubiera al menos una relación cuántas entidades de cada tipo había, y cuántas relaciones.

Considerando que los casos con más de dos AE o FI son menos del 18 % del total, es posible agrupar los datos obtenidos de la forma que se indica en la Tabla 4.7.

AE	FI	# Relaciones	# Oraciones
1	1	0	15
		1	183
		2	1
2	1	0	5
		1	13
		2	41
1	2	0	1
		1	4
		2	9
2	2	0	1
		1	1
		2	5
AE < FI		0	3
		1	4
		2	2
		3	2
		4	3
AE ≥ FI		0	12
		1	11
		2	9
		3	6
		4	5

Tab. 4.7: Cantidad de oraciones que tienen cada combinación observada de número de AE, número de FI y número de relaciones. Por ejemplo, hay 183 oraciones que tienen exactamente 1 AE, 1 FI y una relación entre AE y FI.

Cada bloque de filas de la tabla indica cuántas oraciones hay que tienen determinada cantidad de relaciones entre cierto número de AE y FI. Si se observa el segundo bloque, por ejemplo, se puede ver que cuando hay dos AE y un FI, en 41 casos (69,5 % de las oraciones que tienen esa cantidad de AE y FI) hay dos relaciones. Entonces se busca que el algoritmo prediga dos relaciones siempre que haya dos AE y un FI.

De este modo, el algoritmo sigue la lógica de asumir el caso más frecuente para cada bloque. Las reglas resultantes son:

- Si en una oración hay una AE y un FI, asumimos que están relacionados (ver ejemplo en la Figura 4.3).
- Si hay dos AE y un FI, asumimos que ambos AE están relacionados con el FI (ver ejemplo en la Figura 4.4).
- Si hay dos FI y un AE, asumimos que ambos FI están relacionados con el AE (ver ejemplo en la Figura 4.5).

- Si hay dos FI y dos AE, asumimos que hay dos relaciones. Analizando los cinco casos la conclusión fue que el más habitual es que el primer AE se relaciona con el primer FI y el segundo AE con el segundo FI (ver ejemplo en la Figura 4.6).
- Si hay mayor cantidad de FI que de AE, asumimos que el primer FI se relaciona con el primer AE.
- Si hay mayor o igual cantidad de AE que de FI, o la misma cantidad, asumimos que el primer FI se relaciona con el primer AE.
- Si no hay AE o no hay FI, no hay relación porque no hay nada que relacionar.

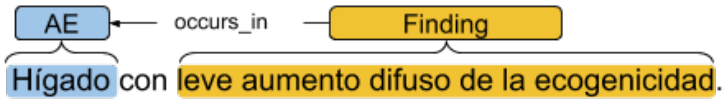

Fig. 4.3: Ejemplo de oración con un AE y un FI.

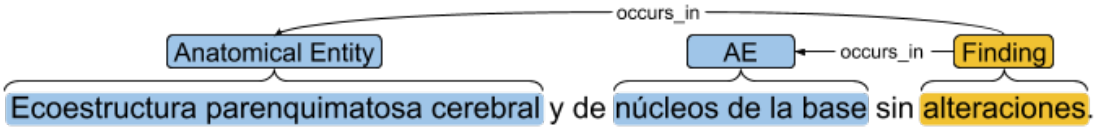

Fig. 4.4: Ejemplo de oración con dos AE y un FI.

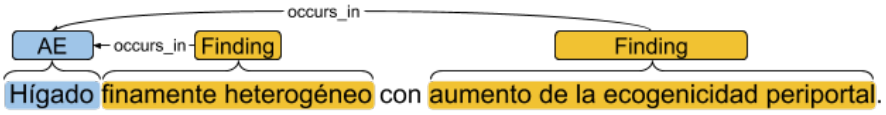

Fig. 4.5: Ejemplo de oración con un AE y dos FI.

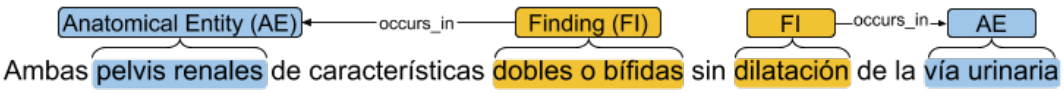

Fig. 4.6: Ejemplo de oración con dos AE y dos FI.

4.4.2. Análisis de relaciones FI-LO

En primer lugar se realizó el mismo análisis que para las relaciones AE-FI, obteniendo los resultados que se muestran en la Tabla 4.8:

LO	FI	# Relaciones	# Oraciones
1	1	0	15
		1	18
2	1	0	0
		1	0
		2	1
1	2	0	1
		1	8
		2	2
2	2	0	0
		1	1
		2	3
LO < FI		0	2
		1	2
		2	1
		3	1
LO ≥ FI		0	3
		1	0
		2	0
		3	1

Tab. 4.8: Cantidad de oraciones que tienen cada combinación observada de número de AE, número de LO y número de relaciones.

En este caso, la distribución de oraciones en cada bloque de la tabla es más uniforme que en el anterior. Luego, la tabla por sí sola no representa una gran cantidad de información para el algoritmo.

A continuación entonces se analizó si había una relación entre el *finding* y la *location*, a través de una AE. Es decir, que el finding estuviera relacionado con una AE, y la AE con una location. Una relación entre una AE y una location puede ser de dos tipos: *located_in* o *area_of*.

Cuando hay una relación *located_in*, como en la Figura 4.7, lo que se ve es que el *finding* no está relacionado con la *location*. La vejiga tiene finos ecos internos, no las paredes. Hay 19 casos en el conjunto de datos de análisis.

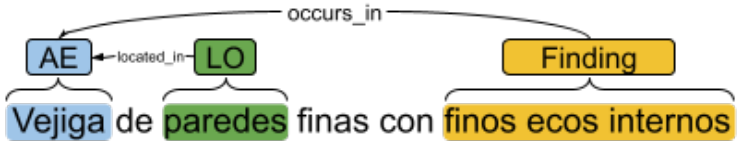

Fig. 4.7: Ejemplo de relación *located_in* entre AE y LO.

Cuando hay una relación *area_of*, como en la Figura 4.8, no está tan claro, pero hay sólo cuatro instancias de esta relación en todo el conjunto de análisis.

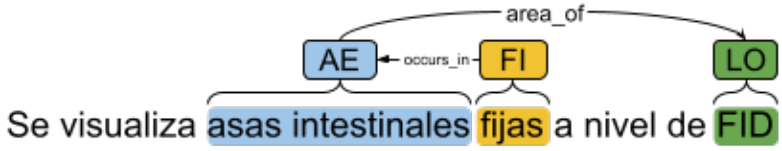

Fig. 4.8: Ejemplo de relación *area_of* entre AE y LO.

Al no obtener nada concluyente de este análisis de relaciones indirectas se decidió para este método aplicar el mismo algoritmo que el utilizado para el otro tipo de relación. Es decir, buscar el caso más frecuente para cada bloque de la Tabla 4.8 y extenderlo a todas las oraciones que tienen esa cantidad de relaciones de cada tipo.

4.5. Métodos basados en reglas

Este método se basa en buscar las expresiones que separan dos entidades que aparecen en una misma oración. Por ejemplo, si se encuentra "LO se observa FI" se puede deducir que el FI está relacionado con la LO.

Por ejemplo, en la frase de la Figura 4.3 (`higado con ligero aumento difuso de la ecogenicidad`) encontramos una expresión de la forma `AE con FI`, donde el conector es " `con` ".

Para todo el conjunto de desarrollo,[5] entonces, se buscó para cada relación *occurs_in* qué expresiones separaban a las entidades, a las que llamaremos "conectores". Luego, para cada conector, contamos cuántas veces aparece entre dos entidades relacionadas, y cuántas veces aparece fuera de una relación.

En base a esto se generaron cuatro tablas (Tabla 4.9, Tabla 4.10, Tabla 4.11, Tabla 4.12), porque se distingue según el tipo de entidad involucrada y el orden en el que se encuentran en la oración.

Para cada tabla, se construyeron expresiones regulares (ver Sección 2.4) que reconocieran a todos los conectores que aparecen más de dos veces y al menos en un 50 % de los casos entre entidades relacionadas. El método entonces asume que dos entidades están relacionadas si y solo si la frase que las separa es reconocida por la expresión regular.

Relaciones entre FI y AE, con la AE apareciendo antes en la oración (AE-FI)

Se muestra a continuación, en la Tabla 4.9, para cada conector que encontramos entre un *finding* y una entidad anatómica, cuántos casos hay en los que el *finding* y la entidad anatómica están relacionados.

[5] El método se desarrolló inicialmente para el conjunto de análisis pero, como la dificultad de construir las reglas no aumentaba demasiado con el tamaño del conjunto de textos y los textos eran homogéneos, se decidió utilizar el conjunto de desarrollo completo. Esto permitió tener reglas más abarcativas.

conector	forman relación		no forman rel.		% que forma relación	
: no	326	(3)	9	(1)	97.31 %	(75.00 %)
: a	153	(1)	20	(0)	88.44 %	(100.00 %)
: sin	145	(3)	0	(0)	100.00 %	(100.00 %)
:	84	(31)	10	(5)	89.36 %	(86.11 %)
␣	84	(66)	9	(4)	90.32 %	(94.29 %)
λ	6	(5)	81	(51)	6.90 %	(8.93 %)
no	53	(9)	0	(0)	100.00 %	(100.00 %)
intra y extrahepatica: no	8	(1)	39	(1)	17.02 %	(50.00 %)
sin	34	(10)	4	(1)	89.47 %	(90.91 %)
con	30	(29)	4	(2)	88.24 %	(93.55 %)
y extrahepatica: no	13	(3)	3	(0)	81.25 %	(100.00 %)
de ecoestructura conservada, sin	14	(1)	2	(1)	87.50 %	(50.00 %)
se observa	10	(10)	5	(3)	66.67 %	(76.92 %)
a	12	(1)	3	(0)	80.00 %	(100.00 %)
, venas porta y suprahepaticas sin	11	(1)	0	(0)	100.00 %	(100.00 %)
intra y extrahepatica no	3	(2)	7	(1)	30.00 %	(66.67 %)
: leve	6	(3)	2	(1)	75.00 %	(75.00 %)
de	8	(8)	0	(0)	100.00 %	(100.00 %)
y suprahepaticas sin	6	(1)	0	(0)	100.00 %	(100.00 %)
y extrahepatica no	4	(2)	1	(1)	80.00 %	(66.67 %)
y de nucleos de la base sin	4	(1)	1	(1)	80.00 %	(50.00 %)
,	3	(2)	1	(1)	75.00 %	(66.67 %)
de paredes finas y	3	(1)	0	(0)	100.00 %	(100.00 %)
(Conectores que aparecen 2 veces)	28	(27)	0	(0)	100.00 %	(100.00 %)
(Conectores que aparecen 1 vez)	210	(210)	9	(9)	95.89 %	(95.89 %)
Total	1257	(429)	211	(82)	85.62 %	(83.95 %)

Tab. 4.9: Para cada conector que separa un *finding* y una entidad anatómica, se muestran cuántos casos hay en los que el *finding* y la entidad anatómica tienen una relación *occurs_in* en el dataset de desarrollo. Se muestran entre paréntesis los casos únicos (en los que coinciden tanto el *finding* como la entidad anatómica). Se muestran sólo los que aparecen al menos 3 veces.

A partir de esta tabla se construyó una expresión regular que acepte conectores si y solo si unen pares de entidades relacionados más de un 50 % de las veces. Para los conectores que aparecen menos de tres veces es indistinto si se aceptan o no. Por ejemplo, se busca que "`: no`" se acepte, ya que une pares relacionados en un 97,31 % de los casos, pero λ no, ya que solamente lo hace en un 6,9 % de los casos.

La expresión regular es la siguiente:

```
AE (textura)? :? (NEG|AFIRM) FI
via biliar intra y extrahepatica: (NEG|AFIRM) FI
(venas porta)? y suprahepaticas (NEG|AFIRM) FI
```

donde

```
NEG = no|sin|a
AFIRM = con
```

Relaciones entre FI y AE, con el FI apareciendo antes en la oración (FI-AE)

En la Tabla 4.10 se muestra una lista de conectores análoga a la anterior, pero con la entidad anatómica apareciendo en la oración antes que el *finding*.

conector	forman relación		no forman rel.		% que forma relación	
en	217	(23)	5	(0)	97.75 %	(100.00 %)
de la	48	(16)	4	(1)	92.31 %	(94.12 %)
␣	7	(6)	28	(21)	20.00 %	(22.22 %)
de	29	(23)	3	(0)	90.62 %	(100.00 %)
del	15	(8)	1	(1)	93.75 %	(88.89 %)
,	3	(3)	9	(9)	25.00 %	(25.00 %)
en el	2	(2)	1	(0)	66.67 %	(100.00 %)
a nivel de	2	(2)	0	(0)	100.00 %	(100.00 %)
1 vez	25	(25)	5	(5)	83.33 %	(83.33 %)
Total	349	(109)	59	(37)	85.53 %	(74.66 %)

Tab. 4.10: Para cada conector que separa un *finding* y una entidad anatómica, se muestran cuántos casos hay en los que el *finding* y la entidad anatómica tienen una relación *occurs_in* en el dataset de desarrollo. Se muestran entre paréntesis los casos únicos (en los que coinciden tanto el *finding* como la entidad anatómica) Se muestran sólo los que aparecen al menos 2 veces.

A partir de la tabla y generalizando los artículos a sus casos femeninos y plurales se construye la siguiente regla:

```
FI en (el|las?|los|λ)) AE
FI del|(de( las?| los|λ)) AE
FI ␣ AE
FI a nivel de AE
```

Relaciones entre FI y LO, con el FI apareciendo antes en la oración (FI-LO)

En la Tabla 4.11 se muestra una lista de conectores entre *finding* y *location*, con el *finding* apareciendo en la oración antes que la *location*.

conector	forman relación		no forman rel.		% que forma relación	
en	18	(16)	1	(1)	94.74 %	(94.12 %)
␣	16	(16)	1	(1)	94.12 %	(94.12 %)
,	2	(2)	5	(5)	28.57 %	(28.57 %)
λ	4	(4)	3	(2)	57.14 %	(66.67 %)
del	4	(4)	0	(0)	100.00 %	(100.00 %)
de la	2	(1)	0	(0)	100.00 %	(100.00 %)
a	2	(2)	0	(0)	100.00 %	(100.00 %)
(conectores que aparecen una vez)	34	(34)	2	(2)	94.44 %	(94.44 %)
Total	82	(79)	12	(11)	87.23 %	(87.78 %)

Tab. 4.11: Para cada conector que separa un *finding* y una *location* en ese orden, se muestran cuántos casos hay en los que el *finding* y la *location* tienen una relación *occurs_in* en el dataset de desarrollo. Se muestran entre paréntesis los casos únicos (en los que coinciden tanto el *finding* como la *location*). Se muestran sólo los que aparecen al menos dos veces.

A partir de la tabla se construye la siguiente regla:

```
FI (de (las?|los|λ))|del|,?( ) LO
FI (en (las?|el|los|λ))|,?( ) LO
FI (a|,?( )) LO
```

Relaciones entre FI y LO, con la LO apareciendo antes en la oración (LO-FI)

En la Tabla 4.12 se muestra una lista de conectores entre *finding* y *location*, con la *location* apareciendo en la oración antes que el finding.

conector	forman relación		no forman rel.		% que forma relación	
␣	18	(17)	2	(2)	90.00 %	(89.47 %)
se observa	11	(11)	1	(1)	91.67 %	(91.67 %)
se visualiza	7	(7)	0	(0)	100.00 %	(100.00 %)
λ	6	(6)	1	(1)	85.71 %	(85.71 %)
,	2	(2)	3	(3)	40.00 %	(40.00 %)
con	2	(2)	3	(3)	40.00 %	(40.00 %)
, se observa	3	(3)	0	(0)	100.00 %	(100.00 %)
se observan	2	(2)	0	(0)	100.00 %	(100.00 %)
se identifico	2	(2)	0	(0)	100.00 %	(100.00 %)
ecogenicos, con	2	(1)	0	(0)	100.00 %	(100.00 %)
de rinon izquierdo se observa	2	(1)	0	(0)	100.00 %	(100.00 %)
con leve	2	(2)	0	(0)	100.00 %	(100.00 %)
(Relaciones que aparecen una vez)	31	(31)	1	(1)	96.88 %	(96.88 %)
Total	89	(85)	12	(12)	88.11 %	(87.62 %)

Tab. 4.12: Para cada conector que separa una *location* y un *finding* en ese orden, se muestran cuántos casos hay en los que el *finding* y la *location* tienen una relación *occurs_in* en el dataset de desarrollo. Se muestran entre paréntesis los casos únicos (en los que coinciden tanto el *finding* como la *location*. Se muestran sólo los que aparecen al menos dos veces.

A partir de la tabla se construye la siguiente regla:

```
LO,? se (observ|visualiz|detect|identific)(an?|aron|o) (ADJ)? FI
LO,?( ) FI
LO AFIRM|NEG (ADJ)? FI
LO λ FI
ADJ → mucho|leve|poco
AFIRM → con
NEG → no|sin
```

4.6. Métodos basados en Redes Neuronales Convolucionales

Para este método se partió de la idea utilizada en dos trabajos previos [30, 3], mencionada en la Sección 3.4.

La arquitectura de la red planteada por los autores consiste en una capa convolucional, seguida de una capa de *pooling*, luego una capa densa; y finalmente la salida. La arquitectura utilizada para este trabajo se muestra en la Figura 4.9.

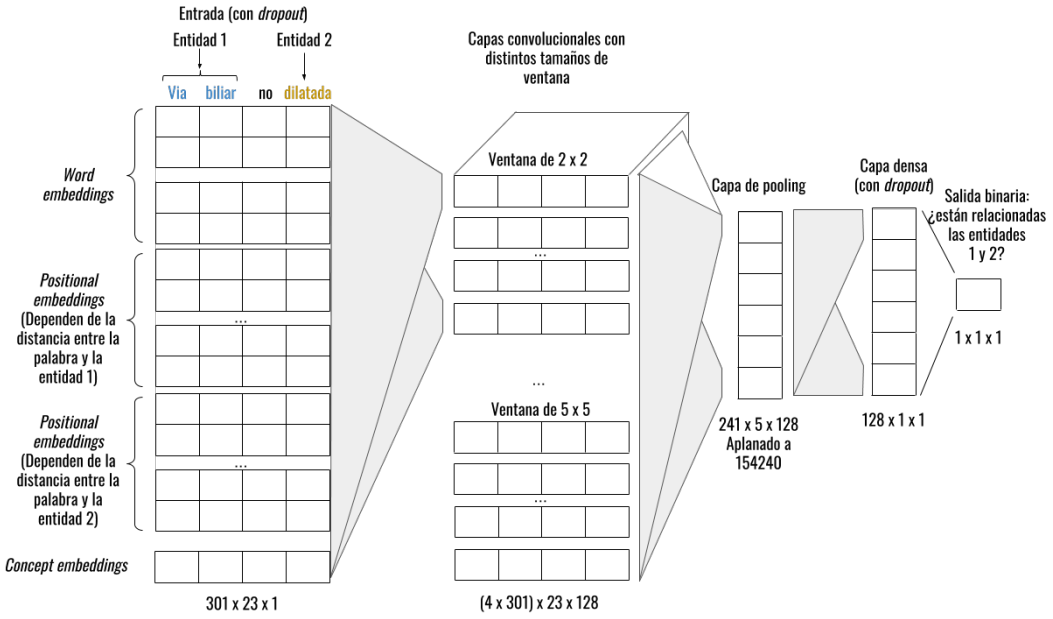

Fig. 4.9: Arquitectura de la red utilizada. Los valores debajo de cada capa corresponden a su dimensión, asumiendo que se utilizan *concept embeddings* y 2, 3, 4 y 5 como tamaños de ventana. Para cada capa se muestra en gris la entrada de algunos vectores de ejemplo.

En primer lugar se tiene una capa convolucional, con dimensión $x \times l \times 1$, donde x es el largo del vector de $features$ de cada palabra y l el largo máximo de una oración, como se explica más abajo. Sobre la capa convolucional se pueden aplicar distintos tamaños de ventana. Para esto se concatenan las salidas obtenidas de cada uno de los tamaños utilizados. La salida de la capa convolucional agrega además una dimensión para la cantidad de filtros utilizados.

La salida de la capa convolucional pasa luego por una capa de *pooling*. El resultado de esto se aplana y se toma como entrada para una capa densa. Finalmente, la salida es binaria, indicando si las entidades de la entrada están relacionadas.

La entrada de la red es una matriz, en la que cada fila es una oración y la i-ésima columna corresponde a la i-ésima palabra de la oración. A cada palabra se le asigna un vector dado por la concatenación de diferentes vectores asociados a la palabra, que se explicarán en las próximas subsecciones.

En el método en el que nos basamos cada entrada tiene exactamente dos entidades a relacionar. En este trabajo cada oración puede tener n entidades. La adaptación entonces consistió en generar una instancia de aprendizaje para cada posible par de entidades, repitiendo cada oración tantas veces como fuera necesario para cubrirlos todos. Es decir, la misma oración puede generar varias instancias de aprendizaje. Los *word embeddings* van a ser idénticos, pero el resto de la entrada no, ya que la información posicional de cada palabra depende de cuál sea la entidad contra la que se mida.

Como cada oración puede tener una cantidad diferente de palabras, las matrices podrían tener una cantidad diferente de columnas. Para evitar esto, los autores definen un largo máximo para las oraciones y agregan *tokens* de *padding* en todas las oraciones para que lleguen a dicho largo.

Por problemas de segmentación de oraciones había en el conjunto de datos utilizado casos de oraciones muy largas. Se tomó como largo máximo dos desviaciones estándar de la media (23 palabras), descartando las que excedieran ese largo. La distribución de las oraciones se puede ver en la Figura 4.10.

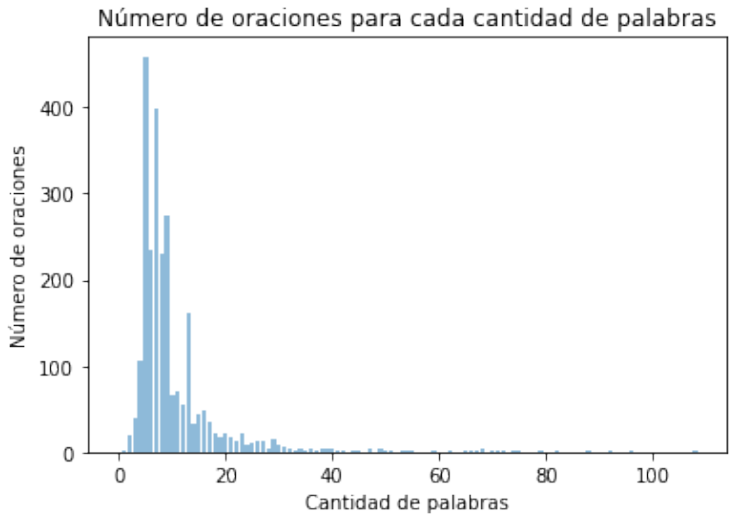

Fig. 4.10: Distribución del largo de las oraciones.

La red se entrena y evalúa por separado para relaciones FI-AE y FI-LO. Luego se suman los TP, FP y FN obtenidos por ambas relaciones.

4.6.1. *Word embeddings*

Para el input se experimentó con los siguientes conjunto de *word embeddings*:

- **Uchile**:[6] Entrenado a partir del *billion word corpus* [40], un corpus de 1400 millones de palabras en español, que no son del dominio específico de la biomedicina.
- **Uchile reentrenado**: Los vectores del conjunto anterior, reentrenados con todo el corpus utilizado para este trabajo (80000 informes, 4.5 millones de palabras).
- **Scielo + wikipedia**:[7] Entrenado a partir de dos corpus del dominio biomédico y presentado en [41]. El primero es todo el texto en español existente en diciembre de 2018 en la *Scientific Electronic Library Online* [42], un proyecto de biblioteca

[6] Disponible en https://github.com/dccuchile/spanish-word-embeddings a agosto de 2020
[7] Disponible en https://github.com/PlanTL-SANIDAD/Embeddings a agosto de 2020

electrónica de revistas científicas. El segundo son todos los artículos de Wikipedia en español de las categorías Farmacología, Farmacia, Medicina y Biología. En total son 182 millones de palabras. Si bien ambos son del area de la biomedicina, son textos distintos a los informes médicos utilizados en este trabajo.

- **Scielo + wikipedia reentrenados**: Los vectores del conjunto anterior, reentrenados con todo el corpus utilizado para este trabajo.
- **Custom**: Vectores entrenados desde cero con todo el corpus de este trabajo.
- **Random**: Vectores inicializados aleatoriamente, como control para verificar la influencia real de la calidad de los *embeddings* utilizados en el resultado final.

Todos (salvo Random) fueron entrenados o reentrenados utilizando el algoritmo *skip-gram* de la biblioteca *fastText* [38] con los siguientes parámetros:

- Cantidad mínima de apariciones: 5
- Largo de *n*grama de palabras: 1
- Largo mínimo de *n*grama de caracteres: 3
- Largo máximo de *n*grama de caracteres: 6
- Tamaño de los vectores generados: 100
- *Epochs*: 20
- Tamaño de la ventana de contexto: 5

4.6.2. *Positional embeddings*

A la entrada utilizada es necesario agregarle información del orden de las palabras y de cuál es el par de entidades que se busca clasificar. Para esto se tomó la idea de Vaswani et al. [43] de codificar la distancia como una onda siguiendo la siguiente fórmula:

$$PE(x,\ i) = sin(x/10000^{i/d_{model}}) \text{ si } i \text{ es par.}$$
$$PE(x,\ i) = cos(x/10000^{(i-1)/d_{model}}) \text{ si } i \text{ es impar.}$$

Donde x es el valor a codificar, d_{model} el largo del vector a generar (se tomó como igual al largo de los *word wmbeddings*) e i el valor en la i-ésima posición de dicho vector.

Esto refuerza la idea de orden, porque $PE(x,\ i)$ es una función lineal de $PE(x,\ i+k)$ para todo x y para todo k.

4.6.3. *Offset embeddings*

Una alternativa a los *positional embeddings* es agregarle a cada palabra dos valores que indiquen la distancia a las entidades de interés. La distancia se mide en palabras y el signo nos indica la posición relativa. Un ejemplo se puede ver en la Figura 4.11. El primer vector representa la distancia de cada palabra a la entidad `pelvis renales`. Así,

`ambas` tiene un valor de -1, ya que se encuentra justo antes de la entidad y `pelvis` vale 0 porque forma parte de ella. El segundo vector es análogo para la otra entidad.

Ambas	pelvis	renales	de	características	dobles	o	bífidas
-1	0	0	1	2	3	4	5
-5	-4	-3	-2	-1	0	0	0

Fig. 4.11: Ejemplo de *offset embeddings*. El primer vector representa la distancia a la entidad `pelvis renales`, mientras que el segundo es la distancia a `dobles o bífidas`.

4.6.4. *Concept embeddings*

Un dato importante sobre una palabra es a qué entidad de todas las anotadas pertenece. Para esto se asignó un número natural arbitrario a cada tipo de entidad. A cada palabra se le agrega entonces el número correspondiente a su tipo de entidad, o 0 si no está etiquetada. Esto se puede ver en la Figura 4.12.

Esta información puede estar contenida de alguna forma en los *word embeddings*; tener un vector específico sirve para reforzar su importancia.

Ambas	pelvis	renales	de	características	dobles	o	bífidas
0	1	1	0	0	2	2	2

Fig. 4.12: Ejemplo de *concept embeddings*. Un valor de 1 representa que el *token* indicado con dicho número corresponde a una entidad del tipo AE, mientras que 2 representa que corresponde a una entidad FI.

5. RESULTADOS

En esta sección se describen los resultados obtenidos por cada método por separado; luego se realiza una comparación entre ellos.

Para todos los métodos tomamos en cuenta precisión (P), *recall* (R) y $F1$-*score* tanto para casos sin repetidos como con repetidos (ver Subsección 4.1.5 para la definición utilizada de con repetidos y sin repetidos).

5.1. Método basado en co-ocurrencia

En esta sección se muestran los resultados obtenidos por el método basado en co-ocurrencia. En la Tabla 5.1 se muestra un análisis centrado únicamente en las relaciones AE-FI en el que se tiene en cuenta que las relaciones pueden ser entre entidades de distintas oraciones.

En la Tabla 5.2 y la Tabla 5.3 se muestran los resultados obtenidos por este método para ambos tipos de relación solo para entidades dentro de la misma oración,

	TP	FN	FP	P	R	F1
Existencia de relación	319	0	46	0.87	1	0.93
Relaciones intra oración detectadas	364	65	106	0.77	0.84	0.80
Relaciones totales detectadas	364	88	106	0.77	0.80	0.79

Tab. 5.1: Resultados del *baseline* sobre las relaciones entre entidades anatómicas y $findings$ en el conjunto de test. Llamamos "existencia de relación" a detectar correctamente que en una oración existe una relación, independientemente de que hayamos detectado la cantidad o las relaciones correctas. Dado que el algoritmo no detecta relaciones entre entidades que se encuentran en diferentes oraciones, consideramos por separado los resultados obtenidos sobre las relaciones entre entidades en la misma oración y sobre todas las relaciones.

Repetidos	Entidades relacionadas	TP	FN	FP	P	R	F1
Con repetidos	Relaciones FI-AE	1329	277	375	0.780	0.828	0.803
	Relaciones FI-LO	123	48	110	0.528	0.719	0.609
	Total occurs_in	1452	325	485	0.750	0.817	0.782
Sin repetidos	Relaciones FI-AE	556	218	328	0.629	0.718	0.671
	Relaciones FI-LO	122	48	102	0.545	0.718	0.619
	Total occurs_in	679	266	438	0.608	0.719	0.659

Tab. 5.2: Resultados del *baseline* en el conjunto de desarrollo.

Repetidos	Entidades relacionadas	TP	FN	FP	P	R	F1
Con repetidos	Relaciones FI-AE	364	65	106	0.77	0.85	0.81
	Relaciones FI-LO	31	17	39	0.44	0.65	0.53
	Total occurs_in	395	82	145	0.73	0.83	0.78
Sin repetidos	Relaciones FI-AE	156	59	97	0.62	0.73	0.67
	Relaciones FI-LO	31	17	39	0.44	0.65	0.53
	Total occurs_in	187	76	136	0.58	0.71	0.64

Tab. 5.3: Resultados del *baseline* en el conjunto de test.

5.1.1. Análisis de resultados

En primer lugar podemos mencionar que se obtuvo un mejor resultado contando los repetidos. Esto tiene una explicación simple que es que las oraciones que más se repiten se corresponden con los casos más frecuentes en la Tabla 4.7.

Por otro lado, los resultados fueron mejores sobre el conjunto de desarrollo que sobre el conjunto de test, tanto con repetidos como sin repetidos. Esto también es previsible, ya que las reglas para este algoritmo se construyeron sobre un subconjunto del conjunto de desarrollo. Sin embargo, los resultados son muy similares, indicando que las reglas fueron útiles al generalizar.

De los dos pares de entidades que se reconocen, los mejores resultados se obtuvieron para AE-FI. En este punto hay que mencionar que se disponía de muchas menos relaciones FI-LO para construir las reglas, y el algoritmo no era el más adecuado para la distribución de las relaciones, como se explica en la Subsección 4.4.2.

Por último, hay que notar que se obtuvo un *recall* alto, pero una precisión no tan alta.

5.2. Método basado en Reglas

En la Tabla 5.4 y la Tabla 5.5 se muestra el resultado obtenido por el método basado en reglas en el conjunto de desarrollo y en el conjunto de test, respectivamente.

Repetidos	Entidades relacionadas	TP	FN	FP	P	R	F1
Con repetidos	Relaciones FI-AE	339	66	27	0.92	0.83	0.88
	Relaciones FI-LO	31	19	8	0.79	0.62	0.70
	Total occurs_in	370	85	35	0.91	0.81	0.86
Sin repetidos	Relaciones FI-AE	166	63	19	0.89	0.72	0.80
	Relaciones FI-LO	31	19	7	0.82	0.62	0.71
	Total occurs_in	197	82	26	0.89	0.71	0.79

Tab. 5.4: Resultados del método basado en reglas en el conjunto de desarrollo.

Repetidos	Entidades relacionadas	TP	FN	FP	P	R	F1
Con repetidos	Relaciones FI-AE	359	71	27	0.93	0.83	0.88
	Relaciones FI-LO	25	23	8	0.75	0.52	0.61
	Total occurs_in	384	94	35	0.91	0.80	0.85
Sin repetidos	Relaciones FI-AE	145	71	15	0.91	0.67	0.77
	Relaciones FI-LO	25	23	8	0.75	0.52	0.61
	Total occurs_in	170	94	23	0.88	0.64	0.74

Tab. 5.5: Resultados del método basado en reglas en el conjunto de test.

5.2.1. Análisis de resultados

Una vez más se halló que se obtienen mejores resultados con repetidos. Esto se explica porque las oraciones que más veces se repiten están incluidas en las reglas. Por ejemplo, `no se observó líquido libre en cavidad` es aceptada por la regla `en`.

Tanto con repetidos como sin repetidos se obtuvieron mejores resultados sobre el conjunto de test que sobre el de desarrollo. Son resultados parecidos, lo que nos indica que las reglas fueron lo suficientemente generales para poder aplicarse a un conjunto de textos distinto al que se utilizó para generarlas.

Sin embargo, la precisión fue alta, pero no el *recall*. Es decir: las reglas son buenas para distinguir los casos positivos pero es necesario analizar una cantidad de textos mayor para crear reglas que permitan incluir casos menos comunes.

Los resultados obtenidos fueron mucho mejores para pares FI-AE que para pares FI-LO. Al igual que en el método anterior, encontramos que hay pocas relaciones FI-LO para sacar reglas y pocas para evaluar. Además, no hay oraciones repetidas con relaciones FI-LO en el conjunto de test.

5.3. Método basado en Redes Neuronales Convolucionales

En esta sección se muestran los resultados obtenidos por el método basado en CNN. Dado que hay casos en los que las diferencias dadas por un parámetro son difíciles de apreciar, se darán con tres decimales de precisión.

Para este método existen diversas combinaciones de valores de entrada que se pueden utilizar:

- **Word embeddings (WE)**: Todos los mencionados en la Subsección 4.6.1.
- **Información posicional**: Utilizar *positional embeddings*, *offset embeddings* o ninguno de los dos.
- ***Concept embeddings* (CE)**: Utilizarlos o no utilizarlos.

Además, se consideraron dos hiperparámetros de la red:

- Los tamaños de ventana a utilizar (**CWS** por las siglas en inglés). Basándonos en Nguyen y Grishman [30] consideramos como posibles valores 2, [2-3], [2-4] y [2-5]. La notación [*a*-*b*] implica todos los naturales entre a y b, ambos incluidos.
- ***Dropout***: Utilizar dos capas de dropout con probabilidad 0.2, de las que una se sitúa luego de la entrada y otra antes de la última capa, o no utilizar *dropout*.

5.3.1. Análisis de $folds$

Para evaluar este método se utilizó *5-fold-cross-validation* sobre el conjunto de desarrollo. La división de los datos en $folds$ se hizo por informe. Es decir, cada $fold$ tiene 82 informes del total de 410. Esta división es fija y se mantuvo a lo largo de todo el proceso de experimentación. Sin embargo, cada informe puede tener un número variable de oraciones, y cada oración un número distinto de relaciones y entidades.

Para obtener la precisión, *recall* y F1, existen varias posibilidades. Para evaluarlas se hizo una ejecución de prueba del algoritmo con una combinación de parámetros fija y con repetidos. Los resultados otenidos en cada $fold$ se pueden ver en la Subsección 5.3.1. Los parámetros utilizados fueron:

- Word embeddings: Uchile
- Tamaños de ventana convolucional: 2-5
- Información posicional: $Offset\ Embeddings$
- Dropout: primera y última capa
- $Concept\ Embeddings$: no

Fold	# relaciones		# oraciones		TP	FN	FP	P	R	F1
1	458	21.45 %	646	18.94 %	416	42	79	0.840	0.908	0.873
2	380	17.80 %	555	16.27 %	344	36	120	0.741	0.905	0.815
3	338	15.83 %	617	18.09 %	280	58	67	0.807	0.828	0.818
4	400	18.74 %	767	22.49 %	371	29	65	0.851	0.928	0.888
5	559	26.18 %	826	24.22 %	521	38	64	0.891	0.932	0.911
Total	2135	100.00 %	3411	100.00 %	1932	203	395	0.830	0.905	**0.866**

Tab. 5.6: Análisis de los resultados por $fold$.

Las posibilidades a analizar son:

1. Calcular un promedio del $F1$ de cada $fold$ ("promedio macro"). Se obtiene **0.861**.
2. Ponderar el F1 de cada $fold$ por la proporción de **relaciones** que tiene. En este caso se obtiene un valor de **0.867**.
3. Ponderar el F1 de cada $fold$ por la proporción de **oraciones** que tiene. El valor obtenido es **0.866**.

4. Sumar los verdaderos positivos, falsos positivos y falsos negativos de cada $fold$ para todas las relaciones de interés. También se obtiene **0.866**.

Si bien se observa heterogeneidad entre los $folds$, tanto en cantidad de palabras como de oraciones, los resultados de las últimas tres opciones son muy parecidos, es decir que cualquiera de las tres serviría para obtener un resultado representativo.

La decisión fue utilizar la última opción, calcular los resultados sumando lo obtenido en cada $fold$, ya que es un punto intermedio entre pesar por oraciones y por relaciones.

5.3.2. Word Embeddings

Para evaluar los distintos word embeddings, en primer lugar, se realizó un test de analogías, como se describe en la Subsección 2.3.1. Para esto se utilizó el conjunto desarrollado para Word2Vec [23], traducido por Cardellino [40].[1]

El test utilizado tiene algunos problemas. En primer lugar tiene defectos de traducción. Por ejemplo, la línea

`going went reading read`

es traducida como

`yendo fue leyendo leer`

porque `read` es tanto el presente como el pasado del verbo `leer`. En segundo lugar, este test es inadecuado porque no tiene casi palabras del dominio exclusivo de la biomedicina, pero no se encontró uno en español que lo sea. Por lo que, por completitud, se presentarán los resultados obtenidos pero se dará prioridad a otros análisis.

El test está dividido en doce secciones, que se muestran en la Tabla 5.7. En negrita se marcan las que consideramos más importantes para este análisis.

Sección	Ejemplo			
capital-common-countries	Atenas	Grecia	Bagdad	Irak
capital-world	Abuja	Nigeria	Accra	Ghana
currency	Argelia	dinar	Angola	kwanza
city-in-state	Chicago	Illinois	Houston	Texas
family	chico	chica	hermano	hermana
gram1-adjective-to-adverb	usual	usualmente	alegre	alegremente
gram2-opposite	aceptable	inaceptable	consciente	inconsciente
gram5-present-participle	describir	describiendo	ir	yendo
gram6-nationality-adjective	Argentina	Argentino	Australia	Australiano
gram7-past-tense	yendo	fue	viendo	vio
gram8-plural	botella	botellas	edificio	edificios
gram9-plural-verbs	disminuir	disminuye	describir	describe

Tab. 5.7: Secciones del test de analogías.

[1] Disponible en http://cs.famaf.unc.edu.ar/∼ccardellino/SBWCE/clean_corpus.tar.bz2 a agosto de 2020

Los resultados obtenidos se muestran en la Tabla 5.8. En este punto es necesario remarcar que los resultados obtenidos por Scielo + wikipedia en el trabajo en el que se lo presenta [41] son mejores a los obtenidos en este experimento y mejores que los de Uchile, pero el test que utilizaron no está disponible públicamente y el problema que buscan resolver es distinto al de este trabajo.

Word Embeddings	Accuracy (12 secc.)	Accuracy (6 secc.)
Uchile	**0.527**	**0.550**
Random	0	0
Custom	0.499	0.499
Scielo + wikipedia reentrenado	0.100	0.100
Scielo + wikipedia	0.377	0.526
Uchile reentrenado	0.313	0.313

Tab. 5.8: Análisis intrínseco de *word embeddings*. En la columna derecha se muestran resultados obtenidos utilizando únicamente las secciones indicadas en la Tabla 5.7. En **negrita** se indica el mejor resultado para cada columna.

A continuación se realizó un análisis extrínseco. Es decir, se fijó el resto de los parámetros y se analizó el resultado de utilizar los distintos *word embeddings* en la red. Los parámetros utilizados para esto fueron:

- Tamaño de ventana: [2-5]
- Información posicional: Offset embeddings
- Concept embeddings: Sí
- Dropout: Primera y última capa

Se hizo *5-fold cross validation* sobre el conjunto de datos de desarrollo, como se explica en la Subsección 5.3.1.

Los resultados se pueden ver en la Tabla 5.9. Los mejores resultados fueron obtenidos por Custom y Uchile. Tomando en cuenta repetidos, Custom fue el mejor, seguido por Uchile, mientras que sin repetidos el resultado fue el inverso. Como consecuencia de esto, se tomaron estos dos *word embeddings* para el resto del análisis.

	Con repetidos			Sin Repetidos		
	P	R	F1	P	R	F1
Random	0.858	0.824	0.841	0.784	0.728	0.755
Scielo + wikipedia reentrenado	0.830	0.864	0.846	0.719	0.846	0.778
Uchile	0.830	0.905	0.866	0.734	0.888	**0.804**
Custom	0.821	0.922	**0.868**	0.716	0.910	0.801
Scielo + wikipedia	0.821	0.911	0.864	0.725	0.885	0.797
Uchile reentrenado	0.808	0.921	0.861	0.713	0.870	0.784

Tab. 5.9: Análisis extrínseco de *word embeddings*

5.3.3. Tamaño de ventana

A continuación se realizó un procedimiento similar al análisis extrínseco de *word embeddings* para el resto de los parámetros. Para cada uno de ellos, se analizó el impacto que tenía en el resultado para algunas combinaciones del resto de los parámetros. Este procedimiento se conoce como *grid search*.

Para analizar los distintos tamaños de ventana se fijaron los siguientes parámetros:

- Información posicional: Offset embeddings
- Dropout: Primera y última capa
- Concept embeddings: Sí

Los resultados se pueden ver en la Tabla 5.10. Los mejores resultados, tanto con repetidos como sin repetidos, se obtuvieron utilizando 2 como tamaño de ventana para Uchile y [2-5] para Custom.

WE	CWS	Con repetidos			Sin Repetidos		
		P	R	F1	P	R	F1
Uchile	[2-5]	0.830	0.905	**0.866**	0.734	0.888	**0.804**
	[2-4]	0.838	0.886	0.861	0.742	0.859	0.796
	[2-3]	0.822	0.911	0.864	0.742	0.858	0.795
	2	0.857	0.870	0.864	0.731	0.869	0.794
Custom	[2-5]	0.821	0.922	0.868	0.716	0.910	0.801
	[2-4]	0.829	0.912	0.869	0.753	0.830	0.790
	[2-3]	0.807	0.933	0.866	0.733	0.889	0.803
	2	0.830	0.915	**0.871**	0.733	0.908	**0.811**

Tab. 5.10: Análisis de los distintos tamaños de ventana. WE es *Word Embeddings* y CWS tamaño de ventana.

5.3.4. Información posicional

En la Tabla 5.11 se puede ver el resultado de utilizar distintos tipos de información posicional para las dos mejores combinaciones de la sección anterior. En todos los casos se obtuvo un resultado similar o peor agregando *positional embeddings*, mientras que se obtuvo un mejor resultado utilizando $offset\ embeddings$.

Parámetros fijos:

- Dropout: 1ra y última capa
- Concept embeddings: Sí

WE	CWS	Información posicional	Con repetidos			Sin Repetidos		
			P	R	F1	P	R	F1
Custom	2	No	0.81	0.87	0.837	0.73	0.75	0.744
		Positional embeddings	0.81	0.86	0.836	0.73	0.70	0.716
		Offset embeddings	0.83	0.92	**0.871**	0.73	0.91	**0.811**
Uchile	[2-5]	No	0.80	0.90	0.847	0.71	0.84	0.772
		Positional embeddings	0.83	0.84	0.838	0.71	0.75	0.732
		Offset embeddings	0.83	0.90	**0.866**	0.73	0.89	**0.804**

Tab. 5.11: Análisis de los distintos tipos de información posicional.

5.3.5. *Concept embeddings*

Los parámetros fijos para este experimento fueron:

- Dropout: 1ra y última capa
- Información posicional: Offset embeddings

Como se puede ver en la Tabla 5.12, los *concept embeddings* mejoraron el resultado en todos los casos.

WE	CWS	CE	Con repetidos			Sin Repetidos		
			P	R	F1	P	R	F1
Custom	2	Sí	0.830	0.915	**0.871**	0.733	0.908	**0.811**
		No	0.841	0.864	0.852	0.697	0.864	0.772
Uchile	[2-5]	Sí	0.830	0.905	**0.866**	0.734	0.888	**0.804**
		No	0.813	0.917	0.862	0.748	0.868	0.803

Tab. 5.12: Análisis de resultados con y sin *concept embeddings* (CE).

5.3.6. *Dropout*

Los parámetros fijos para este experimento fueron:

- Concept embeddings: Sí
- Información posicional: Offset embeddings

En la Tabla 5.13 se puede ver que el uso de *Dropout* mejoró el resultado obtenido cuando se consideraron oraciones sin repetidos, pero lo empeoró cuando se consideraron todas las oraciones.

WE	CWS	Dropout	Con repetidos			Sin Repetidos		
			P	R	F1	P	R	F1
Custom	2	No	0.845	0.922	**0.882**	0.739	0.860	0.795
		1ra y última capa	0.830	0.915	0.871	0.733	0.908	**0.811**
Uchile	[2-5]	No	0.822	0.920	**0.868**	0.750	0.794	0.771
		1ra y última capa	0.830	0.905	0.866	0.734	0.888	**0.804**

Tab. 5.13: Análisis de resultados con y sin *dropout*.

Para tomar una decisión en este punto se decidió priorizar el resultado con repetidos, es decir, no usar dropout. Esto se puede justificar recordando que en caso de utilizar este algoritmo en un contexto real, los datos van a tener repetidos, no van a estar filtrados.

Por otro lado, dado que ya se había tomado una decisión sobre información posicional y sobre *Concept Embeddings*, queda elegir si se utilizan los *Word Embeddings* Custom con tamaño de ventana 2 o Uchile con ventana entre 2 y 5. Lo que se observa en la Tabla 5.13 es tanto con repetidos como sin repetidos la primera opción es la que devuelve un mejor F1.

5.3.7. Resultados sobre el conjunto de test

La combinación óptima de parámetros que se obtiene de las subsecciones anteriores entonces es:

- Word embeddings: Custom
- Tamaño de ventana: 2
- Información posicional: Offset embeddings
- Concept embeddings: Sí
- Dropout: No

Con esta combinación se realizó entonces un entrenamiento sobre todo el conjunto de desarrollo, para luego evaluar la red con el conjunto de test. Los resultados se pueden ver en la Tabla 5.14.

En la Tabla 5.15 se hace una comparación entre estos últimos resultados y los obtenidos para el conjunto de desarrollo con la combinación de parámetros elegida. Estos se corresponden con las filas sin *dropout* de la tabla Tabla 5.13.

Repetidos	Entidades relacionadas	TP	FN	FP	P	R	F1
Con repetidos	Relaciones FI-AE	350	37	46	0.884	0.904	0.894
	Relaciones FI-LO	26	2	29	0.473	0.929	0.627
	Total occurs_in	376	39	75	0.834	0.906	0.867
Sin repetidos	Relaciones FI-AE	152	21	61	0.714	0.879	0.788
	Relaciones FI-LO	24	4	27	0.471	0.857	0.608
	Total occurs_in	176	25	88	0.667	0.876	0.757

Tab. 5.14: Resultados del método basado en CNN en el conjunto de test. Hay una menor cantidad de relaciones que para los otros métodos por el límite de palabras.

Repetidos	Conjunto de datos	P	R	F1
Con repetidos	Desarrollo	0.845	0.922	0.882
	Test	0.834	0.906	0.867
Sin repetidos	Desarrollo	0.739	0.860	0.795
	Test	0.667	0.876	0.757

Tab. 5.15: Comparación de los resultados en el conjunto de test y en el de desarrollo.

5.3.8. Análisis de resultados

Los resultados obtenidos sobre el conjunto de test son peores que los del conjunto de desarrollo, pero no hay una gran diferencia, al menos con repetidos.

Se obtiene un *recall* alto y una precisión no tan alta.

Los resultados para relaciones FI-AE son mucho mejores que para relaciones FI-LO. Al igual que en los otros métodos, esto se puede explicar en parte por la diferencia en la cantidad de relaciones, y en la cantidad de repeticiones de las relaciones FI-AE.

Algo que se debe mencionar en este punto es que la diferencia en el resultado dada por los distintos *word embeddings* es muy menor, como se puede ver en la Tabla 5.8.

De hecho, se obtiene un resultado muy alto utilizando *word embeddings* aleatorias. Si bien es el de F1 más bajo, la diferencia con el siguiente es de muy pocos puntos, y es el que obtuvo mejor precisión.

En el caso con repetidos esto se puede explicar por la cantidad de oraciones que aparecen varias veces, pero la situación es similar en el caso sin repetidos.

Observando la tabla Tabla B.1 encontramos que incluso filtrando las oraciones repetidas hay palabras que aparecen más de cien veces, y en más del 90 % de los casos son parte de una relación, como `alteraciones`, `retroperitoneo` y `vascular`. Esto podría ser una explicación a este resultado: la red estaría aprendiendo que hay relación siempre que tenga como entrada los vectores correspondientes a esas palabras, y eso llevaría a un resultado correcto en más del 90 % de los casos que haya alguna de estas palabras.

5.4. Comparación entre los métodos

Como se explica en la Sección 4.6, al implementar el método basado en CNN no se tuvieron en cuenta las oraciones de más de 23 palabras de largo. Para poder comparar los resultados de este método con los otros dos resulta entonces necesario usar el mismo conjunto de datos para los tres. Para eso se volvieron a ejecutar los dos primeros métodos, pero solamente sobre las oraciones de ese largo del conjunto de test.

Los resultados obtenidos se muestran en la Tabla 5.16 y en la Figura 5.1.

Método	Con repetidos			Sin Repetidos		
	P	R	F1	P	R	F1
Co-ocurrencia	0.731	0.828	0.777	0.594	0.830	0.693
Reglas	0.935	0.831	**0.880**	0.920	0.683	**0.784**
CNN	0.834	0.906	0.867	0.667	0.876	0.757

Tab. 5.16: Comparación de los resultados obtenidos por cada método en oraciones de hasta 23 palabras.

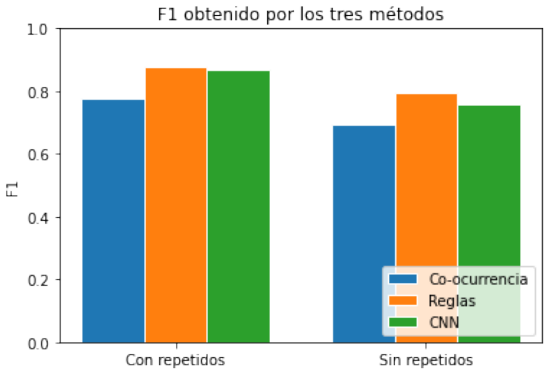

Fig. 5.1: Comparación de los resultados obtenidos por cada método en oraciones de hasta 23 palabras.

5.4.1. Análisis de resultados

Lo primero que podemos ver es que el mejor resultado se obtiene con el método basado en reglas. Esto ocurre tanto con repetidos como sin repetidos. Sin embargo, el método basado en CNN obtiene un mayor *recall*.

Ambos métodos obtienen resultados mejores que el *baseline* propuesto.

No resulta apropiado comparar los resultados obtenidos con los de los trabajos destacados en la Tabla 3.1. En primer lugar los trabajos mencionados trabajan con extracción y clasificación de relaciones, mientras que este solo se ocupa de lo primero. En segundo lugar, son tareas distintas, se buscan relaciones distintas y en textos en distintos idiomas. Por último, en el conjunto de datos utilizado en particular, hay una gran cantidad de oraciones y palabras repetidas, como se explica en la Subsección 4.1.5 y en la Subsección 5.3.8.

Un punto a destacar de todas formas es que, en la tabla mencionada, el principal método basado en reglas, desarrollado por Zweigenbaum y Ben Abacha [11], obtiene resultados muy similares a trabajos posteriores basados en CNN, con lo que cabía esperar un resultado como este.

5.4.2. Resultados según largo de la oración y distancia entre entidades

En la Figura 5.2 se muestra el resultado obtenido por cada método diferenciando según el largo de las oraciones sobre el conjunto de test, con repetidos. Los tres métodos obtienen mejores resultados cuanto más cortas son las oraciones. También se puede ver que la diferencia sobre el *baseline* de los otros dos métodos empieza a aparecer con las oraciones más largas. El método basado en CNN tiene el menor F1 con oraciones cortas, pero es el método menos afectado por este factor.

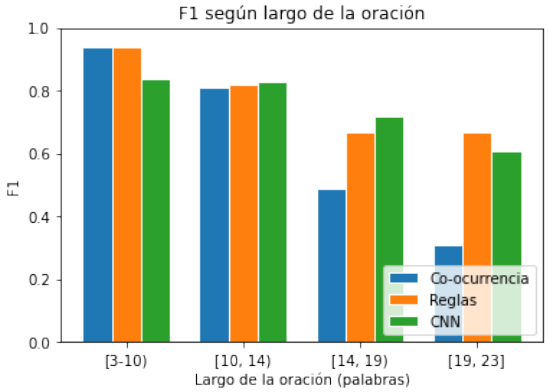

Fig. 5.2: Influencia del largo de las oraciones en el resultado obtenido, para cada uno de los métodos.

En la Figura 5.3 se muestra el resultado según la distancia entre las entidades de cada par. Al igual que en el caso anterior, los métodos obtienen mejores resultados cuanto menor sea la distancia entre las entidades. En el caso del método basado en reglas, no consideramos conectores largos, por lo que el F1 queda en 0. El método basado en CNN parece ser el menos afectado por este factor.

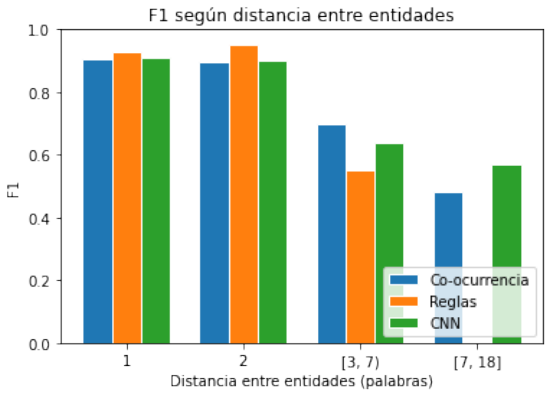

Fig. 5.3: Influencia de la distancia entre entidades en el resultado obtenido, para cada uno de los métodos.

6. CONCLUSIONES Y TRABAJO FUTURO

El objetivo de este trabajo era desarrollar distintos métodos de extracción de relaciones en informes médicos en español. En las secciones anteriores se describieron los tres métodos desarrollados y los experimentos realizados.

Para el método basado en CNN, además, se entrenaron *word embeddings* en español con los textos del corpus y se compararon los resultados con los obtenidos utilizando algunos vectores ya existentes, resultando mejores cuando solo se realizó el entrenamiento con los textos del *dataset*.

Encontramos que tanto el método basado en reglas como el método basado en CNN mejoraron el *baseline* propuesto. Los resultados obtenidos fueron mejores para el primero de estos métodos que para el segundo, aunque con una diferencia menor.

Al utilizar un método basado en reglas encontramos algunas desventajas esperables. En primer lugar requiere de un mayor esfuerzo manual para construir las reglas. Por otro lado, estas reglas son muy específicas del conjunto de datos utilizado; en cambio, la red ya entrenada debería poder utilizarse para detectar relaciones en otros textos similares. Estas desventajas pueden compensar de alguna forma la diferencia entre los resultados obtenidos con ambos métodos.

El método basado en reglas requiere tiempo para construirlas a partir de un análisis manual, pero su ejecución es muy rápida. Las reglas obtenidas son específicas para el corpus. El método basado en CNN requiere tiempo de entrenamiento, pero la red, una vez entrenada, puede utilizarse para otros textos.

Una desventaja de este último método es que tiene menos interpretabilidad. Por ejemplo, dado un falso negativo, no es claro por qué esa relación no fue detectada por la red ni qué habría que hacer para que lo hiciera.

También encontramos que la cantidad de repeticiones influye en el resultado, así como el tener más o menos entidades para construir reglas o para entrenar a la red. También influye el largo de las oraciones y la distancia entre las entidades.

6.1. Trabajo futuro

La principal dificultad a la hora de evaluar los métodos propuestos es el no utilizar un conjunto de datos estandarizado como los que existen en inglés. Es decir que sería prioritario desarrollarlo y, una vez disponible, utilizarlo para conocer el alcance real de los métodos aquí presentados. Idealmente se trataría de un *dataset* sobre el que se podría trabajar con **clasificación** de relaciones.

Otro obstáculo encontrado es la escasez de *word embeddings* en español, especialmente del dominio específico, y de tests estandarizados para evaluarlos. Dada la utilidad de estos para muchos problemas de procesamiento del lenguaje natural, creemos que hay mucho trabajo por hacer sobre ese tema.

Durante este trabajo se supuso dada en todo momento la anotación de entidades. Un

posible trabajo futuro sería desarrollar métodos de detección de entidades nombradas para este dominio; para luego comparar los resultados obtenidos al detectar relaciones sobre entidades etiquetadas con un algoritmo con los obtenidos utilizando entidades anotadas por expertos.

Los métodos desarrollados no hicieron uso de algunas técnicas del procesamiento del lenguaje natural con las que se podrían obtener mejores resultados, como el análisis sintáctico y el *PoS tagging*. Para cada palabra no se incluyó información sobre su rol en el lenguaje. Eso serviría tanto para incluir esa información en la entrada de la CNN, como se hace en Peng y Lu [14] como para generalizar algunas reglas. Por ejemplo, no es lo mismo tener como regla `AE y extrahepatica: no FI` que `AE y (ADJETIVO): no FI`.

Dado que con ambos métodos se obtuvieron buenos resultados, una posibilidad es crear un método híbrido que use tanto las reglas como las técnicas de aprendizaje profundo. Teniendo en cuenta los resultados obtenidos en la Subsección 5.4.2, podría consistir en usar reglas cuando se trate de oraciones cortas y entidades cercanas y la salida de la CNN en caso contrario.

Por último, sería interesante agregar un mecanismo de atención a la red, basándose en lo hecho en Asada et al. [31].

6.2. Difusión de resultados

Los resultados preliminares de este trabajo fueron presentados en un póster científico en la Latin American Meeting in Artificial Intelligence, Khipu 2019,[1] realizada en Montevideo, Uruguay.

[1] "Khipu - Latin American meeting in Artificial Intelligence". En: http://khipu.ai, visitado en agosto de 2020.

7. BIBLIOGRAFÍA

[1] Aurélie Névéol, Hercules Dalianis, Sumithra Velupillai, Guergana Savova y Pierre Zweigenbaum. "Clinical Natural Language Processing in languages other than English: opportunities and challenges". En: *Journal of Biomedical Semantics* 9.1 (2018), pág. 12.

[2] David M. Eberhard, Gary F. Simons y Charles D. Fennig (eds.) "What are the top 200 most spoken languages?" En: *Ethnologue: Languages of the World.* Twenty-second edition. SIL International. (2019).

[3] Roland Roller, Nils Rethmeier, Philippe Thomas, Marc Hübner, Hans Uszkoreit, Oliver Staeck, Klemens Budde, Fabian Halleck y Danilo Schmidt. "Detecting Named Entities and Relations in German Clinical Reports". En: *Lecture Notes in Computer Science*. Vol. 10713. 2018, págs. 115-124.

[4] D Jurafsky y JH Martin. *Speech and Language Processing (2nd ed.)* 2008.

[5] Dina Demner-Fushman, Kevin Bretonnel Cohen, Sophia Ananiadou y Junichi Tsujii, eds. *Proceedings of the 18th BioNLP Workshop and Shared Task*. Florence, Italy: Association for Computational Linguistics, ago. de 2019. URL: `https://www.aclweb.org/anthology/W19-5000`.

[6] Yuhao Zhang, Yuhui Zhang, Peng Qi, Christopher D. Manning y Curtis P. Langlotz. *Biomedical and Clinical English Model Packages in the Stanza Python NLP Library*. 2020. arXiv: `2007.14640 [cs.CL]`.

[7] Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Kinney, Yunyao Li, Ziyang Liu y William Merrill et al. *CORD-19: The COVID-19 Open Research Dataset*. 2020. arXiv: `2004.10706 [cs.DL]`.

[8] Christopher D Manning, Prabhakar Raghavan e Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[9] Markus Bundschus, Mathaeus Dejori, Martin Stetter, Volker Tresp y Hans-Peter Kriegel. "Extraction of semantic biomedical relations from text using conditional random fields". En: *BMC Bioinformatics* 9.1 (2008), pág. 207.

[10] Viviana Cotik. "Information extraction from Spanish radiology reports". En: *PhD Thesis*. 2018.

[11] Asma Ben Abacha y Pierre Zweigenbaum. "Automatic extraction of semantic relations between medical entities: A rule based approach". En: *Journal of biomedical semantics* 2 Suppl 5 (oct. de 2011), S4. DOI: `10.1186/2041-1480-2-S5-S4`.

[12] Abdul Wahab Muzaffar, Farooque Azam y Usman Qamar. "A Relation Extraction Framework for Biomedical Text Using Hybrid Feature Set". En: *Computational and Mathematical Methods in Medicine* (2015).

[13] Sunil Kumar Sahu y Ashish Anand. "Drug-Drug Interaction Extraction from Biomedical Texts". En: *Journal of Biomedical Informatics* 86 (2018), págs. 15-24.

[14] Yifan Peng y Zhiyong Lu. "Deep learning for extracting protein-protein interactions from biomedical literature". En: *BioNLP 2017*. Vancouver, Canada, Association for Computational Linguistics, ago. de 2017, págs. 29-38. DOI: 10.18653/v1/W17-2304.

[15] Linlin Wang, Zhu Cao, Gerard de Melo y Zhiyuan Liu. "Relation Classification via Multi-Level Attention CNNs". En: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, 2016, págs. 1298-1307.

[16] Simon Haykin. *Neural networks: A Comprehensive Foundation*. Prentice Hall PTR, 1994.

[17] David E Rumelhart, Geoffrey E Hinton y Ronald J Williams. "Learning representations by back-propagating errors". En: *Nature* 323.6088 (1986), págs. 533-536.

[18] Diederik Kingma y Jimmy Ba. "Adam: A Method for Stochastic Optimization". En: *International Conference on Learning Representations* (dic. de 2014).

[19] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever y Ruslan Salakhutdinov. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". En: *The Journal of Machine Learning Research* 15.1 (2014), págs. 1929-1958.

[20] Julián Palladino. "Adaptación de dominio no supervisada para segmentación de resonancias magnéticas cerebrales mediante redes adversarias de consistencia cíclica". En: *Tesis de Licenciatura. Director: Enzo Ferrante*. 2020.

[21] Ian Goodfellow, Yoshua Bengio y Aaron Courville. *Deep Learning*. MIT Press, 2016.

[22] Ludwig Wittgenstein. *Philosophical Investigations*. Oxford: Basil Blackwell, 1953. ISBN: 0631119000.

[23] Tomas Mikolov, G.s Corrado, Kai Chen y Jeffrey Dean. "Efficient Estimation of Word Representations in Vector Space". En: ene. de 2013, págs. 1-12.

[24] Alfred V. Aho, Ravi Sethi y Jeffrey D. Ullman. *Compilers: Principles, Techniques, and Tools*. USA: Addison-Wesley Longman Publishing Co., Inc., 1986. ISBN: 0201100886.

[25] Andres Duque, Juan Martinez-Romo y Lourdes Araujo. "Unsupervised extraction of adverse drug reaction relationships". En: *Procesamiento de Lenguaje Natural* 55 (sep. de 2015), págs. 83-90.

[26] Renata Kabiljo, Andrew B Clegg y Adrian J Shepherd. "A realistic assessment of methods for extracting gene/protein interactions from free text". En: *BMC Bioinformatics* 10.1 (2009), pág. 233.

[27] Olivier Bodenreider. "The Unified Medical Language System (UMLS): Integrating Biomedical Terminology". En: *Nucleic acids research* 32 (feb. de 2004), págs. D267-70. DOI: 10.1093/nar/gkh061.

[28] Arantza Casillas, Koldo Gojenola, Alicia Pérez y Maite Oronoz. "Clinical text mining for efficient extraction of drug-allergy reactions". En: dic. de 2016, págs. 946-952. DOI: 10.1109/BIBM.2016.7822651.

[29] Iakes Goenaga, Sergio Santana, Sara Santiso Gonzáles, Koldo Gojenola, Alicia Pérez y Arantza Casillas. "IxaMed at eHealth-KD Challenge 2019: Using Different Paradigms to Solve Clinical Relation Extraction." En: *IberLEF@ SEPLN*. 2019, págs. 43-50.

[30] Thien Huu Nguyen y Ralph Grishman. "Relation Extraction: Perspective from Convolutional Neural Networks". En: *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*. 2015, págs. 39-48.

[31] Masaki Asada, Makoto Miwa y Yutaka Sasaki. "Extracting Drug-Drug Interactions with Attention CNNs". En: *BioNLP 2017* (2017), págs. 9-18.

[32] Víctor Suárez-Paniagua, Renzo Rivera, Isabel Segura-Bedmar y Paloma Martinez. "A two-stage deep learning approach for extracting entities and relationships from medical texts". En: *Journal of Biomedical Informatics* 99 (sep. de 2019), pág. 103285. DOI: 10.1016/j.jbi.2019.103285.

[33] Alan Aronson. "Effective Mapping of Biomedical Text to the UMLS Metathesaurus: The MetaMap Program". En: *Proceedings / AMIA Symposium* (feb. de 2001), págs. 17-21.

[34] Viviana Cotik, Darío Filippo, Roland Roller, Hans Uszkoreit y Feiyu Xu. "Annotation of Entities and Relations in Spanish Radiology Reports". En: *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*. 2017, págs. 177-184.

[35] Vanesa Stricker. "Detección de Negaciones en Informes Radiológicos escritos en Español". En: *Tesis de Licenciatura. Directora: Viviana Cotik*. 2016.

[36] Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou y Jun'ichi Tsujii. "BRAT: a Web-based Tool for NLP-Assisted Text Annotation". En: *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics. 2012, págs. 102-107.

[37] Radim Řehůřek y Petr Sojka. "Software Framework for Topic Modelling with Large Corpora". English. En: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta: ELRA, mayo de 2010, págs. 45-50.

[38] Piotr Bojanowski, Edouard Grave, Armand Joulin y Tomas Mikolov. *Enriching Word Vectors with Subword Information*. 2016. arXiv: 1607.04606 [cs.CL].

[39] Steven Bird y Edward Loper. "NLTK: The Natural Language Toolkit". En: *Proceedings of the ACL Interactive Poster and Demonstration Sessions*. Barcelona, Spain: Association for Computational Linguistics, jul. de 2004, págs. 214-217.

[40] Cristian Cardellino. *Spanish Billion Words Corpus and Embeddings*. March 2016. URL: https://crscardellino.github.io/SBWCE/ (visitado 01-07-2020).

[41] Felipe Soares, Marta Villegas, Aitor Gonzalez-Agirre, Martin Krallinger y Jordi Armengol-Estapé. "Medical Word Embeddings for Spanish: Development and Evaluation". En: *Proceedings of the 2nd Clinical Natural Language Processing Workshop*. Minneapolis, Minnesota, USA: Association for Computational Linguistics, jun. de 2019, págs. 124-133. DOI: 10.18653/v1/W19-1916.

[42] Abel Laerte Packer. "SCIELO-An Electronic Publishing Model for Developing Countries". En: *ELPUB*. 1999.

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser e Illia Polosukhin. "Attention Is All You Need". En: *Advances in Neural Information Processing Systems*. 2017, págs. 5998-6008.

Apéndice

A. REPETICIONES POR ENTIDAD

Entidad	Relacionados	No relacionados	Total	%
alteraciones	132	10	142	92.96 %
retroperitoneo vascular	107	8	115	93.04 %
higado	51	24	75	68.00 %
liquido libre	48	26	74	64.86 %
bazo	42	24	66	63.64 %
dilatacion	34	23	57	59.65 %
dilatada	32	22	54	59.26 %
rinones	34	20	54	62.96 %
vejiga	24	19	43	55.81 %
cavidad	32	7	39	82.05 %
vesicula biliar	18	11	29	62.07 %
pancreas	5	22	27	18.52 %
rinon izquierdo	14	13	27	51.85 %
aumentado de tamano	22	4	26	84.62 %
via biliar intra y extrahepatica	14	11	25	56.00 %
via biliar	8	14	22	36.36 %
esplenomegalia	11	8	19	57.89 %
aumento de la ecogenicidad	16	2	18	88.89 %
rd	2	16	18	11.11 %
asas intestinales	13	1	14	92.86 %
meteorismo	3	11	14	21.43 %
pelvis	9	4	13	69.23 %
ri	0	13	13	0.00 %
rinon derecho	8	5	13	61.54 %
apendice cecal	6	5	11	54.55 %
piloro	8	3	11	72.73 %
retroperitoneo	8	3	11	72.73 %
adenomegalias	4	6	10	40.00 %
asas	8	1	9	88.89 %
fosa iliaca derecha	5	4	9	55.56 %
grasa mesenterica	6	3	9	66.67 %
aorta	0	8	8	0.00 %
dilatadas	4	4	8	50.00 %
disminuido de tamano	7	1	8	87.50 %
imagen quistica	2	6	8	25.00 %
litiasica	7	1	8	87.50 %
ovarios	5	3	8	62.50 %
via urinaria	7	1	8	87.50 %
abdomen	1	6	7	14.29 %
bazo accesorio	1	6	7	14.29 %

Tab. A.1: Repetidos por entidad. Se muestra para cada entidad cuántas veces se repite como parte de una relación y cuántas veces aparece sin ser parte de una relación. Este análisis se realizó sobre el conjunto de desarrollo, sin oraciones repetidas.

B. REPETICIONES POR PALABRA

Palabra	Relacionados	No relacionados	Total	%
alteraciones	132	10	142	92.96 %
de	99	41	140	70.71 %
retroperitoneo	116	11	127	91.34 %
vascular	109	9	118	92.37 %
biliar	50	40	90	55.56 %
liquido	58	31	89	65.17 %
higado	51	29	80	63.75 %
via	46	31	77	59.74 %
libre	48	27	75	64.00 %
bazo	43	30	73	58.90 %
tamano	50	16	66	75.76 %
dilatacion	41	23	64	64.06 %
dilatada	33	23	56	58.93 %
rinones	34	20	54	62.96 %
ecogenicidad	38	15	53	71.70 %
rinon	27	26	53	50.94 %
aumento	37	13	50	74.00 %
la	35	12	47	74.47 %
cavidad	36	10	46	78.26 %
vejiga	24	19	43	55.81 %
aumentado	32	8	40	80.00 %
asas	33	6	39	84.62 %
izquierdo	22	16	38	57.89 %
y	24	13	37	64.86 %
con	23	12	35	65.71 %
intra	19	13	32	59.38 %
vesicula	19	12	31	61.29 %
pelvis	18	11	29	62.07 %
derecho	16	12	28	57.14 %
en	16	12	28	57.14 %
extrahepatica	16	11	27	59.26 %
pancreas	5	22	27	18.52 %
epiplon	4	21	25	16.00 %
contenido	16	7	23	69.57 %
imagen	8	15	23	34.78 %
intestinales	20	2	22	90.91 %
disminuido	15	6	21	71.43 %
rd	2	19	21	9.52 %
apendice	9	11	20	45.00 %
ectasia	8	12	20	40.00 %

Tab. B.1: Repetidos por palabra. Se muestra para cada palabra cuántas veces es parte o es una entidad que forma parte de una relación y cuántas veces aparece sin ser parte de una relación. Este análisis se realizó sobre el conjunto de desarrollo, sin oraciones repetidas.