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

Detección de epidemias en textos periodísticos escritos en español

Tesis de Licenciatura en Ciencias de la Computación

Claudia Antonella Dellanzo

Directora: Viviana Cotik

Buenos Aires, septiembre de 2021

RESUMEN

Existen diversas enfermedades que se encuentran presentes tanto en el mundo como en Latinoamérica, siendo algunas de ellas: Chagas, dengue, Guillain-barré, Zika, microcefalia, sarampión y hantavirus. La disponibilidad de información sobre las mismas, como ser la cantidad de casos y la ubicación geográfica en que se manifiestan, es crucial para estudiar cómo se propagan y para contar con más herramientas que permitan tomar medidas para disminuir su incidencia.

Para poder extraer información sobre epidemias en desarrollo en Latinoamérica necesitamos contar con los recursos apropiados. Debido a la falta de corpora en español sobre el dominio de brotes de enfermedades, creamos un corpus anotado para la detección de entidades nombradas y relaciones, basado en artículos periodísticos de ProMED-mail. Por un lado, trabajamos con los artículos enteros y, por otro, solo con los títulos de los mismos. Se implementó el coeficiente *kappa de Cohen* para evaluar la consistencia entre las anotaciones, obteniendo un resultado de 0.53.

Implementamos dos algoritmos para la detección de entidades nombradas: uno basado en reglas y otro de redes neuronales profundas basado en la propuesta *Flair*, que utiliza una red *bidirectional long-short term memory (Bi-LSTM)* con *Conditional Random Fields (CRF)*. El corpus anotado se utilizó para testear ambos algoritmos y para entrenar el de redes neuronales. Para la evaluación, se implementó un *F1-score* (match exacto y parcial), obteniendo los mejores resultados con el algoritmo de redes (tanto *micro-averaged* como para la mayor parte de las entidades).

Por último, se implementó un *baseline* para la extracción de relaciones basado en la coocurrencia de entidades nombradas, permitiéndonos establecer vínculos entre las mismas. Para evaluar su desempeño, se implementó un *F1-score* match exacto.

Palabras claves: reconocimiento de entidades nombradas, detección de relaciones, creación de corpus anotado, español, NLP, detección de epidemias, extracción de información.

ABSTRACT

There are several diseases present both in the world and in Latin America, being some of them: Chagas, dengue fever, Guillain-Barré syndrome, zika, microcephaly, measles, and hantavirus. The availability of information about them, such as the number of cases and the location in which they occur, is crucial to understand the propagation of the illnesses and to have more resources that allow taking action to diminish their incidence.

To extract information about diseases outbreaks in progress, we need to have the appropriate resources. Due to the lack of corpora in Spanish about diseases outbreaks, we created an annotated corpus to detect named entities and relations, based on news articles from ProMED-mail. On one side, we worked with the entire articles and, on the other, only with the titles. We evaluated the consistency between the annotations through the implementation of *Cohen's kappa coefficient*, achieving a result of 0.53.

We implemented two algorithms for named entity detection: a rule-based and a deep learning one based on *Flair's* proposal, which uses a *bidirectional long-short term memory* (Bi-LSTM) network with *Conditional Random Fields* (CRF). The annotated corpus was used to test both algorithms and to train the machine learning one. To evaluate their performances, we implemented an *F1-score* (exact and partial match), achieving better results with the neural network's algorithm (both for the micro-average value and for most entities).

Finally, we implemented a *baseline* for relation extraction based on the co-occurrence of named entities, allowing us to establish links between them. To evaluate its performance, we implemented an *F1-score* exact match.

Keywords: named entity recognition, relation detection, annotated corpus creation, Spanish, NLP, epidemic detection, information extraction.

AGRADECIMIENTOS

Quiero agradecer a todas aquellas personas que me ayudaron en este largo proceso. A Vivi, por haberme ofrecido este tema de tesis y ayudarme a concretarlo. Y a todas las personas que nos ayudaron con la creación y anotación del corpus: José, Daniel L., Daniel P., Fernando, Jonathan y Alexander.

A mi novio Juan y su familia, que me bancaron durante la cuarentena mientras hacía la tesis.

A mi familia, que me dieron el espacio necesario para poder concentrarme en esta etapa.

A todos aquellos que estuvieron a mi lado durante la carrera, a aquellos que todavía están y a los que perdí en el camino.

Y, finalmente, a mi papá, que siempre me apoyó en todos mis sueños y nunca dudó de mí, junto a quien desearía estar compartiendo este logro.

A mi papá: fuiste tan breve y tan para siempre.

Índice general

Índice de figuras

Índice de cuadros

1. INTRODUCCIÓN

Existen diversas enfermedades que han cobrado relevancia en los últimos años debido a la tasa de personas infectadas, como el dengue, el síndrome de Guillain-Barré, el Zika y la microcefalia, que afectan a regiones tropicales del mundo como así también a Latinoamérica. Otras enfermedades, como el mal de Chagas, llevan siglos de existencia, pero no se han empleado muchos recursos para combatirlas, posiblemente debido a que se encuentran en lugares con baja densidad poblacional y a que los portadores de dicha enfermedad son personas que suelen tener bajos recursos económicos. También existen enfermedades endémicas, como el hantavirus, que aparece recurrentemente en regiones de Argentina y Chile, entre otros países. Finalmente, existen pandemias, como la del COVID-19, que estamos atravesando actualmente.

Todas estas enfermedades tienen varios factores en común: 1) pérdidas humanas o consecuencias irreparables para los portadores de las mismas, 2) pérdidas económicas importantes (como aquellas ocasionadas por la falta de turismo a causa del hantavirus o el aislamiento obligatorio impuesto en 2020 por distintos gobiernos debido al COVID-19) y 3) la falta de información disponible por el estado o las autoridades sanitarias, lo cual dificulta la rápida toma de decisiones.

En este trabajo nos concentramos en atacar la tercera problemática. Conocer la cantidad de casos de determinada enfermedad, quiénes tienen más riesgo de ser atacados por la misma y la ubicación geográfica en donde ocurre el contagio, entre otras cuestiones, es crucial para entender cómo se propaga la misma y tomar acciones que permitan disminuir su esparcimiento, ofreciendo una ayuda adicional a la brindada por los sistemas de vigilancia. Cuando la información se encuentra disponible, a veces es de mala calidad (incompleta, inconsistente, incorrecta, no está disponible públicamente y no es oportuna, entre otros).

Nuestra propuesta se basó en analizar posibles brotes de enfermedades, frecuentes en Latinoamérica, a partir de un conjunto de noticias específicas del tema, con el fin de contribuir con herramientas automatizadas y recursos. Para esto utilizamos ProMED-mail [16], un sistema de informes dedicado, entre otros, a la rápida difusión de información sobre epidemias de enfermedades infecciosas. Los artículos publicados en ProMED-mail han sido editados en base a notas periodísticas de diferentes medios por un equipo interdisciplinario. Una vez recolectados los artículos deseados, se realizó una limpieza y normalización de los datos. A partir de los mismos, se generó un corpus[1] anotado para la detección de entidades nombradas (por ejemplo, enfermedades, fechas, ubicaciones geográficas y causas) y relaciones entre las mismas (por ejemplo, cantidad de casos reportados de cierta enfermedad), en artículos que mencionasen la presencia de Chagas, dengue, sarampión, hantavirus, Guillain Barré, zika o microcefalia en países latinoamericanos.

Se implementaron algoritmos para extraer automáticamente información de los artículos con el fin de detectar la presencia de enfermedades, la cantidad de casos reportados, la ubicación geográfica en donde ocurre la enfermedad, las posibles causas asociadas y la fecha de los reportes, entre otras cosas. Para la detección de entidades nombradas, se

[1] Un corpus (pl. corpora) es una colección de textos o discursos utilizados para un propósito específico.

implementó un algoritmo basado en reglas y otro basado en redes neuronales profundas. Para el primero, presentado en la sección 5.4, se separó un subconjunto de artículos que fue analizado para generar reglas. Estas consistieron en expresiones regulares para detectar ciertas entidades nombradas, y para otras (fechas, ubicaciones geográficas y cantidad de casos) se utilizaron herramientas de procesamiento del lenguaje natural. El segundo algoritmo, presentado en la sección 5.5, está basado en la propuesta de Akbik, Blythe y Vollgraf [2], que utiliza un módulo etiquetador de secuencias con una arquitectura *bi-directional long-short term memory (Bi-LSTM)*, combinado posteriormente con una capa de *Conditional Random Fields (CRF)*. Además, utiliza *embeddings*[2] contextualizados a nivel de caracter. Se dividió el conjunto de artículos del corpus anotado en entrenamiento y *test*. Luego, se realizó *five-fold cross validation* para seleccionar los parámetros óptimos, entrenando finalmente el algoritmo sobre el conjunto de entrenamiento con los parámetros seleccionados. Para evaluar el desempeño de ambos algoritmos, se implementó la medida *F1-score micro average* a través de un *match exacto* y un *match parcial*. Este último nos permitió contabilizar aquellas entidades que no tuvieron una coincidencia total en las anotaciones.

También se trabajó en la tarea de extracción de relaciones, que consiste en extraer relaciones semánticas entre las entidades de los artículos, tarea denominada *extracción de relaciones* (RE). Se desarrolló un *baseline*[3] basado en la co-ocurrencia de entidades nombradas con algunas reglas adicionales. Este se presenta en la sección 6.4 y nos permite responder otro tipo de interrogantes como, por ejemplo, cuántos casos de cierta enfermedad fueron reportados en determinada ubicación geográfica. También se implementó la medida de calidad *F1-score micro average* a través de un match exacto para medir el desempeño del algoritmo.

Como parte del trabajo escribimos una publicación en la *Conference on Computational Natural Language Learning (CoNLL)* de la *Special Interest Group on Natural Language Learning (SIGNLL)*, Association of Computational Linguistics (ACL) [24], presentando una primera versión del corpus anotado (ver capítulo 4).

El trabajo está organizado en siete capítulos y un apéndice. En el capítulo 2, se presentan distintos trabajos relacionados con el presente. En cada capítulo, además, se realiza una breve introducción al tema y se presentan los trabajos existentes relacionados con lo desarrollado en cada uno. En el capítulo 3, se desarrollan los conceptos teóricos necesarios para comprender lo propuesto en este trabajo. En el capítulo 4, se describe la generación del corpus anotado. Se detalla su forma de creación, los esquemas, los criterios y el proceso de anotación del mismo, y distintas medidas implementadas para evaluar la consistencia entre las anotaciones realizadas. En el capítulo 5, se presentan dos métodos para realizar reconocimiento de entidades nombradas: uno basado en reglas y otro basado en redes neuronales profundas. Además, se detalla la implementación de las medidas de calidad para evaluar el desempeño de los algoritmos, utilizando un *F1-score micro average* exacto y uno parcial. En el capítulo 6, se presenta el método para extraer relaciones automáticamente junto con la medida de calidad implementada para evaluar su desempeño (*F1-score micro average* exacto). En el capítulo 7, se presentan las conclusiones obtenidas en este trabajo junto a posibles trabajos a realizar en el futuro. Finalmente, en el apéndice, se presentan

[2] Un embedding es una técnica para representar texto a través de vectores de números reales.

[3] Un algoritmo *baseline* ofrece una solución básica para un problema, siendo la base para algoritmos más complejos.

las siglas y abreviaturas utilizadas a lo largo del trabajo, información sobre cómo interpretar un tipo de gráfico utilizado (el gráfico violín), ejemplos de términos para las entidades de tipo *Past* y *Conditional* que se obtuvieron con el algoritmo de reglas utilizando reconocimiento de tiempos verbales[4] y, finalmente, detalles de implementación. Dentro de este último, se presenta cómo se implementó el κ de *Cohen* para medir la consistencia entre las anotaciones, qué herramientas de procesamiento del lenguaje natural se evaluaron para procesar texto en el trabajo, cómo se integran *gazetteers*[5] en las distintas herramientas y cómo se detectaron ubicaciones geográficas en los títulos de los artículos[6].

[4] Esta técnica fue descartada posteriormente debido a que producía resultados no eficaces.

[5] Un *gazetteer* es un diccionario o directorio geográfico con información sobre lugares.

[6] Como será explicado posteriormente, esto fue crucial ya que define si un artículo se anota o no, por lo que se aplicaron medidas para asegurarnos la detección de las ubicaciones geográficas en los títulos.

2. TRABAJOS PREVIOS

A continuación, se presentan los trabajos previos generales para este trabajo. Luego, en cada capítulo específico, se presentan los trabajos previos correspondientes a cada tema.

La vigilancia digital ha sido un tema importante en la salud pública en los últimos años, ya que el acceso público a una alta cantidad de datos digitales permite estudiar patrones sobre cómo se desarrollan y se generan distintas enfermedades. Esto nos permite promover la salud, ayudando a prevenir enfermedades y mitigar sus consecuencias. Existe un gran número de trabajos publicados sobre la vigilancia epidemiológica[7] basados en la gran disponibilidad de datos existentes en formato digital [64]. Estos datos pueden ser tanto de redes sociales como de las historias clínicas electrónicas. Otras investigaciones muestran que las redes sociales podrían ser herramientas valiosas para detectar brotes de enfermedades debido a que pueden ser más rápidas que los métodos tradicionales [17]. Esto nos permite tener información en tiempo real. Por ejemplo, Twitter ha sido utilizado para el seguimiento y localización de enfermedades, demostrando obtener resultados positivos [26]. Por lo tanto, la vigilancia de la salud pública es una aplicación natural para las técnicas de inteligencia artificial, requiriendo técnicas de Procesamiento del Lenguaje Natural (NLP) para extraer información de los datos de la web.

En este contexto, existen sistemas informales de vigilancia de enfermedades que pueden, a falta de información oficial, ser útiles para entender el curso de ciertas enfermedades [25]. Existen sistemas como ProMED-mail [16], un sistema de reportes dedicado a la diseminación rápida de información de epidemias de enfermedades infecciosas y exposición a toxinas que afectan a la salud humana. Este será utilizado en este trabajo para construir nuestro corpus. Regiones y países podrían beneficiarse complementando sus sistemas de inspección digital sin diagnóstico utilizando la herramienta de ProMED-mail [54]. Cabe destacar la eficacia del mismo como fuente de datos epidemiológicos centrándose en el coronavirus [10]. Mientras que ProMED es una fuente confiable de información, no está equipado para proveer detalles de datos epidemiológicos. Por ejemplo, no suele reportar normalmente la cantidad de casos o muertes, más allá de lo mencionado en los artículos. Si se quieren extraer datos sistemáticamente, como la cantidad de casos de una enfermedad, se debe realizar un análisis sobre los mismos [16].

Un trabajo importante en esta dirección es el de *Platform for Automated extraction of Disease Information from the web (PADI-web)* [4]. Esta herramienta genera información epidemiológica sobre enfermedades, ubicaciones geográficas, fechas, portadores de las enfermedades y cantidad de casos nombrados en textos periodísticos y artículos de redes sociales. Para realizar esto, combina técnicas basadas en reglas y minería de datos[8] para la extracción de información. También existen algoritmos de inteligencia artificial entrenados para poder clasificar o identificar información sobre enfermedades. Esta propuesta es la que mayor similitud presenta a la nuestra, pero con la diferencia de que, debido a la escasez de corpus sobre el área de interés en idioma español, nosotros nos enfocamos en crear

[7] La vigilancia epidemiológica es un proceso continuo que consiste en la recolección de datos relacionados a la salud pública para su análisis, interpretación y divulgación.

[8] La minería de datos es un proceso que busca descubrir patrones en grandes conjuntos de datos utilizando técnicas de aprendizaje automático, estadísticas y bases de datos.

un corpus con artículos provenientes de ProMED-mail sobre enfermedades prevalentes en América Latina.

3. MARCO TEÓRICO

En este capítulo, se presenta el marco teórico necesario para comprender el desarrollo de este trabajo. Se comienza con una breve introducción sobre el procesamiento del lenguaje natural. Luego, se definen el mismo junto a una serie de tareas importantes, como las de extracción de información. Posteriormente, se explican distintos conceptos sobre redes neuronales y se desarrolla sobre distintas arquitecturas existentes. Finalmente, se presentan distintas medidas de calidad que existen para evaluar el desempeño de los algoritmos de aprendizaje automático.

3.1. Introducción

A partir de la década de 1950, comenzaron a aparecer diversos estudios dentro del área del procesamiento del lenguaje natural (NLP por sus siglas en inglés). En 1950, Alan Turing publicó un artículo llamado *Computing Machinery and Intelligence* [65], en el cual propuso un *Turing test* o, como fue llamado originalmente, un *Juego de imitación* (*The imitation game*), el cual busca demostrar la capacidad de una máquina de pensar. El juego consiste en un interrogador que debe analizar una conversación entre un humano y una máquina, y descubrir cuál es cuál. La conversación puede estar dada de manera escrita (por ejemplo, a través de una teleimpresora) o puede ser repetida por un intermediario, para así no depender de la capacidad de la máquina de renderizar la voz humana. En el caso en el que el interrogador no sea capaz de distinguir a la máquina del humano, se dice que la máquina pasó el test.

Poco tiempo después, en 1952, *Hodgkin y Huxley* presentaron un modelo matemático que simula el funcionamiento de las neuronas en el cerebro [33], inspirando los desarrollos de inteligencia artificial y procesamiento del lenguaje natural.

Luego de 1980, se comenzaron a utilizar los modelos estadísticos para crear modelos de lenguaje para realizar tareas como reconocimiento de voz [5] o traducción automática [12], utilizando por ejemplo modelos de *n-gramas* para la predicción de palabras [13]. Previo a esto, existían sistemas basados en reglas hechas a mano, como el programa Eliza [67] o SHRDLU [68].

En 1997, se introdujeron los modelos de redes neuronales recurrentes *long-short term memory* [32] que serán utilizados en las siguientes décadas para diversas tareas, como el procesamiento del habla [30, 55], parsing [6], problemas de etiquetados de secuencias [3], análisis de sentimientos [66], etc.

Luego del año 2010, los métodos de *deep learning* (aprendizaje profundo) se extendieron al procesamiento del lenguaje natural y demostraron resultados vanguardistas en diversas tareas de NLP [37, 18, 47]. Algunas de las mismas que podemos realizar gracias a estos modelos son *question answering* (búsqueda de respuestas)[9] [72], traducción automática [20] y *named entity recognition* [35].

En este capítulo, definimos una serie de conceptos del procesamiento del lenguaje na-

[9] El *question answering* es un tipo de recuperación de información que permite a un sistema recuperar respuestas a preguntas planteadas en lenguaje natural basándose en una cantidad de documentos (como la World Wide Web).

tural que serán utilizados a lo largo de este trabajo. En la sección 3.2 introducimos qué es el procesamiento del lenguaje natural y desarrollamos ciertos conceptos importantes del mismo: segmentación de oraciones, tokenización, normalización, lematización, *stemming*, *Part-of-Speech tagging*, modelos de lenguaje, n-gramas, *word embeddings* y *conditional random fields*. Luego, en la sección 3.3, introducimos el concepto de extracción de información junto con algunas de las tareas que involucra: reconocimiento de entidades nombradas y extracción de relaciones. En la sección 3.4, hacemos un repaso por distintos conceptos de redes neuronales, explicando qué son, qué es una neurona, cuáles son las distintas arquitecturas que existen, cómo aprende una red neuronal, qué es una red neuronal bidireccional y hablamos de una arquitectura particular: *long short-term memory* y, su variante, *bidirectional long short-term memory*. Finalmente, en la sección 3.5, presentamos las distintas formas que existen para medir la calidad de un modelo de aprendizaje: *classification accuracy*, matriz de confusión y *F1-score*.

3.2. Procesamiento del lenguaje natural

El *procesamiento del lenguaje natural*, es un subcampo de las ciencias de la computación que utiliza técnicas computacionales de inteligencia artificial para procesar y comprender el lenguaje humano hablado y escrito. Esta definición abarca tareas triviales como contar palabras, a aplicaciones innovadoras como traducción de lenguaje hablado en tiempo real o *chatbots*[10].

Jurafsky y Martin sostienen que lo que distingue estas aplicaciones de procesamiento del lenguaje de otras tareas de procesamiento de datos es que utilizan el *conocimiento del lenguaje*. Supongamos que tenemos un programa que se encarga de leer bytes[11] y palabras. Para leer bytes, solo se necesita una aplicación común de procesamiento de datos, pero para procesar las palabras, se necesita saber qué significa ser una palabra, convirtiéndose en un sistema de procesamiento del lenguaje. Si, además, el programa quiere entender el significado de las palabras, necesita conocer su semántica. [39]

Segmentación de oraciones

La segmentación de oraciones es un proceso por el cual se divide un texto en oraciones. Uno de los aspectos que hacen que esta tarea no sea sencilla es la presencia de los signos de puntuación, que pueden indicar el fin de una oración, la abreviación de una palabra (por ejemplo, *Dra.* o *Sra.*) o expresar cantidades numéricas (por ejemplo, el número *1.000*). Para lidiar con este problema, se puede poseer una lista de abreviaciones y otra con los signos de puntuación que pueden indicar el fin de una oración (símbolos como '.' o '?'). La dificultad de esto es que esta lista varía según cada idioma.

Tokenización

La tokenización es una tarea que consiste en separar partes de un texto en unidades más pequeñas llamadas *tokens*. Esta forma de estructuración de los datos en tokens es utilizada por las herramientas de NLP para realizar cómputos sobre los mismos como, por ejemplo, para la segmentación de oraciones. Alguna de las maneras que existen para tokenizar el texto son:

[10] Los *chatbots* son aplicaciones capaces de llevar a cabo conversaciones automáticamente.
[11] Un byte es una unidad de memoria de las computadoras

- *Tokenización en palabras:* Consiste en separar el texto en palabras, siendo la forma más común de tokenización. Se puede separar utilizando un caracter como delimitador, como el *espacio* en el idioma español (por ejemplo, la oración "El oso panda" quedaría tokenizada en 3 palabras: ['El', 'oso, 'panda']). Sin embargo, es difícil definir qué es una palabra, ya que varía según el idioma. Por ejemplo, en el idioma chino, no se utilizan espacios entre las palabras. Incluso en lenguajes en los que sí se utilizan los espacios, existen casos particulares que deben ser considerados (por ejemplo, en el idioma español existen contracciones como *"del"* que se refiere a las palabras *"de el"*, o en el idioma inglés existe la *hyphenation* de palabras como, por ejemplo, "state-of-the-art").

- *Tokenización por caracter:* Consiste en separar partes de un texto en un conjunto de caracteres (por ejemplo, la palabra "hola" quedaría tokenizada en 4 caracteres: ['h', 'o', 'l', 'a']). Este tipo de tokenización demostró ser útil en distintas tareas como en traducción automática, como se puede observar en el trabajo de Lee, Cho y Hofmann [46], o para análisis de sentimientos, como se puede observar en el trabajo de Radford, Jozefowicz y Sutskever [52].

- *Tokenización en subpalabras:* Consiste en mantener aquellas palabras con alta frecuencia tal como son y dividir aquellas palabras raras o pocos frecuentes en subpalabras que existen dentro de un vocabulario seleccionado (por ejemplo, la palabra en inglés *"playing"* se separaría en ['play', 'ing']). Algunos modelos conocidos para realizar lo mismo son *Byte-Pair Encoding* [58] y *WordPiece* [57]. Esta forma permite tener un vocabulario de tamaño razonable y procesar palabras que no reconoce en subpalabras que sí. Este tipo de tokenización es muy útil para idiomas como el *alemán* en el que se arman palabras complejas y largas a través de la unión de subpalabras.

Normalización

La normalización es un proceso que consiste en transformar el texto a su forma canónica para que su procesamiento sea más fácil, entre otras razones. Existen muchos aspectos sobre los cuales se pueden aplicar la normalización: remover espacios o signos de puntuación duplicados, remover o sustituir caracteres especiales como emojis, sustituir las contracciones, expandir abreviaturas y acrónimos (por ejemplo, transformar "EE.UU." en "Estados Unidos"), correcciones ortográficas (por ejemplo, corregir palabras en textos de redes sociales), remover géneros de palabras con lematización o stemming, entre otros.

Lematización

La lematización es una forma de normalización que consiste en reducir una palabra a su *lema* o *forma canónica*. El lema sería la forma base en la que encontraríamos la palabra en el diccionario tradicional: singular para sustantivos, masculino singular para adjetivos e infinitivo para verbos (por ejemplo, el lema de la palabra *gatas* sería *gato*). Algunas de las maneras de realizar la lematización son mediante análisis morfológico o mediante análisis sintáctico. En el primero de ellos, se busca determinar la categoría de cada palabra en una oración, por lo que en el caso de las palabras homógrafas[12], alguna de ellas van

[12] Las palabras homógrafas son aquellas que se escriben y pronuncian igual, pero tienen más de un significado.

a tener más de un lema (por ejemplo, la palabra *"vino"* puede referirse al sustantivo o al verbo *venir*). En cambio, en los análisis sintácticos, se busca determinar la función de cada palabra dentro de una oración teniendo en cuenta su contexto (por ejemplo, en la oración *"Juan toma vino"*, *vino* sería un sustantivo).

Stemming

El *stemming* es un proceso que consiste en reducir palabras a su *stem* (*tema*) o raíz (por ejemplo, el stem de la palabra "biblioteca" es "bibliotec"). Desde el punto de vista del procesamiento, el stemming es más rápido que la lematización. Además, permite reconocer relaciones entre palabras de distintas clases (por ejemplo, el stem de "picante" y "picar" es "pic"). Una desventaja del stemming es que sus algoritmos son muchos más simples que la lematización y corre el riesgo de recortar demasiado la raíz de una palabra, llevando a encontrar relaciones entre palabras que no existen (*overstemming*). También puede suceder lo opuesto, que deje las raíces demasiado extensas o específicas (*understemming*), de forma tal que palabras que deberían tener la misma raíz no la tengan.

Part-of-Speech (PoS) tagging

El *part-of-speech tagging* (etiquetado gramatical) es un proceso que consiste en asignar etiquetas a cada palabra de un texto en su categoría gramatical, basándose tanto en su definición como en su contexto. Existen diversas categorías y subcategorías que se le pueden asignar a una palabra, que varían según el lenguaje (por ejemplo, en algunos idiomas se considera el género de las palabras y en otros no). Algunas categorías podrían ser sustantivos, verbos, adjetivos, preposiciones, adverbios, etc.; y para un sustantivo sus subcategorías podrían ser plural, posesivo, singular. En la figura 3.1 podemos observar un ejemplo de PoS-tagging utilizando la herramienta FreeLing[13] para la oración *"El martes se detectaron 15 casos de hantavirus."*. Esta es una tarea compleja porque no basta con tener una lista de palabras y sus posibles categorías gramaticales, ya que de acuerdo al contexto en el que se encuentran esto puede variar (como es el caso de las palabras homógrafas que pueden tener distinto significado de acuerdo al contexto en el que se encuentren).

El	martes	se	detectaron	15	casos	de	hantavirus	.
el	[M:??/??/??:??.??:??]	se	detectar	15	caso	de	hantavirus	.
DA0MS0	*W*	*P00CN00*	*VMIS3P0*	*Z*	*NCMP000*	*SP*	*NCMN000*	*Fp*

▼ CONLL format

```
1 El         el                 DA0MS0  DA  pos=determinante|tipo=articulo|gén=masculino|num=singular
2 martes     [M:??/??/??:??.??:??] W    W   pos=fecha
3 se         se                 P00CN00 P0  pos=pronombre|gén=común|num=invariante
4 detectaron detectar           VMIS3P0 VMI pos=verbo|tipo=principal|modo=indicativo|tiempo=pasado|persona=3|num=plural
5 15         15                 Z       Z   pos=número
6 casos      caso               NCMP000 NC  pos=sustantivo|tipo=común|gén=masculino|num=plural
7 de         de                 SP      SP  pos=adposición|tipo=preposición
8 hantavirus hantavirus         NCMN000 NC  pos=sustantivo|tipo=común|gén=masculino|num=invariante
9 .          .                  Fp      Fp  pos=puntuación|tipo=punto
```

Fig. 3.1: Ejemplo de PoS-tagging con FreeLing.

Existen los llamados *tagsets*, que son una recopilación de las etiquetas utilizadas en un corpus. Por ejemplo, para el inglés americano, contamos con el tagset *Penn* o, para el español, el *Spanish FreeLing part-of-speech tagset*, el cual está basado en las etiquetas

[13] Disponible en `http://nlp.lsi.upc.edu/FreeLing/` [Accedido en Junio de 2021].

propuestas por el grupo *EAGLES*, quienes ofrecen las etiquetas para todas las lenguas europeas.

La entrada de un algoritmo de *PoS-tagging* es una lista de palabras con un tagset específico, y la salida es el mejor tag asignado a cada palabra.

Modelos de lenguaje

Un *modelo de lenguaje* asigna una probabilidad a una secuencia de n palabras o caracteres $P(w_1, ..., w_n)$. Estos proveen el contexto necesario para distinguir palabras y frases que suenan similar, o predecir palabras u oraciones enteras. Es utilizado en tareas de NLP como traducción, generación de texto, *PoS-tagging*, parsing, reconocimiento de escritura a mano, recuperación de información, etc. Existen dos tipos de modelos de lenguaje:

- *Modelo de lenguaje estadístico:* Utilizan técnicas estadísticas tradicionales como n-gramas, modelos ocultos de Markov y otras reglas lingüísticas para aprender la probabilidad de la distribución de palabras.
- *Modelo de lenguaje neuronal:* Utilizan distintas técnicas de redes neuronales para modelar el lenguaje (por ejemplo, utilizando *word embeddings*).

Por ejemplo, en un *Neural character-Level Language Model* (modelo de lenguaje neuronal a nivel de caracteres), la idea es predecir, utilizando redes neuronales recurrentes, el siguiente caracter de una secuencia de datos (generando caracter por caracter).

N-gramas

Un *n-grama* es una secuencia contigua de n elementos de una secuencia dada (pueden ser letras, palabras, sílabas, etc.). Un n-grama de tamaño 1 se llama *unigrama*, uno de tamaño 2 *bigrama*, uno de tamaño 3 *trigrama*, etc. Un *modelo de n-grama* es un modelo de lenguaje probabilístico que sirve para realizar una predicción sobre el próximo elemento de una secuencia. Este puede estar definido por una cadena de Markov[14] de orden $n-1$. Formalmente, un modelo de n-gramas predice x_i basado en $(x_{i-(n-1)}, ..., x_{i-1})$.

Word embeddings

Los *word embeddings* son un conjunto de modelos de lenguaje que consisten en representar palabras o frases de un vocabulario a través de vectores de números. La idea es capturar la mayor cantidad de información semántica, morfológica y contexto, entre otras cosas, de forma tal que palabras con significado parecido se encuentren cercanas en el espacio de vectores.

Una de las formas más básicas de transformar palabras en vectores, llamada *one-hot encoding*, se basa en contar la ocurrencia de cada palabra en cada documento. De esta forma, se tiene una matriz en la que las columnas representan palabras y las filas documentos, y se cuenta en cada celda la cantidad de veces que aparece la palabra x en el documento y. La idea es que palabras con un contexto similar ocupen posiciones espaciales cercanas.

[14] La cadena o modelo de Markov es un tipo de proceso estocástico discreto en el que la probabilidad de que ocurra un evento depende solamente del elemento anterior (en nuestro caso, de los $n-1$ elementos anteriores)

Existen otras técnicas más complejas como *Word2vec* que utiliza modelos de redes neuronales para aprender asociaciones de palabras dentro de un corpus. Una vez entrenado, el modelo es capaz de, entre otras cosas, detectar sinónimos entre palabras.

Expresiones regulares

Las *expresiones regulares* (*regex*) son una secuencia de caracteres que especifican un patrón de búsqueda. Son muy útiles a la hora de extraer información de un texto, y algunas de sus aplicaciones más comunes son realizar validaciones (por ejemplo, verificar que un texto contenga solo valores numéricos), *parsing*, reemplazo de texto, etc. Se definen a través de una sintaxis, en la que cada caracter es un metacaracter (por ejemplo, '.' refiere a cualquier caracter menos a un salto de línea) o un caracter regular (por ejemplo, la letra 'a'). En la teoría de lenguajes formal, una expresión regular describe un lenguaje regular. Formalmente, dado el alfabeto Σ[15], una expresión regular se define de manera inductiva de la siguiente forma:

- ε denota el conjunto que solo contiene al string vacío, es decir que no contiene ningún caracter: $L(\varepsilon) = \{\varepsilon\}$
- Para todo caracter a del alfabeto ($\forall a \in \Sigma$), a denota al conjunto que solo contiene al caracter a: $L(a) = \{a\}$
- (RS) denota al conjunto que se obtiene de la concatenación del lenguaje R y S: $L(RS) = L(R)L(S)$. Por ejemplo, sea R el conjunto $\{ab, c\}$ y S el conjunto $\{d\}$, luego $L(RS) = \{abd, cd\}$.
- $(R|S)$ denota al conjunto unión de los conjuntos definidos por R y S: $L(R|S) = L(R) \cup L(S)$. Por ejemplo, sea R el conjunto $\{ab, c\}$ y S el conjunto $\{ab, d\}$, luego $L(R|S) = \{ab, c, d\}$.
- (R^*) denota la clausura de Kleene, que representa al conjunto que se forma de la concatenación finita (inclusive cero veces) de los strings definidos en R. Formalmente, para un lenguaje L, se define como $\bigcup_{i=0}^{\infty} L_i$, siendo L_i la concatenación de L consigo misma i veces. Por ejemplo, sea R el conjunto $\{ab, c\}$, luego R^* denota $\{\varepsilon, ab, c, abab, abc, cab, cc, ...\}$

Conditional Random Fields (CRF)

El *conditional random fields* (CRF) [45] es un método de modelado condicional que sirve para la segmentación y etiquetado de secuencia de datos, entre otras cosas. Son utilizados en aquellas tareas en las que la información del contexto influye en la predicción actual. Son similares a los modelos ocultos de Markov, pero se diferencian en que estos últimos modelan la probabilidad conjunta de secuencias de etiquetas y pares observados ($P(S, O)$), mientras que el CRF modela la probabilidad de la secuencia de etiquetas condicionada a las observadas ($P(S|O)$).

[15] En la teoría de lenguajes, el alfabeto de un lenguaje L está definido como el subconjunto de símbolos que pueden ocurrir en cualquier cadena de caracteres de L.

3.3. Tareas de extracción de información

La *extracción de información* (IE por sus siglas en inglés de *Information Extraction*) es una tarea que consiste en extraer automáticamente información estructurada de documentos o archivos electrónicos. De esta forma, se pueden extraer entidades de los textos y establecer relaciones entre ellas, popular bases de datos para realizar procesamientos, entre otras cosas.

Reconocimiento de entidades nombradas

El *reconocimiento de entidades nombradas* (NER por sus siglas del inglés *Named entity recognition*) es una tarea de extracción de información que consiste en reconocer y clasificar (NERC por sus siglas del inglés *named entity recognition and classification*), entidades nombradas que se encuentran dentro de un texto en categorías predefinidas, como nombres de personas, organizaciones, lugares geográficos, expresiones numéricas, entre otras. Esta tarea juega un rol importante para distintas áreas del NLP y ayuda a responder distintas preguntas del mundo real como, por ejemplo, *qué compañías son mencionadas en cierto artículo* o *qué ubicación geográfica es mencionada*. Dependiendo del dominio del problema y las incógnitas que se busquen resolver, las categorías en las cuales se van a clasificar las entidades varían (en nuestro dominio de detección de epidemias nos va a interesar detectar enfermedades y no organizaciones).

Extracción de relaciones

La *extracción de relaciones* (RE por sus siglas del inglés *relation extraction*) es una tarea dentro de la extracción de información y se encarga de extraer relaciones semánticas de las entidades de un texto. Por ejemplo, si tenemos el texto *"Buenos Aires se encuentra en Argentina"*, tenemos la relación "se encuentra en" que vincula "Buenos Aires" con "Argentina". Esta relación podría estar representada por el trío *(Buenos Aires, se encuentra en, Argentina)*.

3.4. Redes neuronales

Las *redes neuronales* son un modelo computacional basado en la manera en la que el cerebro humano realiza ciertas tareas. Utilizan unidades de procesamiento llamadas *neuronas*, las cuales son capaces de adquirir conocimiento a través de un proceso de aprendizaje y almacenan dicho conocimiento a través de sus conexiones. Tienen la habilidad de *generalización*, lo cual permite generar soluciones para problemas que no fueron vistos durante el proceso de entrenamiento.

Una *neurona* es una célula del sistema nervioso que recibe, procesa y transmite información a las demás células del cuerpo a través de impulsos nerviosos. Están formadas por un cuerpo celular (*soma*) que se encarga del procesamiento. De este surge un denso árbol de ramificaciones (*dendritas*) que es el canal de entrada y una fibra larga (*axón*) que es el canal de salida. En la figura 3.2 puede observarse la estructura de una neurona biológica típica. Los impulsos se procesan en el soma y se transmiten a través del axón, que emite un impulso nervioso hacia las neuronas contiguas. Existe una conexión unidireccional entre las neuronas que se llama *sinapsis*. La transmisión de información se hace de manera eléctrica en el interior de la neurona y de forma química entre neuronas, a través

de una sustancia llamada *neurotransmisores*. Si bien no todas las neuronas son iguales (por ejemplo, pueden tener distinto número de ramificaciones de sus dendritas o distinta longitud del axón), todas operan a través de los mismos principios básicos.

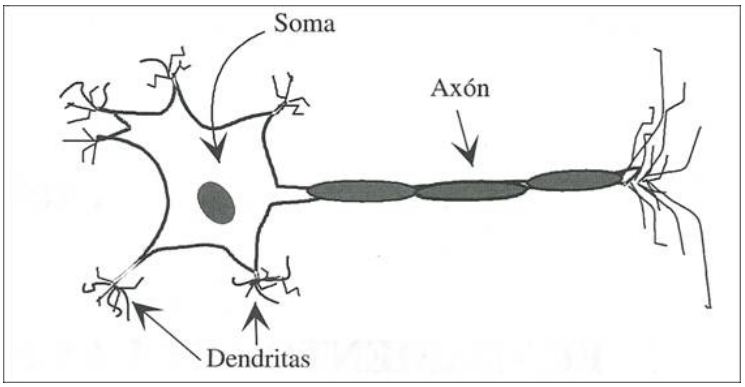


Fig. 3.2: Estructura de una neurona biológica. Gráfico tomado de `http://grupo.us.es/gtocoma/pid/pid10/RedesNeuronales.htm`.

Modelo de una neurona artificial

Una neurona es una unidad de procesamiento de información fundamental para el funcionamiento de una red neuronal. Los elementos básicos del modelo de una neurona se pueden observar en la figura 3.3, y son los siguientes:

- Un conjunto de *sinapsis* o *enlaces de conexiones* caracterizados por un peso. Particularmente, una señal x_j en la entrada de la sinapsis j conectada con la neurona k es multiplicada por el peso sináptico w_{kj}.
- Un *sumador* para las señales entrantes, pesadas por la respectiva sinapsis de la neurona, que retorna una *combinación lineal* de las mismas.
- Una *función de activación* que limita la amplitud de la salida de una neurona a cierto valor finito.
- Un umbral b_k cuya función es incrementar o disminuir la entrada neta de la función de activación, dependiendo de si es positiva o negativa, correspondientemente.

Matemáticamente, podemos describir una neurona k a través del siguiente par de ecuaciones:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{3.1}$$

y

$$y_k = \varphi(u_k + b_j) \tag{3.2}$$

donde $x_1, x_2, ..., x_m$ son las señales entrantes; $w_{k1}, w_{k2}, ..., w_{km}$ son los pesos sinápticos de la neurona k; u_k es la salida del combinador lineal correspondiente de las señales entrantes; b_k es el umbral; $\varphi(\cdot)$ es la función de activación; y y_k es la señal de salida de la neurona.

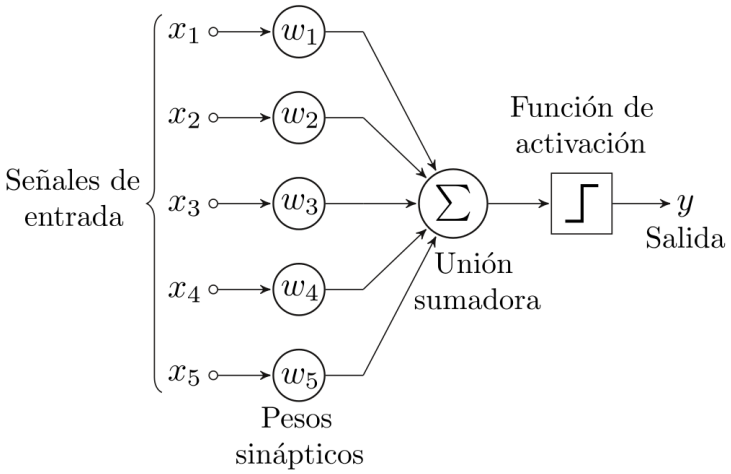


Fig. 3.3: Modelo matemático de una neurona. Gráfico tomado de `https://es.wikipedia.org/wiki/Perceptr%C3%B3n`.

En la mayoría de los modelos, la función de activación es monótona creciente y continua. Algunas de las posibles funciones de activación son (ver figura 3.4 para observar ejemplos):

1. *Escalón:* Es la forma más simple. Consiste en una función binaria que indica si la neurona se activa o no. Está dada por la fórmula:

$$g(v_k) = \begin{cases} 1 & \text{si } v_k \geq 0 \\ 0 & \text{si } v_k < 0 \end{cases}$$

donde:

$$v_k = \sum_{j=1}^{m} w_{kj} x_j + b_k \tag{3.3}$$

2. *Lineal a tramos:* Va incrementando a medida que la entrada aumenta. Está dada por la fórmula:

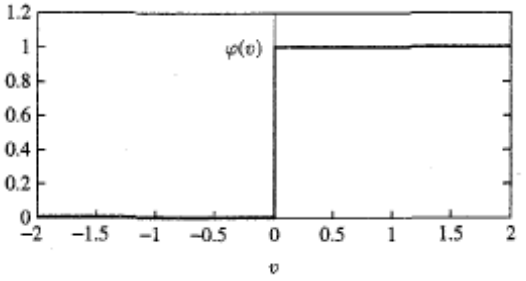
$$g(v_k) = \begin{cases} 1 & \text{si } v_k \geq 0 \\ v_k & \text{si } +\frac{1}{2} > v_k > -\frac{1}{2} \\ 0 & \text{si } v_k \leq -\frac{1}{2} \end{cases}$$

3. *Sigmoidea:* La función sigmoidea, cuyo gráfico tiene forma de S, es la función de activación más común a la hora de construir redes neuronales. Está dada por la fórmula:

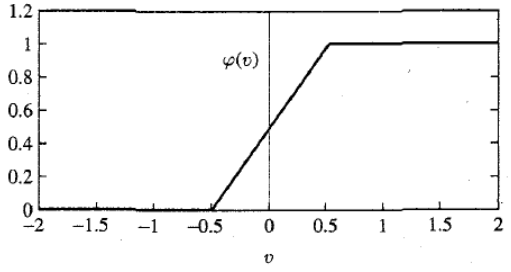
$$g(v_k) = \frac{1}{1 + e^{-av_k}} \tag{3.4}$$

en donde a es la pendiente de la función. A medida que la pendiente se acerca al infinito, la función se convierte en una función escalón. Se diferencian en que, mientras la función escalón tiene únicamente valores 0 o 1, la sigmoidea tiene valores que van desde 0 a 1. Y también en que la función sigmoidea es diferenciable, mientras que la función escalón no lo es. Una de las funciones sigmoideas comúnmente usada es la *tangente hiperbólica*, dada por la fórmula:

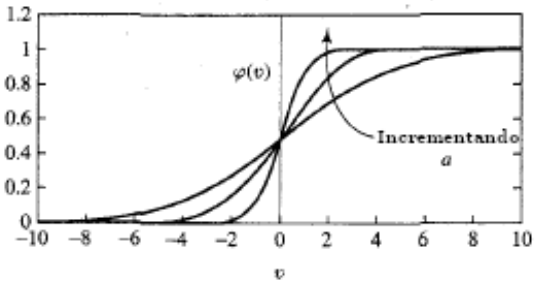
$$g(v_k) = tanh(v_k) \tag{3.5}$$



(a) Función escalón



(b) Función lineal a tramos



(c) Función sigmoidea con parámetro de pendiente variante a

Fig. 3.4: Distintos tipos de funciones de activación. Gráficos adaptados de Kubat [44].

Arquitectura de las redes neuronales

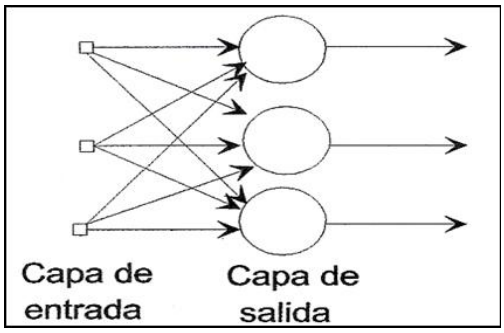
Se denomina *arquitectura* a la estructura en la que las distintas neuronas de la red neuronal se asocian, lo cual se establece según el algoritmo de aprendizaje utilizado para entrenar la red.

Una *red neuronal prealimentada* (*feedforward neural network*) es una red neuronal en la que las neuronas se encuentran organizadas en *capas* (*layers*) y las conexiones entre las neuronas no forman ciclos, por lo que la información solo se mueve en una dirección: desde los nodos de entrada hacia los nodos de salida. Existen dos tipos diferentes de esta arquitectura (Ver figura 3.5):

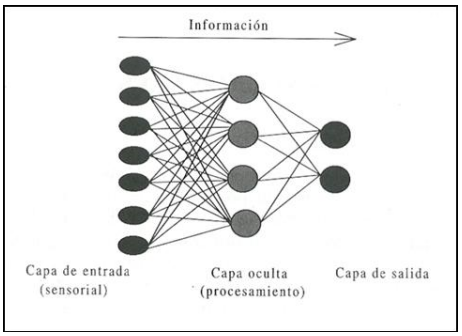
1. *Perceptrón de una capa (single-layer feedforward network):* Existe únicamente una capa de entrada que se encuentra directamente conectada a una sola capa de salida.

2. *Perceptrón multicapa (multilayer feedforward network):* Contienen una o más capas ocultas, cuyos nodos se llaman *neuronas ocultas* o *unidades ocultas*, que intervienen

entre la capa externa de entrada y la capa de salida. De esta forma, la red es capaz de extraer estadísticas más complejas.



(a) Red neuronal de una capa



(b) Red neuronal multicapa

Fig. 3.5: Distintos tipos de redes neuronales *feedforward*. Gráficos tomados de `http://grupo.us.es/gtocoma/pid/pid10/RedesNeuronales.htm`.

También se encuentran las *redes neuronales recurrentes* (*RNN*), que se diferencian de las *feedforward* en que tienen, al menos, un bucle de retroalimentación (*feedback loop*). Estos bucles pueden ser entre neuronas de diferentes capas, neuronas de la misma capa o entre una misma neurona (ver figura 3.6).

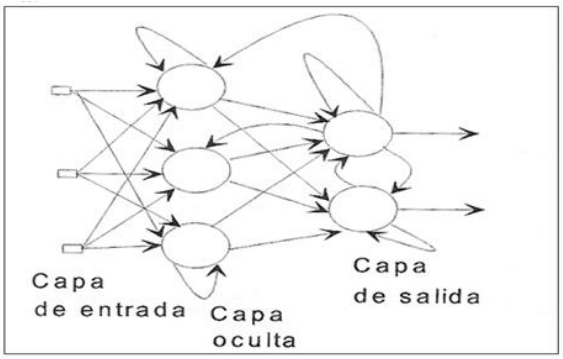


Fig. 3.6: Red neuronal recurrente. Gráfico tomado de `http://grupo.us.es/gtocoma/pid/pid10/RedesNeuronales.htm`.

Aprendizaje de las redes neuronales

Se define como *aprendizaje* al proceso a través del cual se modifican los valores de los parámetros de la red de acuerdo a los estímulos que recibe. El tipo de aprendizaje está determinado por la manera en la que los parámetros cambian. Se llama *algoritmo de aprendizaje* a un conjunto de reglas definidas para solucionar un problema de aprendizaje. También se encuentra el *paradigma de aprendizaje*, que es el modelo que utiliza la red para operar. [59]

Dos de los paradigmas de aprendizaje son el *supervisado* y el *no supervisado*. El *supervisado* se basa en la existencia de un conjunto de ejemplos *entrada-salida* que se llama *conjunto de entrenamiento*. El algoritmo de aprendizaje analiza este conjunto de datos e

infiere una función, que será utilizada para producir salidas para instancias no vistas antes (generalización). Para inferir la función, el algoritmo ajusta iterativamente los pesos para minimizar la diferencia entre el resultado obtenido y el esperado de la red (el error). En el enfoque *no supervisado*, la red no cuenta con ejemplos de entrada-salida, sino que infiere patrones de datos no etiquetados.

Uno de los algoritmos utilizados para minimizar el error en una red neuronal *feedforward* es el *back-propagation* (*propagación hacia atrás de errores* o *retropropagación*). Este utiliza el error cuadrático de la salida de la neurona j en la iteración n, que se encuentra definida por:

$$e_j(n) = d_j(n) - y_j(n) \tag{3.6}$$

donde d es el resultado esperado e y es la predicción del modelo. Luego, se puede definir el *error total* sobre todas las neuronas en la capa de salida como:

$$\xi(n) = \frac{1}{2}\sum_{j=1}^{M} e_j^2(n) = \frac{1}{2}\sum_{j=1}^{M}(d_j(n) - y_j(n))^2 \tag{3.7}$$

donde M es la cantidad de neuronas de la capa de salida de la red. Una medida de error comúnmente utilizada es el *error cuadrático medio* (ECM), que se basa en promediar el error de todos los patrones (ejemplos) de una iteración, y está dado por la fórmula:

$$\xi_{av} = \frac{1}{N}\sum_{n=1}^{N} \xi(n) \tag{3.8}$$

donde N es la cantidad de patrones contenidos en el conjunto de entrenamiento.

El objetivo del proceso de aprendizaje es ajustar los parámetros libres de la red de forma tal de minimizar ξ_{av}.

El proceso de aprendizaje se mantiene en una base de *época-por-época*[16] hasta que los valores del peso sináptico o del umbral de la red se estabilizan, y el ECM converge a cierto valor mínimo. Para un determinado conjunto de entrenamiento, el aprendizaje de retroprograpagación puede ser de las siguientes maneras:

- *Modo secuencial, estocástico u online:* La actualización de los pesos se realiza después de la presentación de cada par de entrenamiento. Sea una época que contiene N datos de entrenamiento en el orden $(x(1), d(1), ..., x(N), d(N))$, el primer ejemplo $(x(1), d(1))$ es presentado a la red, donde $x(i)$ es el i-ésimo elemento del vector de ejemplos de entrada y $d(i)$ es el resultado esperado para la entrada $x(i)$. Luego se computa el paso *forward* y *backward* (que serán explicados posteriormente), y se realiza el ajuste correspondiente a los pesos sinápticos y al umbral. Luego, se presenta el siguiente par de ejemplos y así hasta que se presente el par $(x(N), d(N))$.

- *Batch:* La actualización de los pesos se realiza después de la presentación de todos los ejemplos de entrenamiento que constituyen una época.

[16] Una época es una presentación completa del conjunto de entrenamiento durante el proceso de aprendizaje.

- *Mini-batch:* Se divide el conjunto de datos en una cantidad fija de *batches* (definidos por el valor del *mini-batch*). Se presentan por época cada uno de estos subconjuntos y se actualizan los pesos.

Para ajustar los parámetros de la red, el algoritmo de *back-propagation* computa el gradiente de la *función de error* con respecto a los pesos para minimizar el error. Para lo mismo, se utiliza el método del *descenso por gradiente* (*gradient descent*). Este consiste en encontrar un mínimo local para una función diferenciable. La idea se basa en dar pasos repetidos en la dirección opuesta al gradiente (o gradiente aproximado) de la función en el punto actual, debido a que esta es la dirección del descenso. Por el contrario, dirigirse hacia el gradiente lleva a encontrar un máximo local.

Luego, si utiliza el método *on-line* de *backpropagation*, el algoritmo itera sobre el ejemplo de entrenamiento $\{(x(n), d(n)\}_{n=1}^{N}$ de la siguiente manera:

1. *Inicialización:* Asumiendo que no hay información previa, se elige el peso sináptico y los umbrales de una distribución uniforme cuya media sea cero.

2. *Entrenamiento:* Se presenta una época de ejemplos de entrenamiento. Por cada ejemplo de la red, se computa lo siguiente:

   a) *Paso forward:* Dado el par de entrenamiento $(x(n), d(n))$, con el vector $x(n)$ aplicado a la capa de entrada y el vector $d(n)$ presentado a la capa de salida, se propaga la entrada, capa por capa, a través de la red aplicando la función de activación y se computa la señal de error contra la capa de salida.

   b) *Paso backward:* Se propaga el error de atrás hacia adelante, ajustando los pesos sinápticos y los umbrales de la red en la capa l de acuerdo con la siguiente fórmula:

   $$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \alpha[w_{ji}^{(l)}(n-1)] + \eta\delta_j^{(l)}(n)y_i^{(l-1)}(n) \tag{3.9}$$

   donde η es el *learning-rate*, α es una constante de impulso y se computa el gradiente local de la red con la fórmula:

   $$\delta_j^{(l)}(n) = \begin{cases} e_j^{(L)}(n)\varphi_j'(v_j^{(L)}(n)) & \text{para la neurona } j \text{ de la capa de salida } L \\ \varphi_j'(v_j^l(n)) \sum_k \delta_k^{l+1}(n) w_{kj}^{(l+1)}(n) & \text{para la neurona } j \text{ de la capa oculta } l \end{cases}$$

   Normalmente, el valor de los parámetros η y α son ajustados (usualmente disminuidos) a medida que las iteraciones incrementan.

   c) *Iteración:* Ambos pasos se computan iteradamente presentando nuevas épocas con ejemplos de entrenamiento hasta llegar al *criterio de parada*. En general, no se puede mostrar que el algoritmo de *backpropagation* converja, por lo que se pueden utilizar distintos criterios para finalizar el entrenamiento. Uno de estos puede ser cuando el valor absoluto del cambio en el ECM del error por época es relativamente bajo, lo cual puede generar que el algoritmo termine antes. Otro tipo de corte se puede dar cuando se considera que el desempeño de la *generalización* es adecuada o ha alcanzado su punto máximo.

El desempeño de la *generalización* se calcula sobre ejemplos de entrada-salida de un *conjunto de test*[17], por lo que los datos pueden ser distintos a los de entrenamiento. Cuando la red recuerda los ejemplos de los datos de entrenamiento, se lo llama *memorización*, lo cual puede generar un *sobre-entrenamiento* (*overfitting* u *overtraining*). En estos casos, la red pierde la capacidad de generalizar, ya que genera buenos resultados sobre los datos de entrenamientos, pero falla ante datos nuevos.

Long short-term memory

Uno de los problemas de las RNN es que sufren de memoria a corto plazo (*short-term memory*). Supongamos que queremos predecir la última palabra de la oración *"Yo crecí en España, por lo que hablo español"*. La información cercana a la última palabra sugiere que se va a hacer mención a un idioma, pero para saber a cuál, necesitamos la información que se encuentra al principio de la oración (*España*). *Hochreiter* explica en su trabajo *Long-short term memory* [32] las razones por las cuales una RNN tiene problemas para manejar dependencias a largo plazo. Durante el proceso de *backpropagation*, la RNN padece el *problema de desvanecimiento de gradiente* (*vanishing gradient problem*). Los gradientes eran los valores utilizados por la red para actualizar sus parámetros. El problema surge debido a que el gradiente va disminuyendo a medida que se propaga a través del tiempo, cortando el entrenamiento en etapas tempranas. Debido a que ciertas capas no aprenden, la red olvida lo que ha visto en secuencias largas.

En una RNN tradicional, cada capa combina su estado actual con el anterior, generando un vector que es utilizado en la función de activación para generar una salida. En la figura 3.7a, podemos observar cómo está formada una RNN, siendo x la entrada, o la salida, h el bloque principal de la RNN y v la comunicación entre cada paso. En la figura 3.7b, podemos observar un ejemplo simple de un bloque h de la RNN.

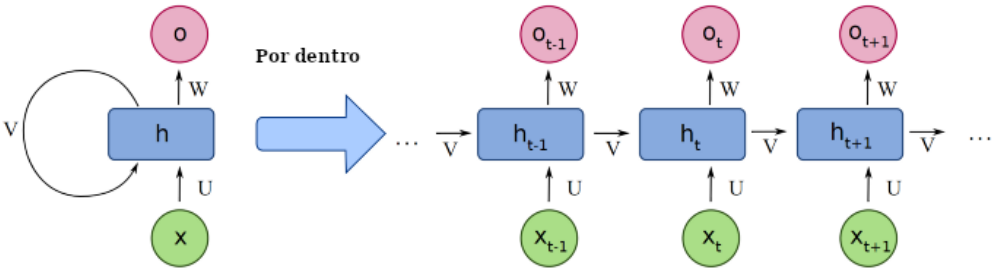
El *long short-term memory* (LSTM) es un tipo de arquitectura de una RNN que sirve para persistir información. Se diferencia de una RNN simple en cómo está formado el bloque h, es decir el interior de una celda (ver ejemplo en figura 3.8).

La clave de esta red es una celda de memoria que puede mantener la información por períodos largos de tiempo (en la figura 3.8, la celda es la línea horizontal en la parte superior que atraviesa el bloque). La red utiliza unos mecanismos llamados *puertas* (*gates*), que controlan la información que entra en la memoria, la que sale y la que se descarta. Las puertas están compuestas por una capa *sigmoidea* y una función de multiplicación puntual, indicando con un valor entre 0 y 1 cuánto de cada componente debe salir (el valor 0 indicaría que nada debería salir y un valor de 1 indicaría que todo debería salir). De esta forma, propaga la información relevante a través de toda la cadena de secuencias.

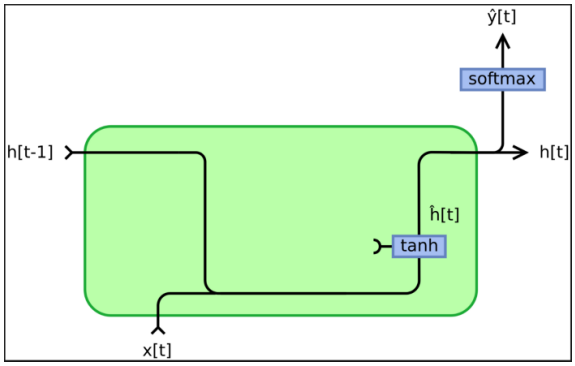
RNN bidireccional

Supongamos que tenemos el ejemplo *"abril es mi mes preferido"* y queremos saber si *abril* hace referencia a un nombre o al mes. Para este caso, no basta con tener solo la información del pasado, ya que si tuviésemos la información del futuro, sabríamos que *abril* hace referencia a un mes. Por esta razón es que entran en juego las *redes neuronales recurrentes bidireccionales*, que procesan la capa de entrada en dos direcciones: una pasada forward de izquierda a derecha para preservar información del pasado al futuro, y otra

[17] Conjunto de datos que es independiente del conjunto de entrenamiento pero que tiene la misma distribución.



(a) Interior de una celda (neurona) en una RNN.



(b) Representación de una celda de una RNN.

Fig. 3.7: Ejemplo de una red neuronal recurrente. Gráficos adaptados de `https://medium.com/deeplearningbrasilia/deep-learning-recurrent-neural-networks-f9482a24d010`.

pasada *backward* de manera inversa (derecha a izquierda) para propagar información de futuro a pasado (ver figura 3.9). De esta manera, cada celda dispone de más información. Luego, las salidas de cada capa son combinadas, por ejemplo concatenadas o sumadas.

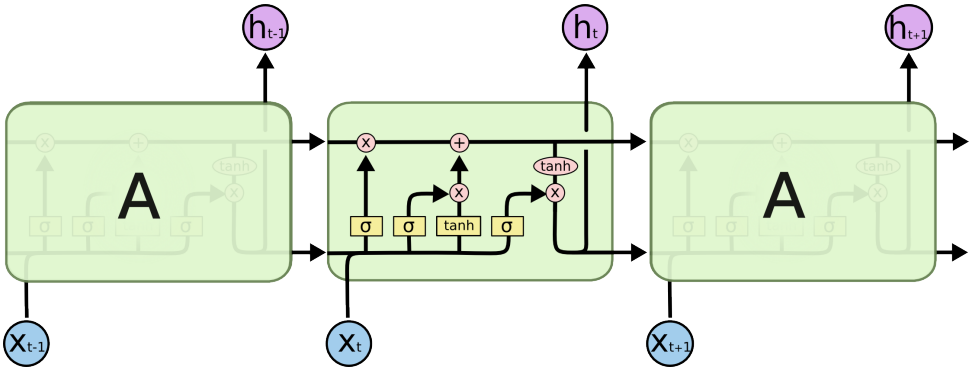


Fig. 3.8: Ejemplo de una celda de una red LSTM. Gráfico tomado de `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`.

LSTM bidireccional

Una arquitectura de red *LSTM bidireccional* (BiLSTM) es una extensión de la arquitectura LSTM utilizando una RNN bidireccional, en la cual se entrenan dos LSTM (una vez procesando la entrada en el sentido original y otra en el sentido opuesto). De esta forma, combinando las dos capas ocultas, se preserva la información tanto del futuro como del pasado. En la figura 3.10 se puede ver un ejemplo de una arquitectura BiLSTM.

3.5. Medidas de calidad

Existen distintas técnicas utilizadas para medir el desempeño de un modelo de aprendizaje automático. Una métrica A puede proveer buenos resultados para un modelo, mientras que otra métrica B puede dar resultados pobres, por lo que se debe evaluar cuál es la métrica adecuada. A continuación, detallamos algunas de estas medidas.

Classification Accuracy

La *classification accuracy* mide, simplemente, la proporción de clasificaciones correctas sobre el total de ejemplos de entrada. Está dada por la fórmula:

$$Accuracy = \frac{\text{Número de predicciones correctas}}{\text{Número total de instancias de ejemplo}} \tag{3.10}$$

Este tipo de medidas funciona bien cuando la cantidad de ejemplos pertenecientes a cada clase es equitativo. Supongamos que el modelo sabe clasificar correctamente los elementos de una clase A, pero no de una clase B. Al recibir un conjunto de entrenamiento con el 98 % perteneciente a la clase A y el resto a la B, tendríamos un modelo con 98 % de accuracy; pero si en el conjunto de entrenamiento poseemos 60 % de ejemplos de la clase A, tendríamos un modelo con 60 % de accuracy. Existen situaciones en las que el costo de clasificar mal a los elementos de la clase B es muy alto, y nuestro modelo nos diría que nuestra precisión es buena.

Matriz de confusión

La *matriz de confusión* es una herramienta que se utiliza para analizar los resultados del algoritmo, y consiste en construir una matriz con los resultados del modelo de clasificación

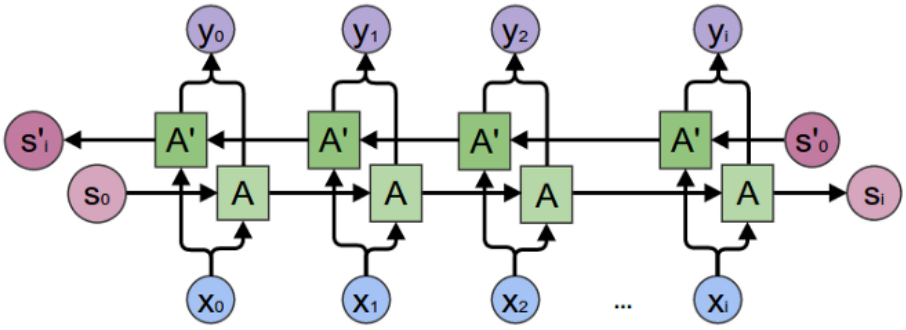


Fig. 3.9: Red neuronal recurrente bidireccional. Gráfico tomado de `https://medium.com/@raghavaggarwal0089/bi-lstm-bc3d68da8bd0`.

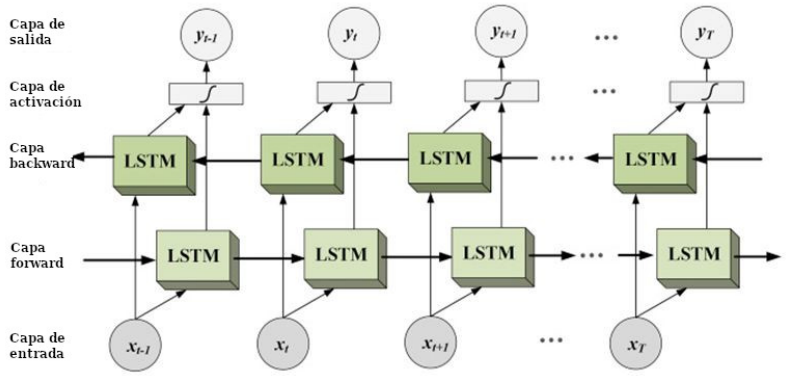


Fig. 3.10: Arquitectura long short-term memory bidireccional. Gráfico adaptado de `https://www.i2tutorials.com/deep-dive-into-bidirectional-lstm/`.

para un conjunto de datos del cual conocemos los valores correctos. En las filas de la matriz se encuentran las instancias predichas de una clase y en las columnas las instancias actuales de las clases. Los resultados se clasifican en cuatro categorías:

- *Verdaderos positivos (VP):* Casos en los que se predice correctamente.
- *Verdaderos negativos (VN):* Casos en los que se predice correctamente que no pertenecen a la clase.
- *Falsos positivos (FP):* Casos en los que se predice erróneamente que pertenece a la clase.
- *Falsos negativos (FN):* Casos en los que se predice erróneamente que no pertenece a la clase.

Supongamos que tenemos un problema de clasificación binaria, luego tendríamos la siguiente matriz de confusión:

	Predicción: Positivo	**Predicción: Negativo**
Real: Positivo	VP	FN
Real: Negativo	FP	VN

Esta matriz forma la base para distintos tipos de métricas. Por ejemplo, podemos calcular la *accuracy*, que mide con qué frecuencia el modelo clasifica correctamente, utilizando la siguiente fórmula:

$$Accuracy = \frac{VP + VN}{TP + FP + TN + FN} \tag{3.11}$$

Si bien no es mandatorio crear una matriz de confusión para obtener estas métricas, es bastante útil para intentar entender dónde se encuentran los casos comunes de error.

F1 score

El *F1 score* es una medida general para evaluar el desempeño de un algoritmo de ML. Se calcula a través de la *precision* (evalúa cuántos de los resultados obtenidos son correctos) y el *recall* (evalúa cuántos elementos fueron correctamente recuperados) sobre cada etiqueta, los cuales ayudan a definir sobre qué aspectos necesita mejorar nuestro algoritmo. Las fórmulas son las siguientes:

- *Precision:* Es la cantidad de resultados correctos positivos dividido por el número de resultados positivos predichos por el clasificador:

$$Precision = \frac{VP}{VP + FP} \tag{3.12}$$

- *Recall:* Es la cantidad de resultados correctos positivos dividido por el número de resultados que deberían haber sido identificados como positivos:

$$Recall = \frac{VP}{VP + FN} \tag{3.13}$$

Una vez que se tienen ambos valores, se puede calcular el desempeño general del algoritmo a través de la media armónica de la *precision* y *recall*. Esta medida se llama *F1-score* y está dada por la fórmula:

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{3.14}$$

Si bien esta medida no ayuda a identificar específicamente en dónde falla el algoritmo, sirve como una medida general.

4. CREACIÓN DE UN CORPUS PARA LA DETECCIÓN DE BROTES DE ENFERMEDADES PREVALENTES EN AMÉRICA LATINA

En este capítulo se describe el proceso para la creación de un corpus anotado, el cual es necesario para entrenar y testear el algoritmo de redes neuronales para la detección de entidades nombradas. Se comienza con una introducción sobre qué es un corpus y su importancia. Se continúa con los trabajos previos sobre el área. Luego, se desarrolla sobre la creación del corpus, seguido del proceso de anotación del mismo. Finalmente, se realiza un análisis sobre el conjunto de datos anotados.

4.1. Introducción

Un corpus (corpora en plural) es un conjunto de textos escritos o de muestras orales (generalmente transcritas) que se utilizan para un propósito específico. Juegan un rol esencial dentro del procesamiento del lenguaje natural para realizar análisis lingüístico estadístico, siendo utilizadas, por ejemplo, para la traducción automática o el reconocimiento del habla. Pueden estar agrupados según su género (como ficción literaria, artículos periodísticos, discursos hablados, blogs, documentos legales, etc.) y pueden estar escritos en un solo idioma o en múltiples. Por ejemplo, para el lenguaje español, existe un corpus que contiene millones de palabras extraídas de artículos de Wikipedia[18] escritos en español.

Un corpus que pasó por un proceso de anotación se denomina *corpus anotado*. Existen distintos tipos de componentes sintácticos y semánticos que pueden ser anotados. Un ejemplo de componente sintáctico es el *PoS-tagging* (ver sección 3.2), que le asigna a cada palabra su categoría gramatical, lo cual es utilizado para tareas como el *named entity recognition*. También puede ser importante anotar en los textos su valor semántico, es decir, qué significa una palabra dentro de una oración. De esta forma, se pueden asignar etiquetas ontológicas[19] (por ejemplo, enfermedades, fechas o ubicaciones geográficas) que nos permiten observar cómo los objetos interactúan unos con otros (a través de relaciones).

El proceso de creación de un corpus anotado es una tarea costosa porque, además de la recolección de datos, es necesario definir un esquema y un criterio de anotación, y llevar a cabo la anotación. Estos son definidos y revisados a través de un proceso iterativo. Debido a la complejidad del lenguaje natural y su ambigüedad, pese a existir un esquema y criterios de anotación, van a haber diferencias entre las anotaciones realizadas entre personas distintas e, incluso, entre las de una misma persona. Para evaluar la consistencia de las anotaciones existen distintas medidas, una de ellas llamada *inter-annotator agreement*. Una vez obtenido el corpus anotado final, se puede proceder a entrenar y testear algoritmos de aprendizaje automático supervisado para detectar entidades nombradas y extraer relaciones.

Algunas de las tareas descriptas por Ide y Pustejovsky [36] como necesarias para el

[18] La Wikipedia es una enciclopedia gratis online multilingüe y colaborativa, creada y mantenida por una comunidad de editores voluntarios que utilizan un sistema de edición que provee el sitio.

[19] En la lingüística computacional, la ontología se utiliza para crear categorías que agrupan conceptos y objetos.

proceso de anotación son: crear los archivos en un formato estándar, definir los criterios de anotación, definir las habilidades y conocimiento previo de los anotadores, entrenar a los anotadores para que el valor de los resultados del IAA sea aceptable, planear el orden de anotación y sus tareas, distribuir los documentos entre los anotadores, monitorear el progreso de los anotadores, recolectar las anotaciones de cada anotador, realizar un seguimiento de los valores del inter-annotator agreement para asegurar la calidad de las anotaciones, agendar reuniones, realizar un seguimiento de las horas de trabajo utilizadas y el presupuesto del proyecto.

Si bien existen corpus sobre el dominio de brotes de enfermedades endémicas (por ejemplo, para el idioma inglés encontramos el de Conway y col. [22] y el *PADI-web* [51]), según nuestro saber y entender no se encuentran disponibles para ese dominio en idioma español. Para poder desarrollar la tarea de alertar e informar sobre posibles brotes de enfermedades endémicas, debemos contar con un corpus sobre el cual desarrollaremos y aplicaremos nuestras herramientas. Para generar uno, necesitamos artículos de medios confiables cuyo tópico abarque las enfermedades de interés. Este proceso involucra: seleccionar la fuente confiable desde la cual se obtendrá la información deseada y luego realizar cierto preprocesamiento sobre la misma. Luego de esto, se podrá proceder a la anotación del corpus para la detección de entidades (enfermedades, causa de las mismas, ubicación geográfica y fecha, entre otros) y relaciones (que establecen vínculos entre las entidades). Una vez que se tiene el corpus anotado, se pueden entrenar y testear algoritmos supervisados de aprendizaje automático.

En este capítulo presentamos el trabajo realizado para la creación del corpus anotado sobre el dominio de brotes de enfermedades que afectan a países de América Latina. Iniciamos describiendo los textos elegidos en la sección 4.3.1. En la sección 4.3.2, presentamos los *data statements*, que son declaraciones sobre el conjunto de datos seleccionado. En la sección 4.3.3, describimos la limpieza de datos que se tuvo que realizar sobre los artículos descargados. Esto último incluye una eliminación de metadatos (4.3.3.1) y una normalización de los datos (sección 4.3.3.2). Luego, procedemos a detallar la anotación del corpus en la sección 4.4. Presentamos en la sección 4.4.1 el esquema de anotación y en la sección 4.4.2 los criterios de anotación. Luego, en la sección 4.4.3, presentamos cómo fue llevado a cabo el proceso de anotación manual. Posteriormente, en la sección 4.4.4, se habla sobre el *inter-annotator agreement*, lo cual es una medida para evaluar la consistencia entre las anotaciones. Para el mismo, se utiliza el coeficiente κ de Cohen, que será presentado en la sección 4.4.4.1. También, se introduce el F1-score en la sección 4.4.4.2, cuyos resultados serán presentados pero no tomados en cuenta. Finalmente, se realiza un análisis del conjunto de datos anotados en la sección 4.5.

4.2. Trabajos previos

El uso de corpus para el análisis de lenguaje se remonta a varias décadas atrás. En 1897, Käding investigaba la frecuencia de secuencias de letras en corpus de millones de palabras del idioma alemán [40]. En 1951, Roberto Busa publicó la primera concordancia generada a máquina [69]. Luego, en 1964, *Juilland y Chang-Rodriguez* presentaron un trabajo [38] en el cual realizaron estudios estadísticos sobre un corpus de 500.000 palabras en español. En 1965, se presentó el primer corpus electrónico por *Kucera y Francis* [27] (conocido como *Brown Corpus*), que poseía más de un millón de palabras en inglés americano recolectadas

de 500 ejemplos (como libros y reportajes, entre otros) de 15 géneros distintos (como reseñas de libros, reportajes políticos y novelas de ficción, entre otros).

Para el idioma español, contamos con distintos corpora digitales, como el *Spanish Billion Word Corpus* [15], el cual consta de una compilación de distintos corpus (el *AnCora* [63] y parte del *Europarl* [43], entre otros), o los *Wikidumps* basados en datos de *Wikipedia* (como artículos). Sin embargo, ninguno de estos es específico del dominio de interés de este trabajo.

4.3. Creación del corpus

En esta sección, presentamos el conjunto de textos seleccionados para la creación del corpus. Luego, introducimos los *data statements* sobre nuestros datos. Finalmente, detallamos el proceso de normalización y limpieza sobre los textos para obtener el corpus final.

4.3.1. Selección del conjunto de datos a anotar

Debido a la falta de corpora para el dominio e idioma deseado, tuvimos que construir nuestro propio corpus. Utilizamos como fuente ProMED-mail (*Program for Monitoring Emerging Diseases*) que, como fue mencionado anteriormente, es un sistema de reportes dedicado a difundir información sobre brotes de enfermedades infecciosas. Los artículos allí publicados han sido editados basándose en notas periodísticas de diferentes medios por un equipo interdisciplinario de profesionales.

Se descargaron un total de 1377 artículos utilizando un *web scraper*[20]. Se filtraron aquellos artículos que satisfacían las siguientes dos condiciones: (1) que estuviesen escritos en español y (2) que mencionasen algunas de las enfermedades de interés (Chagas, dengue, Guillain-barré, sarampión, zika y microcefalia).

En la figura 4.1 se puede ver un ejemplo de un artículo obtenido de ProMED-mail. Como se puede observar en el mismo, la mayoría de los artículos cuentan con un título, una fecha, metadata, el cuerpo del artículo y, ocasionalmente, más metadata. Además, un dato a tener en cuenta que será utilizado posteriormente, es que en gran parte de los títulos, se hace mención a una ubicación geográfica en la cual se encuentra la enfermedad.

4.3.2. *Data statements*

Las declaraciones sobre los datos (*data statements*) son una práctica profesional propuesta por Bender y Friedman [8] para tecnologías del procesamiento del lenguaje natural. Ellas la definen como una caracterización del conjunto de datos que va a permitir ver cómo podrían impactar en la generalización y entender qué sesgos[21] podrían encontrarse en el sistema en función a los sesgos que ya se encuentran en los datos.

Las autoras sostienen que una forma para trabajar con conjuntos de datos que ya contienen sesgos, para mitigar los problemas que pudiesen ocasionar, es proveer esta caracterización de los datos, tanto para el conjunto de datos de entrenamiento como de test.

[20] Técnica utilizada mediante programas de software para extraer información de sitios web

[21] Con sesgos nos referimos a casos en los que sistemas informáticos "sistemática e injustamente discriminan a determinadas personas o grupos de individuos a favor de otros" (Friedman and Nissenbaum, 1996, página 332) [28].

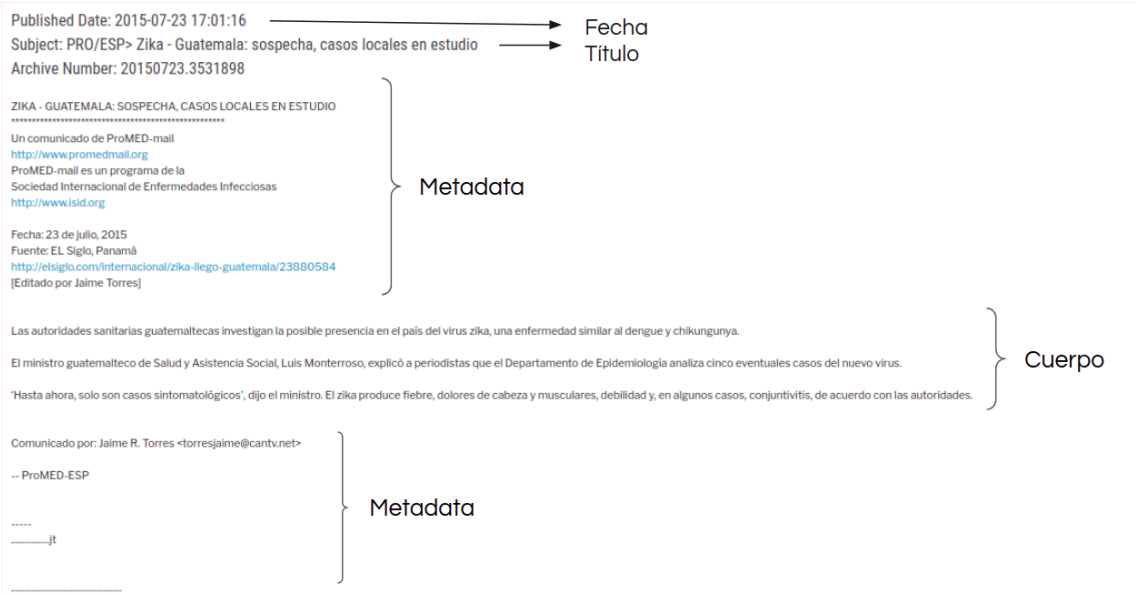


Fig. 4.1: Ejemplo de artículo de ProMED-mail, junto a las distintas secciones de información que se encuentran.

De esta forma, se deja claro cuáles son los grupos que se encuentran o no representados, y cómo los afectarán.

Hemos seleccionado textos provenientes del sitio web ProMED-mail que se encuentran escritos en español y hacen mención a alguna de las siguientes enfermedades: Chagas, sarampión, dengue, Guillain Barré, hantavirus, microcefalia o zika, que fueron escritos entre el 04/01/2001 y 18/08/2020. De estos, solo nos quedamos con aquellos que hacen mención a países latinoamericanos. De esta forma, obtenemos artículos de calidad sobre las enfermedades mencionadas y su prevalencia en América Latina.

El lenguaje utilizado en estos textos es el usual en textos periodísticos, aunque estos textos son más cortos que los usuales. Si bien cada país tiene sus particularidades en cuanto al uso del idioma español (en lo que refiere a textos, diferente uso de palabras), en los artículos se utiliza un español estándar (no identificado con alguno particular de algún país).

Tanto los que desarrollaron los criterios de anotación como los anotadores en sí, hablan distintas variantes del idioma español. Sin embargo, esto no generó un obstáculo para el entendimiento adecuado de los criterios de anotación ni de los artículos. No contamos con información demográfica de los editores de los artículos de ProMED-mail.

4.3.3. Limpieza de datos

Los artículos descargados desde ProMED-mail contenían información que no es de interés y que podían generar confusión en los anotadores. En la figura 4.1 se puede observar un ejemplo de un artículo que contiene metadata no relevante. Por esta razón, queríamos proveer a los anotadores la información justa y necesaria para realizar la tarea. Además, ya que los algoritmos recibían los mismos artículos a anotar (ya que debemos compararlos contra los anotados manualmente), teníamos que asegurarnos que los datos se encontrasen en un formato para que puedan desarrollar la tarea correctamente, adecuándonos a lo

requerido por las herramientas utilizadas para el procesamiento (por ejemplo, debemos unificar el caracter separador de fechas, para que todas contengan el caracter '/' en vez de '-').

En esta sección se habla de dos tipos de procesamientos que fueron realizados sobre todos los artículos: la eliminación de metadatos no relevantes y una normalización sobre los datos necesaria para el procesamiento de los algoritmos.

4.3.3.1. Eliminación de metadatos

Los artículos descargados poseen cierta metadata que no solo no aporta información valiosa para la extracción de información, sino que genera ruido, por lo que se decidió eliminarla a través de expresiones regulares (ver sección 3.2). El objetivo final era obtener un artículo que solo contenga el *título*, la *fecha* y el *cuerpo*. Un ejemplo de formato de archivo con su correspondiente metadata se muestra a continuación:

Título
Fecha
****
Un comunicado de ProMED-mail
http://www.promedmail.org
... (7 líneas de metadata, en este caso: fecha, fuente, etc.)
[Editado por *$nombre*]
Cuerpo del artículo
Comunicado por: *$nombre <$dirección-mail>*
– ProMED-ESP
.....jt

En el encabezado superior se encuentran distintos datos como el título, la fecha de la publicación y, finalmente, la fuente. Esta última, en la mayoría de los casos, se encontraba señalizada con el texto "Fuente:" o "Fuente". Inicialmente, se intentó realizar una expresión regular que encontrase dicho patrón y eliminase toda la información correspondiente a la fuente (que puede ser una secuencia de texto, una URL y el texto "[Editado por...]" o "[Traducido por...]", que pueden encontrarse en la misma línea o no). Un ejemplo de esta expresión regular fue:

```
.*Fuente.*([Ee]ditado  por)(.*?)(\])|
(.*Fuente.*(\[editado\])(.*<a.*?<\/a>(.|\s)*?(<br>)))
```

La misma tuvo que ser descartada porque se encontraron textos que directamente no contenían ninguna palabra señalando la fuente de origen, sino que directamente la información en sí. Luego, se realizó un análisis más profundo de cómo se encuentran dichos encabezados con metadata, y finalmente se decidió primero eliminar todo lo precedente al texto "Fuente" y luego realizar una expresión regular que solamente buscase aquellos textos con las palabras "traducido por" o "editado por" y eliminase todo lo precedente con la siguiente expresión regular:

```
(\[|\().*(([Ee]ditado)|([T|t]raducido)).*?(\]|\))
```

Dado que existía una minoría de textos que no contenían dicha información, se comprueba luego si se eliminó texto posterior a la palabra "Fuente" y, en caso contrario, se elimina todo lo que se encuentre entre ella y el siguiente salto de línea (delimitado por la etiqueta "
" de html). Para eliminar la metadata que se encontraba al final del artículo, se utilizó la siguiente expresión regular:

```
((Comunicado por)(.|\s)*)|(((-)+|(#+)\*)(\s*)?(Pro[mM][eE][dD])(\s|.)*)|
(Referencias\:(\s|.)*)|(Referencia\:(\s|.)*)
```

Este se encarga de eliminar el texto que le procede a la palabra "Comunicado por" o "Referencias" o "Promed" (y sus variantes).

También se eliminó todo caracter no ASCII (por ejemplo "<U+0093>") y las secuencias de caracteres repetidos como "- - - - -". Dado que los textos que se procesan se encuentran en formato html, todos los saltos de línea (identificados por la etiqueta
) fueron transformados efectivamente en saltos de línea, eliminando repeticiones de los mismos, y se eliminaron hipervínculos (identificados dentro de las etiquetas <a href="link"><a>).

Otro tipo de metadata no relevante encontrada se encontraban señalizadas entre dos corchetes con el texto "Ver también" y posibles variaciones, se decidió descartar dicha información con la expresión regular:

```
\[[Vv]er tambi[eé]n)((.+?)\]
```

4.3.3.2. Normalización de datos

Se tuvo que realizar posteriormente una normalización de los datos para que todos los artículos tuviesen un determinado formato. Esto era requerido por las herramientas de procesamiento utilizadas para poder extraer entidades nombradas y para realizar otro tipo de tareas como segmentación de oraciones. A lo largo de este trabajo, cuando hablemos de los algoritmos y herramientas que utilizamos para anotar automáticamente los textos, se podrá observar la necesidad de realizar estos preprocesamientos. Las normalizaciones aplicadas fueron:

- **Unificación de separador de fechas:** En algunos artículos, las fechas están escritas utilizando un guión como separador (por ejemplo, 10-03-2018) y, en otras, con una barra (por ejemplo, 10/03/2018). En este caso, se procedió a unificar las fechas para que todas utilicen la barra como separador utilizando la siguiente expresión regular para detectar fechas con guiones y reemplazarlos por barras:

```
(0?[1-9]|[12][0-9]|3[01])([-])      // día
(0?[1-9]|1[012])([-])               // mes
((19|20)?\d\d)                      // año
```

- **Unificación de formato de fechas:** En algunos artículos, el año se encontraba separado por una coma en vez de la palabra *'de'* (por ejemplo, *"12 de Marzo, 2012"*), lo que generaba que nuestra librería de procesamiento de lenguaje no la detectara como una sola fecha, sino que detectaba dos fechas: *12 de Marzo* (Date) y *2012* (Date). Por esto, se decidió utilizar una expresión regular para reemplazar las comas por la palabra *'de'*. La expresión regular consistía en identificar:

```
(${mes}), (${año}) // (siendo mes una expresión regular
// conteniendo la unión de todos los meses del año,
// y año la expresión regular (19|20)?\d\d)
```

y reemplazarlo por: *"$1 de $2"* (siendo $\$i$ la secuencia de texto identificada por el grupo de la expresión regular i).

- **Punto final en título y fecha:** Se agregó siempre al final de la fecha y del título un punto final, seguido de un salto de línea, para que nuestra librería de procesamiento pudiese identificar cada una como oraciones distintas y poder procesarlas por separado.

- **Saltos de línea:** Se agregaron dos saltos de líneas posteriores al título/fecha para que sea más legible a la hora de realizar las anotaciones.

- **Unificación de separador de decimales:** Al momento de declarar números, en algunos casos se utilizaba como símbolo para separar milésimas el punto y, en otros, la coma. Para esto, se utilizó una expresión regular para reemplazar las comas con puntos, teniendo en cuenta que hay que identificar cuándo corresponde o no. Nuestro interés es únicamente identificar aquellos valores que representen cantidad de población afectada, por esto nuestra expresión regular solo va a considerar como máximo el número 9.999.999.999 (considerando que la población mundial actual es de alrededor 7,800,000,000 personas). La expresión regular utilizada para detectar dichas expresiones con comas fue:

  ```
  ^((\d{1}\,)?(\d{1,3}\,)?\d{1,3}\,\d{3})$
  ```

  Luego de detectado, se procede a reemplazar las comas por puntos.

- **Normalización de nombre de países:** En algunos artículos, se hace mención a los países utilizando su acrónimo (por ejemplo, *EEUU* en vez de *Estados Unidos*). En este caso se decidió tener un listado de acrónimos posibles y reemplazarlos siempre por su nombre expandido, enfocándonos principalmente en países latinoamericanos. Estas expresiones regulares fueron (junto con su reemplazo):

  - (P|p). [Rr][Ii][Cc][Oo] → *Puerto Rico*
  - (C|c). [Rr][Ii][Cc][Aa] → *Costa Rica*
  - (R|Rep|r|rep|REP). [Dd][Oo][Mm][Ii][Nn][Ii][Cc][Aa][Nn][Aa] → *República Dominicana*

- **Eliminación de datos extra en fechas:** A través del *web scrapper* se obtenían archivos en formato JSON[22] que contienen, además del texto obtenido de la página web, un campo con el título y otro con la fecha. Respecto al último, en algunos casos, se poseía además parte de la nota periodística. Por ejemplo:

[22] JSON es un tipo de formato de archivos que consiste en objetos clave-valor y arreglos.

```
10 de noviembre, 2006<br>De: Jaime R. Torres &lt
```

En estos casos, se eliminó todo lo procedente al año, utilizando la siguiente expresión regular para identificarlo:

```
.*?([0-9]{4})
```

4.4. Proceso de anotación del corpus

En esta sección nos enfocamos en describir el proceso, el esquema y los criterios de anotación que se utilizaron para la creación de un corpus anotado para realizar la extracción de entidades nombradas y de relaciones sobre los artículos, cuya recolección fue descripta en la sección 4.3. Comenzamos explicando la importancia y la necesidad de realizar un proceso de anotación. Luego, continuamos con la descripción del esquema, criterio y proceso de anotación seleccionado. Posteriormente, se realiza una descripción de los datos anotados y se evalúa la consistencia entre las anotaciones realizadas utilizando una medida de calidad llamada *inter-annotator agreement*. Finalmente, se presentan los resultados de la anotación junto con un análisis de los mismos.

4.4.1. Esquema de anotación

Una vez recolectado el corpus, se debe modelar el problema. Este va a describir la tarea a través de los términos que son relevantes para el proyecto, definiendo lo que se busca capturar del conjunto de datos. Estas clasificaciones pueden estar pensadas en términos de etiquetas que se aplican al texto (las entidades nombradas en este trabajo) y las relaciones que pueden existir entre ellas. El esquema de anotación es la especificación del modelo, dado en forma de etiquetas y relaciones que queremos extraer del corpus.

Dado que el objetivo del proyecto es detectar enfermedades que afectan a países latinoamericanos, plantear una serie de interrogantes de las cuales queremos obtener respuestas nos puede ayudar a plantear las entidades y relaciones que queremos extraer de cada artículo. Alguna de estas interrogantes son:

- ¿A qué tipo de enfermedad se hace mención?
- ¿Cuántos casos de la enfermedad se reportan?
- ¿A qué tipo de personas afecta?
- ¿En dónde se encuentra localizada?
- ¿Cuáles pudieron ser las posibles causas?
- ¿Cuándo ocurrió el brote de la enfermedad?

La definición de las distintas entidades y relaciones que nos ayudarán a responder las interrogantes se fue estableciendo a través de un proceso iterativo. Inicialmente, se definió un conjunto de estas. Luego de cada tanda de anotación, se prosiguió a revisar y redefinir los mismos (modificando o eliminando existentes, y agregando nuevos). En las próximas secciones presentamos el esquema final de las anotaciones.

4.4.1.1. Entidades

Las entidades nombradas son elementos del mundo real y están formadas por una palabra o una secuencia de ellas. A continuación serán enumeradas, junto con la abreviación con las que serán referenciadas posteriormente en este trabajo:

disease (DS): entidad que corresponde a un hallazgo o diagnóstico, por ejemplo, *"zika"* o *"Chagas"*,

date (DT): mención a una fecha, por ejemplo, *"28 de febrero, 2019"* o *"30 de abril"*,

location (LOC): ubicación geográfica, por ejemplo, *"MÉXICO: (JAL)"*, *"Capiatá"* o *"Montevideo"*,

number of cases (NoC): mención a la cantidad de casos de una enfermedad,

origin (OR): entidades que corresponden a la causa de la enfermedad, por ejemplo, en *"La transmisión de la enfermedad ocurrió a través del consumo de Açai contaminado"*, *"Açai contaminado"* sería anotado como OR.

transmission form (TF): refiere a la forma de transmisión de la enfermedad. En el ejemplo de la entidad anterior, *"consumo"* sería anotado como TF. Otros ejemplos podrían ser: en *"El virus se transmite por picadura de mosquito"*, *"picadura"* sería la TF y *"mosquito"* sería el OR.

host (Hst): mención a la persona o animal que contrajo la enfermedad, por ejemplo, *"bebé*, *"embarazada"*, *"anciano"*, *"inmigrante"*.

Existe otro tipo de conceptos, llamados *modificadores*, que si bien no son entidades, fueron considerados como tales para poder ser anotados con la herramienta:

negation (NT): términos que indican negación, por ejemplo, *"no"*, *"ausencia"* o *"negativo"*,

uncertainty terms (UT): términos que indican faltas de certeza, por ejemplo, *"casos sospechosos"*, *"podrían causar"* o *"bajo observación"*,

past terms (PT): términos que hacen mención a un hecho que ocurrió en el pasado, por ejemplo, *"en el pasado"*, *"en el 2014"* (siendo una nota con fecha actual posterior),

conditional (COND): términos que corresponden a un hecho que podría ocurrir en el futuro, por ejemplo, en *"si el dengue persiste, ciertas acciones deberían ser tomadas para (...)"*, *"si"* es un COND.

4.4.1.2. Relaciones binarias

Las relaciones binarias establecen una relación entre dos entidades nombradas. Estas son:

disease occurs in (DsOccIn): relación entre una enfermedad (DS) y una ubicación geográfica (LOC), que establece la locación en el que se observó la enfermedad, por ejemplo, en *"once casos fueron observados en el estado de Mérida"*, se anotaría: *"Chagas"* (DS), *"Mérida"* (LOC) y se anotaría una relación *DsOccIn* entre ambas entidades,

origin occurs in (OrOccIn): relación entre el origen de una enfermedad y una ubicación geográfica, que indicaría el lugar en el cual sucedió el origen de ella, por ejemplo, en *"se ha encontrado presencia del mosquito transmisor de la enfermedad en el barrio de Belén"*, se anotaría: *"mosquito"* (OR), *"barrio de Belén"* (LOC), y habría una relación *OrOccIn* entre ambas entidades,

cases of disease (NoCDs): relación entre cantidad de casos y una enfermedad, que indicaría la cantidad de casos que se encontraron de la enfermedad, por ejemplo, en *"un*

total de 391 casos de Zika fueron detectados", se anotaría: *"391"* (NOC), *"Zika"* (DS) y existiría una relación *NoCDs* entre ambas entidades,

cases in location (NoCLoc): relación entre cantidad de casos y una ubicación, que establecería la cantidad de casos que se encontraron en dicha locación, por ejemplo, en *"ya hay 100 pacientes con síndrome de Guillain-Barré en Perú"*, se anotaría: *"100"* (NOC), *"Perú"* (LOC) y existiría una relación *NoCLoc* entre ambas entidades,

cases of host (NoCHt): relación entre cantidad de casos y portador, que establecería la cantidad de casos que afectan a dichos portadores de la enfermedad, por ejemplo, en *"(...) un grupo de 32 recién nacidos con microcefalia en Brasil"*, se anotaría: *"32"* (NOC), *"recién nacidos"* (Hst) y existiría una relación *NoCHt* entre ambas entidades,

date of disease (DtDs): relación entre una fecha y una enfermedad, que indicaría cuándo ocurrió la misma, por ejemplo, en *"en lo que va del 2013, 4 personas murieron (...) de Chagas"*, se anotaría: *"2013"* (DT), *"Chagas"* (DS) y existría una relación *DtDs* entre ambas entidades,

cause of disease (OrDs): relación entre origen y una enfermedad, que indicaría cuál fue el origen de la misma, por ejemplo, en *"el zika en mujeres embarazada es la causa del síndrome de Guillain Barré en sus bebés"*, se anotaría: *"zika"* (OR) (DS), *"Guillain Barré"* (DS) y existiría una relación *OrDs* entre Zika y Guillain Barré,

negates *disease, location, number of cases or cause* (NegDs, NegLoc, NegNoC, NegOr, NegTf): relación entre un término de negación y alguno de las entidades: enfermedad, ubicación, cantidad de casos, causa o forma de transmisión. por ejemplo, en *"10 casos no presentaron microcefalia"*, se anotaría: *"microcefalia"* (DS), *"no"* (NT) y existiría una relación *NegDs* entre ambas entidades,

speculates disease, origin or transmission form (UcDs, UcOr, UcTf, UcNoC): relación entre una especulación y algunos de las entidades: enfermedad, causa/origen, cantidad de casos o forma de transmisión, por ejemplo, en *"se sospecha que la forma de transmisión fue a través de la ingesta (...)"*, se anotaría: *"sospecha"* (UT), *"ingesta"* (TF) y existiría una relación *UcTf* entre ambas entidades,

occurs to (OccTo): relación entre una enfermedad y un portador, indicando que dicho portador sufrió dicha enfermedad, por ejemplo, en *"la prevalencia de Chagas en donantes (donantes de sangre) ha disminuido"*, se anotaría: *"Chagas"* (DS), *"donantes"* (Hst) y existiría una relación de tipo *OccTo* entre ambas entidades,

transmission cause (TfOr): relación entre una causa y una forma de transmisión, que indicaría que ambas se encuentran vinculadas, por ejemplo, *"hubo una transmisión oral a través de comida contaminada con las heces del chipo"*, se anotaría: *"oral"* (TF), *"comida contaminadas con las heces del chipo"* (OR) y existiría una relación de tipo *TfOr* entre ambas entidades,

transmission of disease (TfDs): relación entre una forma de transmisión y una enfermedad, que indicaría cuál fue la forma en la que se transmitió la enfermedad, por ejemplo, en *"la enfermedad se contrajo a través de la ingesta de açaí"*, se anotaría: *"enfermedad"* (Ds), *"ingesta"* (TF) y existiría una relación de tipo *TfDs* entre ambas entidades,

transmission in location (TfOccsIn): relación entre una forma de transmisión y una ubicación geográfica, indicando el lugar en el cuál se produjo la transmisión, por ejemplo, en *"la ingesta se produjo en Brasil"*, se anotaría: *"ingesta"* (Tf), *"Brasil"* (LOC) y existiría una relación de tipo *TfOccsIn* entre ambas entidades,

temporal conditional or past (ConDs, TempDs, TempNoC): relación entre un modificador condicional o de tiempo pasado con una enfermedad, o entre el tiempo

pasado y cantidad de casos, por ejemplo, en *"la semana pasada, la aparición de casos de la enfermedad de Chagas han sido reportados (...)"*, se anotaría: *"semana pasada"* (PT), *"Chagas"* (DS) y una relación de tipo *TempDs* existiría entre ambas.

4.4.1.3. Relaciones ternarias

También existen las relaciones que afectan a tres entidades y se llaman relaciones ternarias. Estas son:

negates transmission cause (NegTfOr): relación entre un término de negación, una forma de transmisión y un origen, indicando que niega que la forma de transmisión esté relacionada con el origen, por ejemplo, en *"ninguno ha declarado haber sido picado por el chipo"*, se anotaría: *"ninguno"* (NT), *"picado"* (TF) y *"chipo"* (OR), y existiría una relación ternaria de tipo NegTfOr entre el trío de entidades.

speculates cause of disease (UcOrDs): relación entre un término de especulación con la causa de una enfermedad, indicando que se podría especular que dicha causa fue la causante de la enfermedad, por ejemplo, en *"un número importante de cepas del virus circulan en el país y todas ellas pueden causar síndrome pulmonar por hantavirus"*, se anotaría: *"cepa del virus"* (OR), *"pueden causar"* (UT) y *"síndrome pulmonar por hantavirus"* (DS), y existiría una relación ternaria UcOrDs entre el trío,

speculates number of cases of disease (UcNoCDs): relación entre un término de especulación, una enfermedad y una cantidad de casos, indicando que el número etiquetado podría ser la cantidad de casos de la enfermedad, por ejemplo, en *"283 casos de personas sospechosas de tener zika (...)"*, se anotaría: *"283"* (NoC), *"zika"* (DS) y *"sospechosas"* (UT), y existiría una relación ternaria UcNoCDs entre el trío.

speculates transmission form of disease (UcTfDs): relación entre un término de especulación, una enfermedad y una forma de transmisión, indicando que la enfermedad pudo haber tenido dicha forma de transmisión. Por ejemplo, en *"se trataba de un posible brote de Enfermedad de Chagas de transmisión oral"*, se anotaría *'posible'* (UT), *'Chagas'* (DS) y *"transmisión oral"* (TF), y habría una relación ternaria de tipo *UcTfDs* entre el trío.

4.4.2. Criterios de anotación

Los criterios de anotación son una guía de instrucciones que definen cómo los anotadores deben aplicar el esquema de anotación sobre los datos para la tarea en particular que se busca atacar. Una analogía propuesta por *Pustejovsky y Stubbs* [50] para diferenciar el esquema de anotación de los criterios es la diferencia entre tener una lista de ingredientes (el esquema) y cómo se utilizan los ingredientes (los criterios). Un mismo esquema puede ser utilizado para distintos tipos de tareas (pudiendo sufrir algunas modificaciones), pero lo que normalmente debe modificarse son los criterios de anotación para que se adapten al problema que se busca resolver (por ejemplo, los mismos ingredientes que se utilizan para hacer pan, pueden ser utilizados para hacer una torta, pero la receta es distinta).

Los criterios de anotación fueron definidos de una manera iterativa, siendo revisados y redefinidos luego de cada proceso de anotación, hasta obtener el criterio que mejores resultados proveía (tanto a nivel de qué preguntas queremos responder como a la consistencia entre anotadores a través del *inter-annotator agreement*). El criterio final fue el siguiente:

- No se anotan relaciones entre entidad que se encuentran en distintas oraciones.

- En el caso en el que hubiese una entidad incluida en otra más grande, la que hay que anotar es la más grande. Por ejemplo, en *'mosquito aedes aegipty'*, se debería anotar la entidad [mosquito aedes aegipty], y no las entidades [mosquito] y [aedes aegipty].

- Los términos correspondientes a entidades nombradas, pero que tienen errores ortográficos, también se anotan.

- Sólo anotamos archivos en cuyo título se menciona a un país latinoamericano. Si el título habla sobre un país fuera de Latinoamérica, no se anota el artículo.

- Para la entidad *Number of cases*, se anotan tanto referencias a casos como a muertes. Por ejemplo, en la oración *"Se registraron 4 muertes por hantavirus*, el número *cuatro* debería ser anotado como entidad de tipo *Number of cases*.

- Los *Hosts* solo son anotados si tienen alguna característica particular (por ejemplo, *bebé*, *embarazada*, *anciano*). Si una persona sin ninguna característica particular contrae una enfermedad, no deberá ser anotada como *Host*. Por ejemplo, en la oración *"La mayor incidencia de muertes corresponde a hombres."*, el término *'hombres'* no sería anotado como una entidad de tipo *Host*.

- Las siguientes entidades no se anotan como entidad si no existe otra entidad en la misma oración con la cual se pueda relacionar (es decir, si la entidad está aislada): **negation**, **uncertainty**, **conditional**, **past** y **date**. Esto se debe a que dichas entidades por sí solas no aportan información suficiente. Por ejemplo, en la siguiente oración: *"Funcionarios del territorio temen que se agrave la situación sanitaria en la urbe a partir de las venideras concentraciones públicas asociadas al acto por el 26 de Julio."*, la fecha *"26 de Julio"* no debe ser anotada, ya que no hay otra entidad que refiera información relevante sobre la fecha.

4.4.3. Anotaciones manuales

El proceso de anotación fue llevado a cabo por hablantes nativos de español peruano (variante andina y limeña) y español argentino (variante rioplatense). Como anotadores se contó con un lingüista, cinco estudiantes de licenciatura en ciencias de la computación, y dos doctores en ciencias de la computación que, a la vez, son investigadores en el área de procesamiento del lenguaje natural, con experiencia en realizar anotaciones en diversas áreas (que serán llamados, de aquí en adelante, los *expertos*). La herramienta utilizada para llevar a cabo la anotación fue *brat rapid annotation tool* [23] [62].

El proceso de anotación se realizó utilizando el modelo conocido como *MAMA* (*Model-Annotate-Model-Annotate*) propuesto por *Ide, Nancy y Pustejovsky* [36]. Este es un proceso cíclico que consiste en definir un esquema y criterios de anotación, luego realizar una tanda de anotaciones, evaluar la consistencia entre anotaciones, revisar el modelo, e iterar sobre esto hasta que se obtenga un modelo deseado (ver figura 4.2).

Los expertos generaron un esquema y los criterios de anotación, y se desarrollaron reuniones para resolver dudas. Luego de la primera iteración de anotación, se resolvieron dudas y diferencias en los criterios, y se actualizaron los respectivos documentos. Se realizaron dos iteraciones más de anotación y revisión, a partir de la cual se definieron

[23] Disponible en `https://brat.nlplab.org/` [Accedido en Junio de 2021].

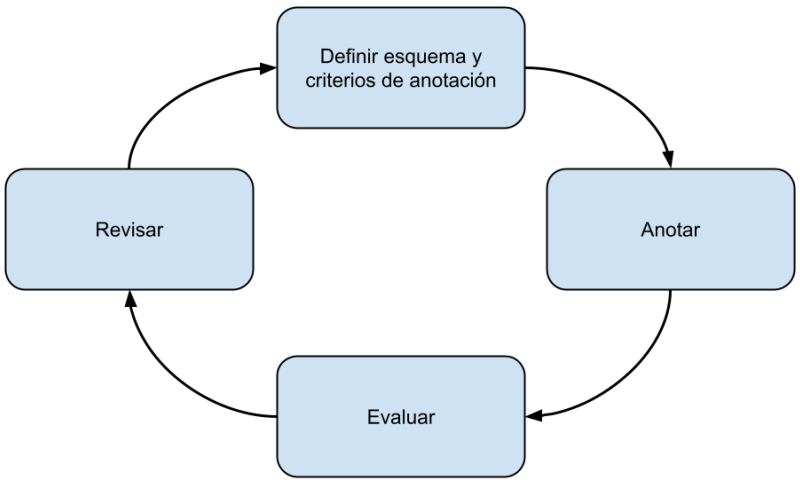


Fig. 4.2: Ciclo de anotación. Gráfico adaptado de Ide y Pustejovsky [36].

los esquemas y criterios finales. Se prosiguió a que cada anotador realizase las correcciones correspondientes sobre cada archivo que anotó. Luego de cada una de las primeras tres iteraciones, el *inter-annotator agreement* fue calculado entre cada par de anotadores *'no-expertos'* contra los expertos para medir la consistencia entre las anotaciones (los resultados pueden encontrarse en la sección 4.4.4.1.2). Posteriormente, se realizaron cuatro tandas más de anotación, sin archivos en común entre los *expertos* y los *no-expertos*.

Los artículos anotados en cada una de las tandas de anotaciones fueron recolectados posteriormente para ser utilizados en los algoritmos de aprendizaje automático supervisado. Dentro de aquellos artículos que fueron anotados tanto por un *experto* como por un *no-experto*, nos quedamos con los de los primeros. Este es el conjunto de archivos que se utilizó para entrenar y testear los algoritmos de aprendizaje automático supervisado.

4.4.4. *Inter-annotator agreement*

Antes de definir el *gold standard*[24] de las anotaciones para utilizarlos en los algoritmos de ML, se debe evaluar la tarea de anotación. Una práctica común es comparar las anotaciones realizadas por distintos pares de anotadores para validar y mejorar los esquemas y los criterios de anotación. Debido a que las anotaciones son creadas por los anotadores, no hay forma de precisar su correctitud comparando contra otro corpus de referencia. Por lo tanto, no se verifica la correctitud del proceso de anotación, sino que sea reproducible, es decir que si aplicamos el mismo proceso múltiples veces sobre la misma fuente con distintos anotadores (ya que los mismos podrían recordar sus anotaciones previas), deberíamos obtener resultados similares. El *inter-annotator agreement* es una medida utilizada comúnmente entre distintos pares de anotadores para medir esto. La misma indica

[24] El *gold standard* es un conjunto de datos anotados manualmente por especialistas que puede ser utilizado para evaluar sistemas de NLP. Debe haber sido anotado (o corregido) según el último criterio de anotación establecido.

qué tan consistente es nuestro proceso de anotación y qué tan claros son los criterios de anotación, y no necesariamente que el corpus anotado producido vaya a generar buenos resultados al ser utilizado en nuestros algoritmos de ML.

Una de las formas más simples para medir dicha consistencia es a través del acuerdo observado, que se basa en contar la cantidad de veces que cierto elemento contiene etiquetas idénticas y reportar dicho número como un porcentaje sobre el total a ser anotado. Este es fácil de medir y entender, pero no arroja un resultado sobre si el proceso es reproducible o no, ya que ciertas concordancias pueden ser accidentales. Una medida muy utilizada es el coeficiente *kappa de Cohen*, el cual utilizamos para evaluar la consistencia del *inter-annotator agreement*. Además, presentamos e informamos los resultados del F1-score (ver sección 3.5).

4.4.4.1. Coeficiente kappa de Cohen

El coeficiente κ mide la concordancia entre pares de anotadores teniendo en cuenta el hecho de que podrían haber asignado etiquetas de anotación por pura casualidad. Sea A_0 el acuerdo observado entre los anotadores (valor entre 0 y 1) y sea A_e el valor esperado si cada anotador hubiese elegido una etiqueta aleatoria. El valor del acuerdo entre anotadores sin tener en cuenta la arbitrariedad es de $A_0 - Ae$ (que podría ser negativo), siendo el máximo valor posible $1 - Ae$. La proporción entre ambos valores es un coeficiente que cuando es 1 indica un acuerdo perfecto y cuando es 0 indica un acuerdo totalmente arbitrario:

$$k = \frac{A_0 - Ae}{1 - Ae} \tag{4.1}$$

4.4.4.1.1 Cálculo de κ

Para calcular el coeficiente κ, utilizamos la herramienta de python *scikit-learn*[25] a nivel de token. Este calcula el valor del coeficiente κ tomando como base dos arreglos, cada uno representando las anotaciones realizadas por un anotador. El resultado es un número entre -1 y 1, con el máximo valor indicando acuerdo completo y 0 o menos indicando acuerdo por casualidad. Como tenemos que comparar tres tipos distintos de etiquetas (entidades, relaciones binarias y relaciones ternarias), se generaron tres arreglos distintos de longitud k (siendo k la cantidad de tokens hallados en un artículo particular) y fueron concatenados para realizar el llamado a la herramienta.

Una dificultad presente es que un término puede tener asociada distintas etiquetas (por ejemplo, *zika* puede ser una enfermedad y una causa al mismo tiempo, o una entidad puede estar involucrada en dos relaciones distintas), por lo que se implementaron dos versiones del coeficiente (lo cual llamamos *multi-labelling*): una considerando dos anotaciones como idénticas si y sólo si el mismo conjunto de etiquetas fue asignado a un término particular, y otro considerando la posibilidad de cada etiqueta por separado.

Otro problema fue al tener en cuenta cuándo comienza y termina una etiqueta (por ejemplo, no es lo mismo tener la entidad *"chagas, zika"* (DS) a tener las entidades *'chagas'* (DS) y *'zika'* (DS)). Para esto, se implementó una versión del coeficiente que utiliza el formato *IOB2* (inside-outside-beginning) y otra que no. Este formato consiste en agregar el prefijo *B-* al comienzo de la etiqueta para indicar que esa etiqueta es el comienzo de

[25] Sklearn Metrics Cohen Kappa Score. Disponible en: `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.cohen_kappa_score` [Accedido en Junio de 2020].

una secuencia, y el prefijo *I-* indica que esa etiqueta se encuentra dentro de la secuencia. La etiqueta *O* indica que ese token no pertenece a ningún fragmento de texto.

En la sección 8.4.1 se pueden observar detalles de implementación de cada una de las variantes del coeficiente. Para los resultados presentados, se utilizó la versión que considera dos anotaciones como iguales si y sólo si el mismo conjunto de etiquetas fue asignado al mismo término, y sin utilizar el formato IOB2.

4.4.4.1.2 Resultados

Si bien se realizaron 7 tandas de anotaciones, solo en las tres primeras se consideraron artículos en común para evaluar la consistencia entre las anotaciones de distintos anotadores. Para el cálculo del coeficiente de estos resultados, se decidió utilizar la versión más simple de κ, la cual no considera múltiples etiquetas para un mismo token, ni tiene soporte del formato *IOB2*.

En la tabla 4.1 podemos observar los valores del coeficiente κ para los archivos anotados en común[26]. Además, se muestran en la tabla la cantidad de artículos anotados en total, cuántos artículos distintos fueron anotados por más de una persona, cuántos fueron anotados en total (puede que un mismo artículo haya sido anotado por más de un experto o un no-experto), el κ promedio, mínimo y máximo.

It.	**# artículos anotados**	**# anotados por dos anotadores (sin repetir)**	**# anotados por dos anotadores (con repeticiones)**	**κ promedio**	**κ Mínimo**	**κ Máximo**
1	25	14	22	0.48	0.13	0.68
2	56	10	16	0.62	0.34	0.79
3	142	7	7	0.49	0.35	0.64
Total	**229**	**37**	**45**	**0.53**	-	-

Tab. 4.1: Resultados *inter-annotator agreement* (κ promedio, mínimo y máximo) junto con la cantidad de artículos anotados en cada una de las tres primeras iteraciones.

Utilizando la versión que calcula el κ para cada entidad por separado, se calculó el κ para cada artículo anotado por un experto y un no-experto para evaluar la consistencia por entidad[27]. Hay que tener en cuenta que los valores de κ (Cohen) no son los mismos que los de la tabla 4.1 ya que en este último se considera un κ global, mientras que en el de por entidad se toma, para cada artículo, el promedio sobre los kappa de cada entidad involucrada. En el gráfico 4.3 se pueden observar los resultados del κ de cada entidad y total (Cohen) de todos los artículos anotados por más de un anotador. Para información sobre cómo interpretar el gráfico, ir a la sección 8.2.

En los mismos, se puede evaluar cuáles son las entidades con mayor *agreement* (por ejemplo, *Date*, *Disease*, *Number of cases* y *Location*) y cuáles las que dan resultados bajos (por ejemplo, *Negation* y *Past*). De esta forma, se puede poner énfasis en las entidades para las cuales hay que mejorar los criterios de anotación. Posteriormente, al realizar el análisis del corpus anotado, se estudiarán las inconsistencias entre las anotaciones.

[26] Un artículo se considera que fue anotado en común si fue anotado por un experto y un no-experto.
[27] Se ignoraron los resultados de la entidad *Abbreviation* y *Conditional ya que daba* 0 *en los gráficos.*

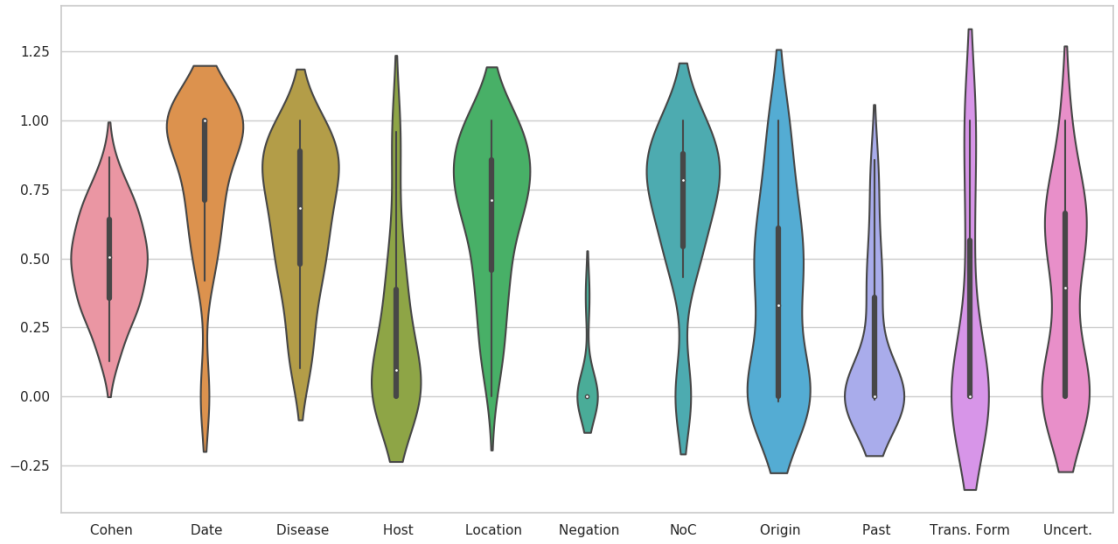


Fig. 4.3: κ por entidad para todos los artículos anotados por más de un anotador de todas las tandas anotaciones.

4.4.4.2. F1-score

El κ de Cohen es la medida estándar que se utiliza para medir la consistencia entre las anotaciones en problemas de clasificación. Sin embargo, existen distintos trabajos que sostienen que no es la mejor medida para la tarea de NER [34, 23]. Una de las razones expuestas es debido a la necesidad de conocer los casos negativos (en nuestro caso de estudio, un caso positivo es un término (una o más palabras) anotado, mientras que los casos negativos son aquellos no relevantes). Para la tarea de NER, los casos negativos son inciertos ya que las etiquetas abarcan distintas cantidades de tokens y pueden superponerse, por lo que no es posible conocer todos los casos negativos. Una posible solución a lo mismo es, a la hora del cálculo, considerar un solo token a la vez y computar el valor de κ a nivel de token. Esta opción tiene dos desventajas: (1) no refleja correctamente la tarea de anotación ya que los anotadores no asignan etiquetas individualmente a tokens, sino que evalúan secuencias de ellos; y (2) la cantidad de tokens no anotados (casos negativos) va a ser mucho mayor a los casos positivos, por lo que el valor del κ no va a encontrarse balanceado (pues los tokens con la etiqueta 'O' que indica que no tiene etiqueta va a ocupar la mayoría de los datos). Existen otros tipos de soluciones para utilizar esta medida considerando los casos negativos, como considerar todos los posibles n-gramas. Por estas razones, se realizaron los cálculos del F1-score, además del κ.

4.4.4.2.1 Match exacto y match parcial

Como fue explicado en la sección 3.5, el F1-score tiene en cuenta dos métricas principales: *precision* que es la proporción de casos que fueron clasificados como positivos y realmente lo eran, y *recall* que es la proporción de casos positivos que efectivamente fueron clasificados como tales. Luego, el F1-score es la media armónica entre *precision* y *recall*. Dado que esta medida es utilizada para la tarea de NER al comparar las etiquetas anotadas entre pares de anotadores, únicamente la utilizamos para medir las entidades anotadas

y no las relaciones. Esto genera que los resultados obtenidos no abarquen toda la tarea de anotación, a diferencia del κ que sí considera las relaciones binarias y ternarias.

Para esta medida, se puede utilizar un *match exacto* o un *match parcial*. En el primero, dos anotaciones se consideran iguales si y sólo si abarcan los mismos términos y el tipo de entidad asignado es el mismo. El segundo caso consiste en asignar un porcentaje del match (por ejemplo, $\frac{1}{2}$) si el match no es exacto. Existen diversas variantes sobre cuándo considerar un match exacto, por ejemplo si los términos abordados no son exactamente iguales (puede ser que una anotación comience en un término anterior o termine en uno posterior) o si a un término se le asignaron distintas entidades (por ejemplo, un anotador anota *zika* como *Disease* y otro como *Origin*). Utilizamos tanto el match exacto como parcial como medida, utilizando en el caso del match parcial solo la primer variante (entidades que abarcan distintos términos, con algunos superpuestos) y no la segunda (mismos términos pero con distintas etiquetas). Más detalles de su implementación pueden encontrarse en la sección 5.3.

4.4.4.2.2 Resultados

Se calculó el F1-score sobre las entidades. En la tabla 4.2 se encuentra el resultado del match exacto y en la tabla 4.3 el del parcial. Estos resultados no deben compararse contra los del κ ya que estos últimos tienen en cuenta las relaciones a diferencia del F1-score calculado aquí.

It.	**# artículos anotados**	**# anotados por dos anotadores (sin repetir)**	**# anotados por dos anotadores (con repeticiones)**	**F1-score ponderado**	**F1-score Mínimo**	**F1-score Máximo**
1	25	14	22	0.57	0.14	0.79
2	56	10	16	0.59	0.41	0.77
3	142	7	7	0.5,1	0.23	0.71
Total	**229**	**37**	**45**	**0.57**	-	-

Tab. 4.2: Resultados F1-score match exacto de solo entidades (ponderado, mínimo y máximo) junto con la cantidad de artículos anotados en cada una de las tres primeras iteraciones.

4.5. Análisis del corpus anotado

Una vez que las anotaciones fueron realizadas, se analizó el conjunto de los datos anotados para saber cuántas entidades y relaciones de cada tipo fueron encontradas. Debido a que el título de cada artículo contiene bastante información, se trabajó con dos conjuntos de datos:

1. **Artículo reducido:** contiene solo el título y la fecha. Este será llamado de ahora en adelante: *'título'*.

2. **Artículo entero:** contiene el título, la fecha y el cuerpo del artículo.

It.	**# artículos anotados**	**# anotados por dos anotadores (sin repetir)**	**# anotados por dos anotadores (con repeticiones)**	**F1-parcial**	**F1-parcial Mínimo**	**F1-parcial Máximo**
1	25	14	22	0.64	0.18	0.88
2	56	10	16	0.67	0.49	0.84
3	142	7	7	0.56	0.32	0.70
Total	**229**	**37**	**45**	**0.64**	-	-

Tab. 4.3: Resultados F1-score match parcial de solo entidades (ponderado, mínimo y máximo) junto con la cantidad de artículos anotados en cada una de las tres primeras iteraciones.

En la tabla 4.4 se pueden observar la cantidad total de artículos anotados, el promedio de oraciones por artículo, el promedio de palabras por artículo, el promedio de palabras por título y la longitud promedio de las palabras (cantidad de caracteres).

# artículos anotados	**# oraciones promedio por artículo**	**# palabras promedio por artículo**	**# palabras promedio por título**	**# caracteres promedio por palabra**
513	10	334	10	5

Tab. 4.4: Información sobre el corpus anotado.

En la tabla 4.5 presentamos la cantidad de artículos que fueron anotados por enfermedad y por país. Para realizar la misma, se analizó el título de los artículos y se extrajeron las enfermedades y países a los que se hace mención (razón por la cual existen más artículos en el total que en los anotados, ya que en ciertos títulos se hace mención a más de un país y/o enfermedad).

Ubicación	**Chagas**	**Dengue**	**Guillain-barré**	**Zika**	**Hantavirus**	**Sarampión**	**Microcefalia**	**Total**
Argentina	8	7	1	12	41	11	1	**81**
Bolivia	4	0	0	2	14	1	1	**22**
Brasil	12	9	2	39	17	40	17	**136**
Chile	1	3	0	3	32	9	0	**48**
Colombia	4	2	0	28	8	14	7	**63**
Costa Rica	0	1	0	0	7	5	0	**13**
Cuba	13	11	3	32	1	42	9	**111**
Ecuador	0	0	0	2	0	12	0	**14**
El Salvador	1	0	1	4	0	0	0	**6**
Guatemala	0	3	0	3	3	0	1	**10**
Honduras	0	12	0	8	1	0	5	**26**
México	0	14	3	6	7	6	1	**37**
Nicaragua	0	3	0	1	0	0	0	**4**
Panama	0	3	0	2	18	0	1	**24**
Paraguay	1	5	0	1	8	0	0	**15**
Perú	1	21	6	5	11	8	0	**52**
República Dominicana	0	1	1	3	0	0	0	**5**
Uruguay	0	14	0	0	3	2	0	**19**
Venezuela	10	1	3	12	0	34	3	**63**
Total	**55**	**110**	**20**	**163**	**171**	**184**	**46**	**749**

Tab. 4.5: Cantidad de enfermedades por país, según la información del título.

Entidades

En la tabla 4.6 presentamos la cantidad de entidades y modificadores anotados para el título y el artículo entero. Para los valores de la tabla, dos entidades anotadas se consideran iguales si y sólo si contienen los mismos términos (por ejemplo, si un anotador anotó la entidad *[ratón colilargo]* y el segundo anotó solamente *[colilargo]*, no serán considerados iguales).

Entidad	**Títulos**		**Art. enteros**	
	Total	**Diferentes**	**Total**	**Diferentes**
Abbreviation (AB)	4	2	10	8
Date (DT)	491	462	613	550
Disease (DS)	560	35	3053	229
Host (HT)	33	19	678	357
Location (LOC)	573	189	2611	884
Number of cases (NoC)	10	5	2176	758
Origin (OR)	47	21	778	270
Transmission form (TF)	30	15	293	168
Conditional (COND)	0	0	3	3
Negation (NT)	3	2	74	37
Past (PT)	1	1	504	328
Uncertainty (UT)	17	8	400	197
Total	**1769**	**759**	**11193**	**3789**

Tab. 4.6: Tipos y cantidad de entidades y modificadores halladas por los anotadores en los títulos y los artículos enteros.

En las tablas 4.7 y 4.8 se pueden observar algunos de los términos frecuentes hallados por los anotadores para ciertas entidades, junto a la cantidad de ocurrencias. Esto fue realizado tanto para los títulos como para los artículos enteros. Para las entidades seleccionadas, se muestran los n términos más frecuentes de manera descendente por cantidad de ocurrencia, no mostrando así todos los términos existentes[28]. Para el caso de los títulos, al poseer pocas ocurrencias de términos, se suelen mostrar alrededor de diez términos frecuentes, llegando usualmente a los términos con una sola ocurrencia. En el caso de los artículos enteros, se suelen mostrar alrededor de veinte términos, usualmente hasta alcanzar aquellos con una o dos ocurrencias para alguna entidad. Además, se intenta ser consistente en la cantidad de términos exhibidos, es decir que se muestra la misma cantidad de términos frecuentes por cada entidad, en caso de que se posea la información (es decir, que existan tantos términos a mostrar). Este criterio de corte se utilizó en todas las tablas de este tipo en el presente trabajo.

[28] En caso de tener solo una ocurrencia, se seleccionaron los términos de manera aleatoria.

Disease	**# ocur.**
hantavirus	125
sarampion	121
zika	116
dengue	91
chagas	38
microcefalia	26
guillain barré	9
sindrome de guillain-barré	2
chikungunya	2
chagas oral	2

Host	**# ocur.**
niños	8
embarazadas	5
embarazada	3
adultos	2
trabajadores	1
donantes de sangre	1
recien nacidos	1
trabajadores de mercado de alimentos	1
comunidades vulnerables	1
inmigrantes	1

Transmssion form	**# ocur.**
transmision oral	9
via oral	5
vectorial	3
transmision interpersonal	2
adultos y machos	1
brote oral	1
transmision sexual	1
transmision vertical	1
trasmision sexual	1
trasmision oral	1

Origin	**# ocur.**
zika	16
no-vacunados	4
brote	3
brote familiar	2
vectores	2
acai contaminado	2
importado	2
acai	2
mosquitos	2
vectorial	1

Tab. 4.7: Entidades comúnmente halladas por los anotadores en los títulos, junto a la cantidad de ocurrencias.

Disease	**# ocur.**
zika	659
dengue	481
sarampion	472
hantavirus	342
microcefalia	195
chagas	148
enfermedad	120
virus	62
guillain-barre	43
chikungunya	35
virus hanta	26
guillain barre	23
hanta	22
brote	18
dengue hemorragico	14
hanta virus	13
síndrome de guillain-barre	12
hantavirosis	12
casos	11
zikv	11
chikunguna	8
rubéola	8
mal	8

Host	**# ocur.**
niños	58
bebés	49
embarazadas	34
mujeres embarazadas	22
recién nacidos	18
bebé	18
adultos	14
niño	10
mujeres	9
fetos	9
mujer	8
madres	7
gestantes	7
feto	6
menores de edad	6
embarazada	5
joven	5
adulto	5
mujer de 27 años	4
madre	4
bebés recién nacidos	3
neonato	3
niña	3

Transmission form	**# ocur.**
picadura	19
transmisión oral	14
vía oral	12
vectorial	8
consumo	7
transmisión sexual	7
mosquito _aedes aegypti_	6
congénito	5
ingestión	5
aedes aegypti	5
orina	4
ingesta	4
sexualmente	4
congénita	4
aedes aegypti	3
mosquitos	3
ratones colilargos	3
transmisión vectorial	3
picaduras	3
picado	2
vía sexual	2
picaduras de mosquitos	2
brote oral	2

Origin	**# ocur.**
zika	134
mosquito	75
aedes aegypti	46
mosquitos	37
roedores	35
mosquito _aedes aegypti_	20
zancudo	18
vinchucas	12
vinchuca	12
virus	12
aedes aegypti	10
vectores	8
virus zika	7
brote	7
mosquito aedes aegypti	6
importado	6
roedor	5
ratón	5
ratas	5
hantavirus	5
ratones	5
insecto	5
trypanosoma cruzi	4

Tab. 4.8: Entidades comúnmente halladas por los anotadores en los artículos enteros, junto a la cantidad de ocurrencias.

Al analizar la tabla 4.6, se puede observar que en los títulos, la mayor parte de la información que se puede obtener es sobre *Disease* (560), *Location* (573) y *Date* (491). Sin embargo, otra información de interés como *Host* (33), *Number of cases* (10), *Origin* (47) y *Transmission form* (30) no se encuentra en cantidad, por lo que es importante recurrir al artículo entero. Por ejemplo, según los reportes encontrados en las notas periodísticas, las enfermedades afectan más a niños y embarazadas que a bebés, ya que en las primeras 10 apariciones solo se encuentra una referencia a un recién nacido. Sin embargo, si se observa la tabla 4.8, allí se pueden ver más apariciones de portadores bebés (49 para el término *bebés* y 18 para *bebé*, más otros valores como *neonato* o *recién nacidos*). En cuanto a *Transmission form*, si solo evaluásemos el título, no encontraríamos las *picaduras* de mosquitos, cuando en el artículo entero se muestra como una de las principales formas de transmisión (19 para el término *picadura*). En cuanto a *origin*, en los títulos no se encuentra *aedes aegypti*, cuando se puede ver en los artículos enteros que es uno de los principales causantes (46 para el término *_aedes aegypti_* y 20 para *mosquito _aedes aegypti_*, entre otras formas de referirse al mismo).

A la hora de analizar por qué se encontraron tantos términos diferentes de cada entidad, un factor que influye es el de los errores ortográficos en los artículos, por ejemplo: *zica*, *guillaim-barre*, *sidrome de guillain-barre*, *gullain barre* y *embarada*.

En cuanto a los términos hallados para los distintos tipos de entidades, se puede observar lo siguiente:

- **Disease:** en el caso de los artículos enteros, que se hayan anotado 229 entidades distintas de este tipo siendo que las de interés son solo 7, se debe a las siguientes razones:
  - Existen distintas formas para referirse a una misma. Por ejemplo, para la enfermedad *guillain-barré*, se encontraron las siguientes formas: *sindrome guillain barre*, *sindrome de guillain-barre*, *sindrome de guillain - barre*, *guillain-barre*, *guillain barre* etc.
  - Existen distintas variaciones de una misma enfermedad. Por ejemplo, para el caso del dengue, hay distintos tipos como *dengue denv-1*, *dengue serotipo den-4*, *denv-1*, *dengue hemorrágico tipo 2*, *dengue salvaje*, etc.
  - Se han anotado otros tipos que no son de interés, por errores en las anotaciones. Por ejemplo: *poliomielitis*, *ceguera*, *glaucoma*, *neumonía*, *rubéola*, etc.
  - Se anotan, siguiendo el criterio de anotación, referencias a la enfermedad para así poder establecer relaciones. Algunas de estas fueron: *enfermedad*, *mal*, *infección* y *virus*, entre otras.
- **Host:** como se puede observar en la tabla 4.8, la mayoría son *embarazadas*, *niños* y *bebés*, y las variaciones para referirse a ellos. Se encontraron distintas formas de referirse a embarazadas (*embarazadas*, *mujeres embarazadas*, *embarazada*) y otra encontrada con menos incidencia fue *gestante*. Para los bebés, se encontraron comúnmente *bebé/s*, *recién nacido/s*, *bebés recién nacidos*, y otros hallados con menor frecuencia fueron *"bebé de X meses"* (con X un número entre 1 y 12), *"bebés por nacer"* y *"neonato"*. Para *los niños*, se encontraron además *"niño/a de X años"*, *"niños/as de X años"*, *"niños y niñas menores de X años"* (X siendo una edad). Otro tipo de *hosts* aislados encontrados son *turista*, *migrantes*, *trabajador rural* y *comunidades indígenas*, entre otros.

- **Origin:** las más comunes son el zika (que causa microcefalia), mosquitos (que causan dengue y zika) y roedores (que causan hantavirus, entre otros), y las distintas variaciones de los mismos. Por ejemplo, para referirse a los ratones, se encontró: *colilargos*, *roedor/es*, *ratón*, *roedores silvestres*, *roedores de cola larga*, *ratas*, *calomys laucha*, *rattus norvegicus*. Para los mosquitos, parásitos, virus e insectos encontramos: *zancudo*, *mosquito*, *vector*, *vectores*), *chipo*, *kissing bug*, *_t. cruzi_*, *tripomastigotes de _trypanosoma cruzi_*, *_aedes aegypti_*, *_culex quinquefasciatus_*, *vinchuca*, *mosquitos del genero _aedes_*. Otros tipos de causas halladas fueron el *zumo de caña de azucar*, *açaí*, *vino de açaí artesanal* e *importado* (es decir, que no se generó en el lugar en el que fue detectado).
- **Transmission form:** fuera de las reflejadas en la tabla, también encontramos *beber*, *trasplante de órganos*, *donación de sangre*, *transfusión sanguínea*, *lesiones en la piel*, entre otras. Aquí se puede observar que hubo dificultades para diferenciar las entidades de tipo *transmission form* y *origin*, anotándose erróneamente una por la otra. Por ejemplo, se anotaron entidades de tipo *Transmission form* erróneas como *mosquitos*, *aedes aegypti* y variaciones para referirse a *ratas*. Otro problema es que la entidad *'picaduras de mosquitos'* debería estar separada en una entidad de tipo *transmission form* (*picaduras*) y otra de tipo *origin* (*mosquitos*), y hubo tendencia a que se anotase una sola entidad.
- **Conditional:** se pueden observar solo tres anotaciones en los artículos enteros, las cuales corresponden a los términos: *"podría deberse"*, *"si no hay acciones continuas de control"* y *"si existen"*. Sin embargo, estas mismas no se encuentran relacionadas con ninguna otra entidad, por lo que violan el criterio de anotación.
- **Past:** Se encontraron casos de términos de fechas que deberían haber sido anotados como *Past*, ya que eran anteriores a la fecha indicada en el artículo, pero fueron anotadas como *Date*.
- Se encontraron ciertas inconsistencias en las anotaciones a la hora de delimitar la cantidad de términos abarcados por una entidad. Algunos casos fueron los siguientes:
  - Para el término *'casos sospechosos'*, hubo anotadores que anotaron la entidad *'sospechosos'* (UT) y otros que anotaron *'casos sospechosos'*. Similarmente, en el término *'mujeres embarazadas'*, algunos anotaron solamente *'embarazadas'*, mientras que otros *'mujeres embarazadas'*.
  - Términos que deberían haber sido anotados como parte de una sola entidad, pero fueron separados en más. Por ejemplo, en el caso *'Brasil (Sao Paulo)'*, se encontraron las anotaciones: *'Brasil'* (LOC) y *'Sao Paulo'* (LOC), y *'Brasil (Sao Paulo)'* (LOC).

Luego del análisis de las entidades anotadas, se puede observar que hubo cierta dificultad para detectar varias de las entidades (como *Origin* y *Transmission form*) o para definir los términos que abarcaban otras (como *Host* y *Uncertainty*). Esto demuestra la necesidad de continuar iterando sobre los criterios de anotación para mejorar la consistencia del corpus y la importancia de haber implementado el coeficiente κ de *Cohen* para evaluar cuáles son las entidades para las cuales se encontraron más problemas a la hora de anotar.

Relaciones

En las tablas 4.9 y 4.10 presentamos las relaciones (binarias y ternarias), las entidades involucradas en cada una de ellas, el total de relaciones y la cantidad de relaciones diferentes que fueron halladas en los títulos y los artículos enteros, respectivamente. Para los valores de la tabla, dos relaciones se consideran iguales si ambas entidades anotadas contienen los mismos términos (por lo que si hubo un match parcial[29] de la relación no va a verse reflejado en dicha tabla) y el tipo de relación es el mismo.

Relación	**Entidades**	**Totales**	**Diferentes**
Disease occurs in (DsOccIn)	DS-LOC	573	254
Negates disease (NegDs)	NT-DS	1	1
Negates origin (NegOr)	NT-OR	2	1
Cases of disease (NoCDs)	NoC-DS	8	5
Cases in location (NoCLoc)	NoC-LOC	4	4
Occurs to (OccTo)	DS-HT	36	26
Cause of disease (OrDs)	OR-DS	38	25
Origin occurs in (OrOccIn)	OR-LOC	14	13
Transmission of disease (TfDs)	TF-DS	14	8
Transmission in location (TfOccIn)	TF-LOC	7	6
Transmission cause (TfOr)	TF-OR	8	6
Speculates disease (UcDs)	UT-DS	6	5
Speculates cause (UcOr)	UT-OR	5	5
Speculates transmission form (UcTf)	UT-TF	4	4
Speculates cause of disease (UcOrDs)	UcTfOr-OR-DS	4	4
Speculates transm. form of disease (UcTfDs)	UcTfDs-DS-TF	3	3
Total		**727**	**370**

Tab. 4.9: Relaciones halladas por los anotadores en los títulos.

[29] Un *match parcial* se da cuando los términos de ambas entidades no son exactamente iguales, pero sí similares, siendo así parcialmente correcto. Existen distintas maneras de establecer cuándo dos términos son similares, las cuales dependen de quién lo implemente. Por ejemplo, si un anotador encontró la entidad [ratón colilargo] (Origin) y el segundo anotador solamente [ratón] (Origin), en un match parcial esto se consideraría parcialmente correcto y no erróneo como en un match exacto.

Relación	**Entidades**	**Totales**	**Diferentes**
Disease occurs in (DsOccIn)	DS-LOC	1838	924
Date of disease (DtDs)	DT-DS	89	75
Negates disease (NegDs)	NT-DS	30	28
Negates location (NegLoc)	NT-LOC	1	1
Negates number of cases (NegNoC)	NT-NoC	28	26
Negates cause (NegOr)	NT-OR	18	16
Negates transmission form (NegTf)	NT-TF	13	12
Cases of disease (NoCDs)	NoC-DS	1164	817
Cases of host (NoCHt)	NoC-HT	214	198
Cases in location (NoCLoc)	NoC-LOC	810	773
Occurs to (OccTo)	DS-HT	408	254
Cause of disease (OrDs)	OR-DS	596	300
Origin occurs in (OrOccIn)	OR-LOC	66	60
Past disease (PtDs)	PT-DS	203	189
Past cases (PtNoC)	PT-NoC	389	379
Transmission of disease (TfDs)	TF-DS	105	77
Transmission in location (TfOccIn)	TF-LOC	18	17
Transmission cause (TfOr)	TF-OR	149	134
Speculates disease (UcDs)	UT-DS	89	75
Speculates number of cases (UcNoC)	UT-NoC	271	249
Speculates cause (UcOr)	UT-OR	58	54
Speculates transmission form (UcTf)	UT-TF	18	17
Negates transmission cause (NegTfOr)	NegTfOr-TF-OR	3	3
Speculates cases of disease (UcNoCDs)	UcNoCDs-NoC-DS	60	59
Speculates cause of disease (UcOrDs)	UcTfOr-DS-OR	36	36
Speculates transm. form of disease (UcTfDs)	UcTfDs-DS-TF	5	5
Total		**6679**	**4778**

Tab. 4.10: Relaciones halladas por los anotadores en los artículos enteros.

Similar al análisis realizado anteriormente con las entidades, al evaluar los resultados de las relaciones de los títulos con respecto al artículo entero, se pierde información que puede resultar valiosa. Gran parte de las relaciones encontradas son de lugares en donde ocurren las enfermedades ($DsOccIn$), ocupando el 79 % (573) de las relaciones totales halladas (727). Sin embargo, si se quiere observar información como las causas o formas de transmisión de una enfermedad (OrDs y TfDs), o la cantidad de casos de una enfermedad (NoCDs), convendría recurrir a los artículos enteros, ya que los títulos no cuentan con mucha información (por ejemplo, de la relación NoCDs en los títulos solo contamos con 8 totales, mientras que en los enteros tenemos 1164).

En las tablas 4.11 y 4.12 se pueden observar algunas de las relaciones binarias comúnmente halladas por los anotadores (indicando los términos de las entidades involucradas en la relación), tanto para los títulos como los artículos enteros, respectivamente, utilizando el mismo criterio de corte que para las entidades.

OrDs	**# ocur.**
zika-microcefalia	9
no-vacunados-sarampion	3
importado-sarampion	2
brote-hantavirus	2
mosquitos-zika	2
vectorial-chagas	1
brote familiar-chagas	1
acai contaminado-chagas	1

TfDs	**# ocur.**
transmision oral-chagas	5
via oral-chagas	3
transmision sexual-zika	1
transmision vertical-microcefalia	1
trasmision sexual-zika	1
trasmision oral-chagas	1
consumo-chagas	1
trasmision vertical-chagas	1

OccTo	**# ocur.**
sarampion-ninos	6
zika-embarazadas	5
hantavirus-embarazada	2
zika-recien nacidos	1
microcefalia-embarazada	1
microcefalia-ninos	1
dengue-embarazada	1
hantavirus-trabajadores	1

TfOr	**# ocur.**
transmision oral-acai contaminado	2
transmision interpersonal-brote	2
brote oral-vino de acai artesanal	1
trasmision oral-acai	1
consumo-acai	1
via oral-acai	1

Tab. 4.11: Relaciones comúnmente halladasen los títulos por los anotadores, junto a la cantidad de ocurrencias.

Al evaluar los términos de las entidades involucradas en las relaciones anotadas, se pueden observar ciertos hechos (por ejemplo, causas más comunes de una enfermedad o formas de transmisión a partir de los artículos analizados), aunque de por sí solos no basta, ya que podría haber una entidad negada o anotada como incierta, por lo que un análisis más profundo debe ser realizado.

En estas tablas, podemos observar por ejemplo las relaciones de tipo *DsOr*, que relacionan una enfermedad con su causa. Aquí se puede ver que la *microcefalia* es causada por el *zika*, o que el *dengue* es generado por un *mosquito*. Si se analiza la relación *TfOr* (que relaciona una causa con la forma en la que se transmite), podemos ver que el *mosquito* genera una *picadura* o que el *açaí contaminado* se transmite oralmente. Si se analiza *OccTo*, se puede observar, por ejemplo, que la *microcefalia* afecta a *bebés* y el *zika* a *embarazadas*.

OrDs	**# ocur.**
zika-microcefalia	73
mosquito-dengue	22
aedes aegypti-dengue	21
mosquito-zika	18
aedes aegypti-zika	14
mosquito _aedes aegypti_-zika	12
mosquito _aedes aegypti_-dengue	10
zika-guillain-barré	10
mosquitos-zika	10
mosquito-enfermedad	8
mosquito _aedes aegypti_-chikungunya	8
mosquitos-dengue	7
roedores-hantavirus	7
mosquito aedes aegypti-dengue	6
virus-microcefalia	5

TfDs	**# ocur.**
transmisión oral-chagas	7
congénito-chagas	4
vía oral-chagas	4
picadura-dengue	4
transmisión sexual-zika	4
picadura-zika	3
sexualmente-zika	3
ingestión-chagas	2
relaciones sexuales-zika	2
trasmisión oral-chagas	2
consumo-chagas	2
picada-zika	2
mosquito _aedes aegypti_-dengue	2
transmisión directa de sus madres-chagas	1
transmisión vertical (de madre a feto)-zika	1

OccTo	**# ocur.**
microcefalia-bebés	23
zika-embarazadas	20
zika-bebés	17
microcefalia-recién nacidos	12
sarampión-niños	12
microcefalia-niños	10
zika-mujeres embarazadas	9
microcefalia-bebé	9
zika-madres	6
microcefalia-fetos	5
zika-niños	5
zika-adultos	4
enfermedad-niños	3
zika-gestantes	3
zika-mujeres	3

TfOr	**# ocur.**
picadura-mosquito	6
picadura-_aedes aegypti_	4
mosquitos-zika	2
transmisión oral-acai contaminado	2
orina-roedores	2
congénitos-zika	2
ingestión-jugo de caña de azucar	2
transmisión interpersonal-brote	2
picadura-vector	2
penetra en el organismo humano a traves de las vías respiratorias o de heridas-ratón	1
picado-mosquito	1
vía sexual-zika	1
respirar-lugares muy cerrados como galpones	1
contaminación-baya de acai	1
consumo-jugo de acai	1

Tab. 4.12: Relaciones comúnmente halladas en los artículos enteros por los anotadores, junto a la cantidad de ocurrencias.

5. EXTRACCIÓN DE ENTIDADES NOMBRADAS

En este capítulo se desarrolla el trabajo realizado para la extracción automática de entidades nombradas. Inicialmente, realizamos una introducción y trabajos previos sobre el tema. Luego, presentamos medidas de calidad. Seguido de eso, desarrollamos sobre un algoritmo basado en reglas y otro en redes neuronales profundas. Finalmente, se presenta una comparación entre ambos métodos implementados.

5.1. Introducción

Como fue mencionado en el capítulo 3, el *reconocimiento de entidades nombradas* es una tarea que consiste en localizar y clasificar en ciertas categorías predefinidas las entidades nombradas en un texto. Es utilizada para distintas tareas de NLP como extracción de información o traducción automática (para poder traducir ciertas entidades). El término *entidad nombrada* fue introducido por primera vez en 1995 en los Estados Unidos en la sexta conferencia *Message Understanding Conference* (MUC-6) [31]. Como se explica en dicho trabajo, estas conferencias nacieron con el fin de analizar automáticamente mensajes militares textuales. Se enfocaron en reconocer información sobre entidades (personas, organizaciones y ubicaciones geográficas), expresiones temporales (fechas y horas) y expresiones numéricas (monetarias y porcentajes). Con el paso del tiempo, se fueron expandiendo las categorías de las entidades nombradas. Estas van a depender del escenario de aplicación ya que, por ejemplo, si el dominio es médico, una categoría de interés será la de *nombre de enfermedades*, mientras que si el dominio es la bioinformática, reconocer las proteínas y genes es crucial. Mientras más amplia sea la tarea que se busca resolver, más categorías serán necesarias.

Normalmente, resolver este problema involucra dos tareas: (1) la detección de la entidad en el texto, que consiste usualmente en definir un bloque contiguo de términos que forman parte de la entidad; y (2) la clasificación, que consiste en asignarle a cada entidad identificada alguna categoría predefinida (por ejemplo, persona, ubicación, fecha).

El resto del capítulo está organizado de la siguiente manera. Iniciamos con trabajos previos en la sección 5.2. Luego, hablamos de medidas de calidad en la sección 5.3. Luego, desarrollamos un algoritmo basado en reglas en la sección 5.4, para el cual utilizamos la librería *FreeLing* en conjunto con expresiones regulares y reglas manualmente realizadas para la detección de los mismos. Luego, introducimos un algoritmo basado en redes neuronales en la sección 5.5, para el cual utilizamos los artículos manualmente anotados en la sección 4.4 para el entrenamiento y testing del mismo. En cada uno de los métodos, presentamos los resultados obtenidos y un análisis sobre los mismos. Finalmente, realizamos una comparación de los métodos en la sección 5.6.

5.2. Trabajos previos

Uno de las formas de desarrollar la tarea de extracción de entidades nombradas es utilizando **métodos basados en diccionarios**, que consisten en tener un diccionario con entidades nombradas y, cuando se detecta una posible entidad, se verifica allí si existe o no.

Estos no tienden a ser performantes, ya que los diccionarios no cubren todos los casos de entidades nombradas (por ejemplo, nombres de personas), las entradas de los diccionarios van cambiando (por ejemplo, con nuevas adiciones), y, por más que se encuentren todas las palabras en el diccionario, la entidad está sujeta al contexto en el que se encuentra (por ejemplo, cómo saber si *'abril'* se refiere a un nombre o al mes).

Otra forma es utilizando **métodos basados en reglas**, el cual fue implementado en este trabajo. Este consiste en establecer una serie de reglas, ya sea manualmente o de manera automatizada. Estos sistemas pueden utilizar patrones gramaticales (por ejemplo, POS-tagging), rasgos ortográficos (por ejemplo, capitalización), y hasta incluso combinarse con diccionarios [14].

También existen los **métodos estadísticos basados en aprendizaje automático**, que convierten el problema de identificación en un problema de clasificación, y utilizan modelos estadísticos de clasificación para resolverlos a través de algoritmos de ML. Uno de los tipos de modelos de ML son los **métodos supervisados**, que utilizan un conjunto de datos de entrenamiento etiquetados para que el algoritmo aprenda a clasificar datos no etiquetados. Existen distintas propuestas utilizando métodos supervisados para el reconocimiento de entidades nombradas: algunos utilizan Modelos Ocultos de Markov para detectar nombres [9], otros utilizan modelos de máxima entropía [11], otros utilizan *support vector machines* [70] y otros utilizan árboles de decisión [7]. Este tipo de métodos requieren un gran conjunto de datos etiquetados manualmente, lo cual puede resultar muy costoso.

Los **métodos no supervisados** son otro tipo de modelos de ML en el que el modelo aprende a crear representaciones de los datos sin la necesidad de datos etiquetados. Dentro de este tipo de enfoque, nos encontramos con el trabajo de Collins y Singer [21] que propone un modelo no supervisado para realizar NER sin datos etiquetados, o el de Kim, Kang y Choi [41] que propone un modelo sin supervisar para clasificar entidades utilizando un diccionario de entidades nombradas y un corpus sin etiquetar. Si bien no suelen ser totalmente sin supervisión, tienen la ventaja de que generalizan bien a nuevos dominios e idiomas.

Finalmente, los **métodos híbridos** combinan enfoques basados en reglas con métodos de aprendizaje automático, utilizando los puntos fuertes de cada método. Por ejemplo, Srihari [60] utilizan un sistema híbrido utilizando Modelos Ocultos de Markov (HMM), modelos de máxima entropía (MaxEnt) y reglas gramaticales hechas a mano.

5.3. Medidas de calidad

Para medir la calidad del algoritmo, utilizamos la medida F1-score (ver sección 3.5). Dentro del mismo, se encuentra la forma *micro* y *macro*. El F1-score macro computa los valores de *precision* y *recall* por cada clase, y luego toma como valor final el promedio entre ellas. Por otro lado, el F1-score micro computa los valores de TP, FP y TN de cada clase por separado y utiliza los mismos para los cálculos de *precision* y *recall*, lo cual es útil si la distribución de las mismas no se encuentra balanceada.

Por ejemplo, supongamos que tenemos solamente las entidades A y B. Luego, los valores de *precision* y *recall* para el cómputo utilizando las distintas variantes son:

$$P_{macro} = \frac{P_A + P_B}{2}$$
$$R_{macro} = \frac{R_A + R_B}{2}$$
$$P_{micro} = \frac{TP_A + TP_B}{TP_A + TP_B + FP_A + FP_B}$$
$$R_{micro} = \frac{TP_A + TP_B}{TP_A + TP_B + FN_A + FN_B}$$

Finalmente, se utilizan dichos valores de *precision* y *recall* para calcular el F1-score micro o macro final a través de la media armónica. Dado que nuestras clases no se encuentran balanceadas, nos interesa el F1-score micro y será siempre el valor por defecto.

Para el cálculo del F1-score, necesitamos que nuestras anotaciones se encuentren en un formato particular, el cual se basa en un archivo que contiene en cada línea un token, su PoS-tag (dado que no nos interesa, utilizamos una etiqueta genérica "TAG") la etiqueta asignada por el gold estándar y la etiqueta asignada por el algoritmo (ambos en formato IOB2). Para obtener el mismo, dado que nuestras anotaciones manuales se encuentran en formato *brat* y las necesitamos comparar contra nuestro algoritmo, llevamos inicialmente cada uno de estos archivos a un formato intermedio. Para esto, utilizamos el script *anntoconll.py* que se encuentra dentro de la librería de brat[30]. Dicho script utiliza los pares de archivos de tipo *$id.ann* y *$id.txt* (siendo *id* el nombre del archivo), y genera un archivo final (nuestro intermedio) que contiene en cada línea un *token*, la etiqueta genérica "TAG" y la etiqueta asignada en formato IOB2. Por ejemplo, si tuviésemos el siguiente texto en un archivo .txt con sus correspondientes anotaciones:

Fecha: 24[B-Date] de[I-Date] abril[I-Date],[I-Date] 2019[I-Date]

Obtendríamos el siguiente archivo:

Palabra	TAG	Etiqueta anotada
Fecha	TAG	O
:	TAG	O
24	TAG	B-Date
de	TAG	I-Date
abril	TAG	I-Date
,	TAG	I-Date
2019	TAG	I-Date

Ciertas modificaciones tuvieron que realizarse al script de *brat* ya que, por ejemplo, no consideraba que el texto pudiese contener vocales con tildes o la letra *ñ*. Otro problema se daba cuando teníamos entidades superpuestas (por ejemplo, *zika* puede ser *Disease* u *Origin*), de manera exacta o parcial. El algoritmo, originalmente, considera dos casos: (1) si todas son del mismo tamaño, descarta todas y (2) en caso de ser de distinto tamaño (es decir, que abarcan distinta cantidad de tokens), se queda con aquella más larga. La modificación que se realizó fue en el caso en el que sean del mismo tamaño, en donde en vez de descartarla, nos quedamos con la que primero se encuentre alfabéticamente.

[30] Se puede encontrar en `https://github.com/nlplab/brat/blob/master/tools/anntoconll.py` [Accedido en Marzo de 2021]

Como lo que necesitamos comparar son dos anotaciones distintas, necesitamos un archivo final contenga, para cada token, las etiquetas asignadas por el gold estándar y el algoritmo (o el símbolo 'O' en caso de que no tenga etiqueta). Luego, al tener el archivo de anotación en formato intermedio, tanto el del gold estándar y el del algoritmo, ejecutamos otro script que combina ambos. Esto resulta en un archivo que contiene en la primera columna el token anotado, en la tercera el valor anotado por el gold estándar y en la cuarta la etiqueta asignada por el algoritmo. Luego de esto, podemos proceder a realizar el cálculo del mach exacto y parcial.

Dado el ejemplo anterior, el archivo resultante sería:

Palabra	TAG	Etiqueta anotada gold estándar	Etiqueta anotada algoritmo
Fecha	TAG	O	O
:	TAG	O	O
24	TAG	B-Date	B-Date
de	TAG	I-Date	I-Date
abril	TAG	I-Date	I-Date
,	TAG	I-Date	O
2019	TAG	I-Date	B-Date

En este caso, se puede observar que el algoritmo anotó dos fechas: "24 de abril" (DT) y "2019" (DT), a diferencia del anotador que anotó "24 de abril, 2019" (DT).

Match exacto

Una vez que se obtuvo el archivo deseado, se procedió a realizar un match exacto de F1-score. Para esto, utilizamos el script en Perl *conlleval.pl*[31] de la *ConLL-2000 shared task* [56]. Este genera, para cada archivo de anotación, un resultado que contiene la *accuracy*, *precision*, *recall* y *F1-score*, tanto general del artículo como por entidad. Por ejemplo, en la figura 5.1 podemos ver un ejemplo de anotaciones realizadas por el gold estándar (5.1a) y por el algoritmo (5.1b).

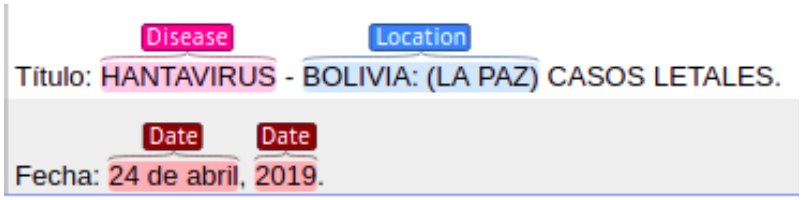
Luego de la ejecución del script, se obtienen los siguientes resultados:

```
processed 21 tokens with 4 phrases; found: 4 phrases; correct: 1.
accuracy:  71.43%; 1 4 4 precision:  25.00%; recall:  25.00%; FB1:  25.00
Date:  0  1  2 precision: 0.00%; recall: 0.00%; FB1: 0.00  2
Disease:  1  1  1 precision: 100.00%; recall: 100.00%; FB1: 100.00  1
Location:  0  2 1 precision: 0.00%; recall: 0.00%; FB1: 0.00  1
```

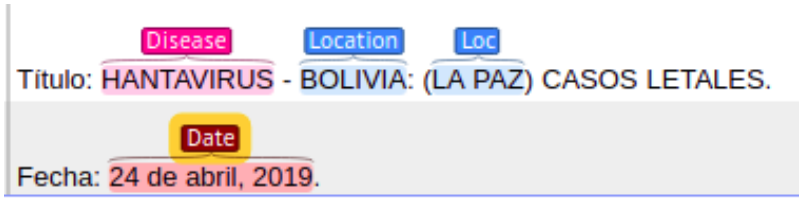
De la cual se puede observar lo siguiente:

	Precision	Recall	$F_{\beta=1}$
Date	0.00 %	0.00 %	0.00
Disease	100.00 %	100.00 %	100.00
Location	0.00 %	0.00 %	0.00
Total	25.00 %	25.00 %	25.00

[31] El conlleval script junto con la explicación y ejemplos puede encontrarse en `http://www.clips.uantwerpen.be/conll2000/chunking/output.html` [Accedido en Marzo 2021].



(a) Ejemplo de anotación manual.



(b) Ejemplo de anotación del algoritmo.

Fig. 5.1: Ejemplo de anotaciones manuales y del algoritmo.

De cada entidad nombrada, el primer valor numérico refiere a los valores *True Positives* (*TP*), el segundo a *True Positive* (*TP*) + *False Negative* (*FN*), y el tercero a los *True Positive* (*TP*) + False Positive (*FP*). En este caso se puede observar que dicho script utiliza un match exacto ya que para las entidades *Location* y *Date* se obtiene un resultado de 0 (es decir, que no hubo ningún tipo de match cuando en parte sí lo hubo).

Para obtener un resultado global por entidad, generamos un nuevo script que itera sobre todos los archivos con formato CoNLL de un directorio y obtiene la suma de todos los valores de los TP, los TP + FN y los TP + FP para cada entidad (por ejemplo, se obtienen la suma de todos los TP de todas las entidades de tipo disease sobre todos los archivos). Una vez obtenidos estos valores, se genera, para cada entidad, el valor de la precision, recall y el F1-score (cuyas fórmulas se pueden encontrar en la sección 3.5), y también se generan los mismos resultados pero sobre la unión de todas las entidades, de forma tal de evaluar los resultados por entidad y globales.

Match parcial

Para evitar perder información en los casos en los que el match no sea exacto, se procedió a implementar un match parcial aproximado. Este algoritmo está basado en la métrica de evaluación de MUC-3 [19], que toma en cuenta aquellos casos de anotaciones con la misma entidad, pero en los que el rango no coincida exactamente (los *tokens* de una de las entidades comenzaron antes o terminaron después), en cuyo caso se le asigna una proporción del valor total (0.5). Se utiliza una versión modificada de *precision* y *recall* (ver ecuaciones 5.1 y 5.2), teniendo en cuenta los criterios establecidos en la tabla 5.1 y se utilizan las siguientes fórmulas:

$$POS = COR + PAR + INC + MIS$$

$$ACT = COR + PAR + INC + SPU$$

Abrev.	**Cat.**	**Criterio**	**Correspondencia en matriz de confusión**
COR	correcto	La clave y la resp son iguales	TP
PAR	parcial	La clave y la resp tienen una coincidencia parcial	Fracción de TP
INC	incorrecto	La resp y la clave son distintas	Equivale a dos valores incorrectos: un FP ya que la resp no está en la clave, y un FN ya que la clave no es vacía y no estaba la respuesta buscada
SPU	espurio	La clave es vacía pero la resp no	FP
MIS	ausente	La resp es vacía pero la clave no	FN
NON	evasivo	La resp y la clave son vacías	TN

Tab. 5.1: Criterios de evaluación utilizados en MUC para el match parcial, siendo *clave* es el valor asignado por el gold estándar y *resp* el valor asignado por el algoritmo.

$$precision = \frac{COR + 0{,}5 * PAR}{ACT} \tag{5.1}$$

$$recall = \frac{COR + 0{,}5 * PAR}{POS} \tag{5.2}$$

5.4. Método basado en reglas

Para el método basado en reglas, se separó un subconjunto de 30 artículos que fueron manualmente analizados para el armado de reglas. La división se realizó de manera equitativa por enfermedad, teniendo en cuenta el porcentaje que se contenía de cada una de ellas, para así evitar que se perdiese información sobre alguna.

Se analizaron distintas herramientas de procesamiento del lenguaje natural para detectar entidades nombradas, cuyo detalle se puede encontrar en la sección 8.4.2, decidiendo finalmente utilizar *FreeLing*. Con esta, generamos heurísticas para detectar ubicaciones geográficas, fechas, cantidad de casos, textos condicionales y textos en pasado. Para el resto de las entidades (disease, origin, host, transmission form, negation y uncertainty) se utilizaron expresiones regulares, las cuales fueron manualmente generadas en base a los artículos separados.

5.4.1. Armado de las reglas

A continuación presentamos las reglas utilizadas para detectar cada tipo de entidad nombrada.

Cantidad de casos (NoC)

La cantidad de casos es una referencia a la cantidad de casos de una enfermedad determinada.

Esta se encuentra representada por un número. Utilizamos la librería de *Freeling* para realizar *PoS-tagging* sobre los artículos periodísticos. Una vez que tenemos todos los *tokens* con sus respectivas etiquetas, consideramos como potenciales candidatos aquellas con la etiqueta de número (Z) o de adjetivo ordinal ($A0$). Para evitar problemas de género y número, se trabajó con su respectivo lema.

No todos los términos con la etiqueta Z son útiles ya que se encontraban casos de *tokens* no numéricos etiquetados como tales (por ejemplo, *D8*), por lo que se realizó un filtrado posterior para obtener solo aquellos candidatos que contienen números, puntos y comas (utilizando la expresión regular $[0-9.,]^+$). En cuanto a los adjetivos ordinales, no todos eran de interés (por ejemplo *último*), por lo que se armó una lista con los mismos para su descarte.

Otro tipo de heurística que se aplicó fue basado en las palabras cercanas al candidato:

- **Referencias a puntos del tiempo:** Para evitar detectar como NoC referencias a puntos del tiempo se descartaron los siguientes candidatos:
  - Aquellos en que la palabra anterior o posterior se encuentra entre las siguientes: *día*, *mes*, *año*, *semana*.
  - Aquellos precedidos por las palabras: *en* y *desde* (por ejemplo, *Desde 2019* o *En 2019*).
- **Referencias a porcentajes:** Como únicamente nos interesan valores puntuales de casos y no proporciones, todas aquellas referencias a porcentajes serán descartadas. Para esto, nos fijamos si la/s palabra/s siguiente es/son alguna de las siguientes: *%*, *porciento*, *por 100*[32]. En dicho caso, se descarta.
- **Otros casos:** Otros casos que se descartaron fueron aquellos en los que la palabra anterior o posterior era alguna de las siguientes: *vez*, *grado*, *º*, *primero*.

Una vez obtenida nuestra lista de potenciales candidatos, no todos los valores se refieren a casos. Por esto, se generó una lista de posibles palabras (lemas) que pueden indicar que en una oración se esté haciendo referencia a cantidad de casos: “caso”, “infectar”, “infectado”, “afectado”, “muerte”, “muerto”, “morir”, “fallecer”, “fallecido”, “defunción”, “deceso”, “contagiar”, “contagiado”, “contagio”, “paciente”, “afectar” y “brote”. Luego, para cada posible candidato, se evalúa si se encuentra a una distancia máxima de 7[33] términos de alguna de estas palabras de interés. En caso de que así sea, se la anota como entidad, caso contrario se la descarta.

Fechas (DT)

La entidad *Fecha* referencia a términos de fechas.

Existen dos formas a través de las cuales las detectamos:

- Analizando el PoS-tagging de *Freeling*, quedándonos con aquellos tokens cuya etiqueta sea de tipo fecha (W).

[32] En algunos artículos se encontraba escrito *X por ciento* y, al realizar el PoS-tagging, la palabra *ciento* resultaba en 100. Por lo tanto, el término se transformaba en *por 100*.

[33] Este valor fue determinado en base a los artículos analizados

- Utilizando una expresión regular para detectar rangos de fechas (por ejemplo, *2018-2020*). La expresión utilizada fue:

```
^(19|20)\d\d-(19|20)?\d\d$)|(^(19|20)\d\d$
```

Una vez que se obtienen los candidatos, se analizó si el término se anotaba como *Fecha* o como *Past*[34]. Para esto, se comparó la fecha hallada contra la del título, considerándola anterior en los siguientes casos:

- Ambas poseen la información del año. En caso de que sean iguales, se desempata por mes.
- En el caso en el que se compare por mes, se lo anota como entidad de tipo pasado si es del mes anterior o anterior.
- No se posee información del año o mes, en cuyo caso se anota como entidad de tipo fecha.
- Posee la palabra *pasado* en ella (por ejemplo, *pasado viernes*), por lo cual se la considera anterior.

Generación de gazetteers

Un *gazetteer* es un diccionario o directorio geográfico que contiene información sobre la composición geográfica, estadísticas sociales y características físicas de un país, región o continente. Provee información como las dimensiones de ubicaciones geográficas, su población, los idiomas oficiales, entre otras cosas.

Los *gazetteers* serán utilizados en conjunción con las librerías de NLP para detectar ubicaciones dentro de los textos periodísticos. La herramienta que se utilizó para recolectar dicha información fue *geonames*.[35]

Se utilizaron las APIs públicas para obtener todos los continentes y países del mundo, todas las regiones de cada país y todas las ciudades que existen cuya población sea mayor a 5.000 habitantes para países de Latinoamérica y mayores a 15.000 para el resto. Para detectar si un país es latinoamericano, se filtra si es de Sudamérica o Norteamérica[36], y si su idioma oficial es español o portugués.

Además de obtener los nombres de cada tipo de ubicación para generar los *gazetteers*, se almacenó otro tipo de información:

- De cada ubicación geográfica: cantidad de habitantes e id.
- De cada país: nombre, continente en el que se encuentra, idiomas oficiales y lista de regiones y ciudades que lo componen.

Esta información es necesaria para, entre otras razones que serán explicadas posteriormente, decidir si una nota será anotada o no (ya que solo decidimos anotar artículos de países latinoamericanos) y también para desambiguar en los casos en los que se poseen ubicaciones con igual nombre. Luego, se procede con la integración en las librerías para poder utilizarlas (para más información, ver sección 8.4.3).

[34] Siguiendo el criterio de anotación, se deben anotar como pasado aquellas fechas anteriores a la establecida en el artículo.

[35] `https://www.geonames.org/` [Accedido en Junio 2021].

[36] Los países de Centroamérica son considerados como parte de América del Norte por la herramienta.

Location (LOC)

Las entidades *Location* son todas aquellas referencias a ubicaciones geográficas.

Detectar ubicaciones geográficas es de interés para saber en dónde se generó cierta enfermedad. Para poder detectarlas, se desarrollaron una serie de heurísticas.

Inicialmente, armamos los correspondientes *gazetteers* con todos los países, continentes, regiones y ciudades (de más de 5.000 habitantes para países de Latinoamérica[37] y más de 15.000 para el resto del mundo). Una vez integrados los mismos en *Freeling*, identificamos los términos que referencian a ubicaciones con la etiqueta *NP00G00*.

Luego, aplicamos las siguientes heurísticas sobre las ubicaciones detectadas:

- *Evaluar la ubicación mencionada en el título del artículo:* Como solo nos interesa anotar aquellos artículos que traten enfermedades que afecten a países de Latinoamérica, utilizamos la ubicación del título (ver sección 8.4.4 para información extra sobre detalles de implementación) para determinar la ubicación del artículo. Dada la ubicación, si no pertenece a latinoamerica, el artículo entonces no es anotado por el algoritmo.

- *Descartar ubicaciones en el cuerpo del artículo:* retuvimos como válidas aquellas ubicaciones, dentro del cuerpo del artículo, que pertenecen al lugar geográfico que se referencia en el título del artículo. Existen distintos casos para esto:

  - **Continente en el título:** en caso en el que el continente mencionado sea alguno perteneciente a *América* (ya sea *América*, *América del sur*, *América Central* o *América del Norte*), obtuvimos cada país, ciudad y región dentro del mismo que, a su vez, pertenezca a Latinoamérica. Por ejemplo, si el continente es *América del Norte*, se obtienen todos los países, ciudades y regiones marcadas como parte de *Norteamérica* por *geonames* y se filtra, posteriormente, si pertenece a Latinoamérica (ej. *México* o cualquier país de América Central, más sus ciudades y regiones). Luego, una ubicación se considera válida si pertenece a alguno de la lista generada.
  - **País en el título:** por cada ubicación geográfica dentro del cuerpo del artículo, filtramos si es una región o ciudad perteneciente al país mencionado en el título.
  - **Ciudad o una región en el título:** obtuvimos el país correspondiente a la ciudad o región mencionada, y filtramos por todas las ubicaciones que se encuentren dentro de dicho país.

- *Casos de ambigüedad*: en el caso en el que existe más de una ubicación geográfica con el mismo nombre en los *gazetteers*, tanto para ubicaciones dentro del cuerpo del artículo como del título, nos quedamos con aquellos con más habitantes. Por ejemplo, si se menciona *El Salvador* en el título, puede hacer referencia al país o a una ciudad de México. En dicho caso, nos quedamos con el país.

- Dado que en los artículos se hacía referencia a muchas ubicaciones geográficas, solo retuvimos aquellas ubicaciones que co-ocurren en una misma oración en la cual se hace referencia a cantidad de casos. Esto fue aplicado solamente en el cuerpo del artículo, ya que en los títulos no suele haber menciones a cantidad de casos/muertes.

[37] Consideramos Latinoamérica aquellos países de América cuyo idioma oficial es el español o el portugués. Esta información la obtuvimos de geonames.

Past terms (PT)

Las entidades de tipo *Past terms* hacen referencia a hechos que ocurrieron en el pasado.

Para poder detectar frases escritas en tiempo pasado, además de las expresiones regulares, se utilizó el PoS-tagging de *Freeling*. Existen cuatro tiempos verbales para hablar en pasado:

- **Pretérito imperfecto:** Nos quedamos con aquellas etiquetas del modo indicativo del tiempo imperfecto (pretérito imperfecto): V M I I (verbo principal indicativo imperfecto). Por ejemplo, la primera persona del singular sería: *'podía'* o *'detectaba'*.
- **Pretérito indefinido o pretérito perfecto simple:** Nos quedamos con aquellas etiquetas del modo indicativo en tiempo pasado: V M I S (verbo principal indicativo pasado). Por ejemplo, la primera persona del singular sería: *'pude'* o *'detecté'*.
- **Pretérito perfecto o pretérito perfecto compuesto:** se deberá conjugar el verbo auxiliar *haber* en presente y el verbo principal en participio. De esta forma, hay que tomar las siguientes etiquetas para cualquier dupla de palabras: V A I P (verbo auxiliar indicativo presente) + V M P (verbo principal participio). Además, se debe verificar que el primer verbo sea *haber*. Por ejemplo, la primera persona del singular sería: *'he podido'* o *'he detectado'*.
- **Pretérito pluscuamperfecto:** se debe conjugar el verbo auxiliar *haber* en imperfecto y el verbo principal en participio. De esta forma, hay que tomar las siguientes etiquetas para cualquier dupla de palabras: V A I I (verbo auxiliar indicativo imperfecto) + V M P (verbo principal participio). Además, se debe verificar que el primer verbo sea *haber*. Por ejemplo, la primera persona del singular sería: *'había podido'* o *'había detectado'*.

Luego de saber cómo obtener las expresiones que pueden indicar pasado, basta con tomar el texto original y quedarse solamente con el *PoS-tag* de cada término. Es decir, que dado el texto:

Se detectaron 15 casos

Nos quedamos con lo siguiente:

P00CN00 VMIS3P0 Z NCMP000

Luego, utilizamos las expresiones regulares de tiempos verbales generadas para detectar si una o dos etiquetas (dependiendo el caso) se encuentran escritas en pasado (en el ejemplo, *detectaron* daría positivo para pretérito indefinido). Para el caso del pretérito perfecto y pluscuamperfecto, se debe analizar, además, que el primer verbo de la dupla sea *haber*.

Conditional (COND)

Las entidades de tipo *Conditional* son términos que corresponden a un hecho que podría ocurrir en el futuro.

Para poder detectar frases escritas en tiempo condicional también se utilizó el PoS-tagging de *Freeling*. Existen dos tipos de formas de hablar en tiempo condicional del indicativo:

- **Condicional simple:** nos quedamos con aquellas etiquetas de verbos en condicional simple: V M I C (verbo principal indicativo condicional). Por ejemplo, la primera persona del singular sería: *'podría'* o *'detectaría'*.
- **Condicional compuesto:** se conjuga el verbo auxiliar *haber* en condicional simple y el verbo principal en participio. De esta forma, debemos obtener las etiquetas: V A I C (verbo auxiliar indicativo condicional) + V M P (verbo principal participio). Además, se debe verificar que el primer verbo sea *haber*. Por ejemplo, la primera persona del singular sería: *habría podido* o *'habría detectado'*.

El procedimiento para obtener aquellos términos escritos en tiempo indicativo condicional es el mismo al del pasado, con la diferencia de las expresiones regulares utilizadas.

Otras entidades nombradas

Para las restantes entidades nombradas (disease, origin, host, transmission form, negation y uncertainty), se utilizaron las expresiones regulares que fueron generadas a través del análisis sobre el conjunto de datos separados. Para evitar problemas de género y número, se lematiza cada token de la oración y en la expresión regular solo se incluyen los lemas de las palabras. Luego, se obtiene la palabra original en el texto para anotar la entidad correspondiente. Las expresiones regulares utilizadas para cada una de las entidades pueden encontrarse en la tabla 5.2 y en la tabla 5.3 pueden observarse expresiones regulares auxiliares.

Entidad	**Expresión regular**
Disease (DS)	(zika|chagas|guillain[-_]barr[ée]|microcefalia|sarampi[ó|o]n| hantavirus|virus de el hanta|hanta|chikungunya|chikunguya|dengue|chikungunya)| (enfermedad|enf.|enf)_de_chagas|dengue(hemorr[a|á]gico((de)*tipo (2|dos))*)*
Origin (OR)	(baya|jugo) de {acai}|(consumo (de)*)*{acai} contaminar|insecto|entero(_)*virus|no estar vacunar| stercoralis|estrongiloidiasis|strongyloides stercoralis|(infección por)*{cruzi}|rat[o|ó](colilargo)| (rata|roedor|chipo)|(mosquito|vector|zancudo|vinchuca)({aedes})*| zika|(presencia de el)*vector({aedes})*|{aedes}
Host (Hst)	bebé|latinoamericano|niño|comunidad judío ortodoxo|mujer embarazar|embarazar| recién nacer|anciano|gestante|asistente de vuelo|auxiliar de vuelo|viajero|no[-]vacunar| trabajador rural|(donante|donador) de sangre|migrante|turista
Transmission form (TF)	contaminación|transmisión vectorial|ingresar por aparato respitatorio|transmisión sexual| contagio (intrapersonal|interpersonal)|(contagio (por)*)*(vía|contacto) sexual|picadura|picar| brotar oral|transmisi[o|ó]n oral|transfusión de sangre|transfusión sanguíneo|(brote|vía) oral
Negation (NT)	fin|no|tampoco|descartar|negativo| resultado negativo|resultar negativo|error (de)*diagnóstico|ausencia
Uncertainty (UT)	poder tratar|sospechar|probable|en observación|poder causar| riesgo de que suceder|bajo estudio|posible|probabilidad de error|(caso)*sospechoso| poder (tratar|ser)|probablemente|a confirmar|poder corresponder|investigar
Past (PT)	desde que comenzar|({days}|{months}|año|mes|semana) pasar| entre {months} y {months}durante el (mes|año|semana)(de {months})*(de el (año)*({year_regex}| pasar))*|(a_principios_de|a principios de|a comienzo de) {months}| (pasar|desde el|desde) ({days})*({days_regex}) de {months}(de {year_regex})*| (a_principios_de|a principios de|a comienzo de) el mes de {months}|({days} de el)*semana pasar| pasar({days}|{months}|{year_regex})|ocurrir hacer|desde hacer (mes|año)|desde {months} hasta (fin de)*{months}|(desde {months})*hasta ((mediado|mediar) de)*{months}|{months} ((de el)| de este)*(año)*{year_regex}|en el que ir de el año|desde {months} de el (año)*({year_regex}| pasar)*|durante este semana|hasta el (fecha|momento)|semana antes| semanas más tarde|en (el año)*({year_regex})
Conditional (COND)	si

Tab. 5.2: Expresiones regulares generadas manualmente para detectar algunas entidades nombradas.

Nombre auxiliar	Expresión regular
acai	(açai|açaí|acai|acaí)
cruzi	((trypanosoma|t|ṫ |t_|ṫ_)(|_ |_)*(cruzi|cruz|cruz_i|cruz i))
aedes	(de el género)*((_ |_)*aedes((|_|)aegypti)*(_)*| (_ |_)*(a|ȧ_|a|a_)( )*aegypti(_)*)
days	(lunes|martes|miércoles|jueves|viernes|sábado|domingo)
months	(enero|febrero|marzo|abril|mayo|junio|julio|agosto| septiembre|octubre|noviembre|diciembre)
days_regex	(0?[1-9]|[12][0-9]|3[01])
year_regex	(19|20)?\d\d

Tab. 5.3: Expresiones regulares auxiliares utilizadas en las expresiones regulares de detección de entidades nombradas.

Para el caso de las entidades *Disease*, *Negation*, *Uncertainty*, *Past* y *Conditional*, estas solo fueron reconocidas como entidades nombradas si co-ocurren en la misma oración y a una distancia máxima de otra entidad con la cual pueden estar relacionadas (considerando las mismas relaciones que las del esquema de anotación).

Un caso particular que hubo que manejar fue el caso en el que el *zika* causa la *microcefalia*. Para esto, se agregó la enfermedad *zika* dentro de los términos de causas y, en caso de que exista, se corroboró que en la misma oración exista la enfermedad *microcefalia*, a una distancia máxima establecida. En caso contrario, se elimina dicha entidad de la lista de causas detectadas para el artículo correspondiente.

5.4.2. Generación de archivos anotados

Una vez obtenidas todas las entidades nombradas, se procedió a generar el formato de archivo para poder evaluar las entidades anotadas y realizar las comparaciones contra las anotaciones manuales. Para realizar esto, se itera sobre cada entidad encontrada y se agrega la siguiente línea al archivo:

$$T_i \ \textit{entity\_name} \ \textit{start-offset} \ \textit{end-offset} \ \textit{value}$$

En donde:

- T_i**:** id de entidad.
- **entity_name:** nombre de la entidad (por ejemplo, *Disease*).
- **start-offset:** índice del primer caracter de la entidad dentro del archivo de texto.
- **end-offset:** índice del último caracter de la entidad dentro del archivo de texto.
- **value:** El texto de la entidad anotada, que se encuentra entre el *start-offset* y el *end-offset*.

Por ejemplo, supongamos que tenemos el texto:

```
Título: ZIKA - Puerto Rico: DECLARACIÓN DE FIN DE EPIDEMIA,  CONTROVERSIAS.
Fecha: 02 de Julio del 2017.
```

Luego, tendríamos las siguientes entidades en el archivo:

```
T1 Disease 8 12 ZIKA
T2 Location 15 26 Puerto Rico
T3 Date 82 102 02 de Julio del 2017
```

5.4.3. Resultados

En esta sección presentamos una serie de tablas con los resultados del algoritmo basado en reglas, tanto sobre los títulos como los artículos enteros, sobre el total del corpus anotado. Inicialmente, se muestra la cantidad total y diferente de términos hallados por entidad nombrada. Luego, algunos ejemplos de términos frecuentes para ciertas entidades. Finalmente, los resultados de *precision*, *recall* y *F1-score*. Se utilizó el match exacto y parcial de F1-score sobre cada entidad, realizando posteriormente el *micro-average* de los resultados.

En la tabla 5.4 podemos observar la cantidad de entidades de cada tipo encontradas por el algoritmo y por los anotadores. Esto se hizo para los títulos y para los artículos enteros. Para la columna de anotadores, no se consideraron las entidades de tipo *Abbreviation*, ya que el algoritmo no detecta la misma.

Entidad	Algoritmo		Anotadores	
	Total	**Diferentes**	**Total**	**Diferentes**
Date	551	489	491	462
Disease	564	10	560	35
Host	43	20	33	19
Location	583	65	573	189
Number of cases	95	15	10	5
Origin	61	9	47	21
Transmission form	22	5	30	15
Conditional	0	0	0	0
Negation	14	4	3	2
Past	99	43	1	1
Uncertainty	28	7	17	8
Total	**2060**	**667**	**1765**	**757**

Entidades en los títulos.

Entidad	Algoritmo		Anotadores	
	Total	**Diferentes**	**Total**	**Diferentes**
Date	813	578	613	550
Disease	3438	37	3053	229
Host	845	53	678	357
Location	1630	304	2611	884
Number of cases	3235	719	2176	758
Origin	963	71	778	270
Transmission form	126	23	293	168
Conditional	73	1	3	3
Negation	383	20	74	37
Past	728	284	504	328
Uncertainty	464	53	400	197
Total	**12698**	**2143**	**11183**	**3781**

Entidades en los artículos enteros.

Tab. 5.4: Tipos y cantidad de entidades y modificadores encontradas por el algoritmo de reglas y por los anotadores en los títulos y los artículos enteros.

Al observar la cantidad de entidades halladas por el algoritmo y por los anotadores, se puede observar una diferencia entre los valores diferentes en los títulos (32 % por el algoritmo y 43 % por los anotadores) contra los artículos enteros (17 % por el algoritmo y 34 % por los anotadores). Principalmente, en los artículos enteros, los anotadores hallaron el doble de valores distintos sobre los totales, comparados contra las del algoritmo. En cuanto a los títulos, solo fue un 30 % veces mayor, ya que estos no poseen tanto contenido. Esta diferencia se debe principalmente a que trabajamos con un conjunto fijo de expresiones regulares, lo cual se puede observar también en los valores sobre cada entidad particular.

En las tablas 5.5 y 5.6 se pueden observar ejemplos de los términos más frecuentes hallados para ciertas entidades por el algoritmo, junto a la cantidad de ocurrencias. Esto se hizo tanto para los títulos como para los artículos enteros.

Disease	**# ocur.**
zika	128
hantavirus	125
sarampión	121
dengue	93
chagas	41
microcefalia	37
guillain barré	11
guillain-barré	5
chikungunya	2
guillain	1

Host	**# ocur.**
niños	10
embarazadas	6
no-vacunados	4
embarazada	3
gestantes	3
viajero	2
donantes de sangre	2
recién nacidos	1
no vacunado	1
turista	1

Transmission form	**# ocur.**
transmisión oral	10
vía oral	6
transmisión vectorial	3
transmisión sexual	2
brote oral	1

Origin	**# ocur.**
zika	37
vectorial	7
roedores	4
vectores	3
vector	3
acai contaminado	2
enterovirus	2
mosquitos	2
mosquito	1

Tab. 5.5: Ejemplos de términos frecuentes hallados por el algoritmo de reglas en los títulos, junto a la cantidad de ocurrencias, para ciertas entidades.

Disease	**# ocur.**
zika	923
dengue	700
sarampión	605
hantavirus	410
microcefalia	301
chagas	156
chikungunya	99
guillain-barré	66
hanta	59
guillain barré	43
dengue hemorrágico	19
hantavirosis	16
enfermedad de chagas	9
hanta virus	6
chagas agudo	3
microcefalias	2
zikav	1
dengue hemorragico tipo 2	1
chikunguya	1
virus del hanta	1

Host	**# ocur.**
niños	245
bebés	117
bebé	69
mujeres embarazadas	57
niño	53
embarazadas	45
niña	41
recién nacidos	33
niñas	19
gestantes	19
turistas	17
viajeros	15
embarazada	12
recién nacido	10
latinoamericanos	9
turista	8
mujer embarazada	7
migrantes	6
niño costero	5
viajero	5

Transmission from	**# ocur.**
picadura	29
transmisión oral	15
transmisión sexual	15
vía oral	14
transmisión vectorial	11
picaduras	9
contaminación	7
picada	4
picado	3
vía sexual	3
brote oral	2
contagio interpersonal	2
contacto sexual	2
piquen	1
contagio intrapersonal	1
transfusiones de sangre	1
contagio por vía sexual	1
transfusión de sangre	1
picadas	1
transfusiones sanguíneas	1

Origin	**# ocur.**
zika	158
roedores	107
mosquito	106
mosquitos	89
vector	61
aedes aegypti	58
mosquito _aedes aegypti	57
vectorial	27
insecto	25
vectores	24
zancudo	21
vinchuca	20
insectos	17
vinchucas	16
trypanosoma cruzi	15
roedor	15
mosquito aedes aegypti	14
ratas	13
zancudos	9
zancudo _aedes aegypti	7

Tab. 5.6: Ejemplo de términos frecuentes hallados por el algoritmo de reglas en los artículos enteros, junto a la cantidad de ocurrencias, para ciertas entidades.

Si se comparan las entidades halladas en los títulos por los anotadores y el algoritmo (ver tablas 4.7 y 5.5), podemos observar que los términos frecuentes y sus ocurrencias son similares. Esto se puede sumarizar en el gráfico 5.2.

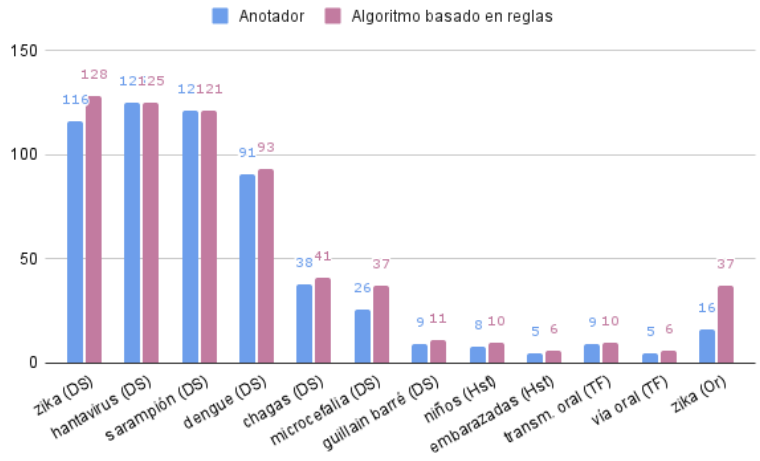


Fig. 5.2: Términos frecuentes hallados en los títulos entre los anotadores y el algoritmo basado en reglas.

En cuanto a los valores para los artículos enteros (ver tablas 4.8 y 5.6), sí se puede observar que el algoritmo encuentra mucha más instancias de un mismo valor contra las halladas por los anotadores. Ejemplos de estos valores pueden encontrarse en la figura 5.3.

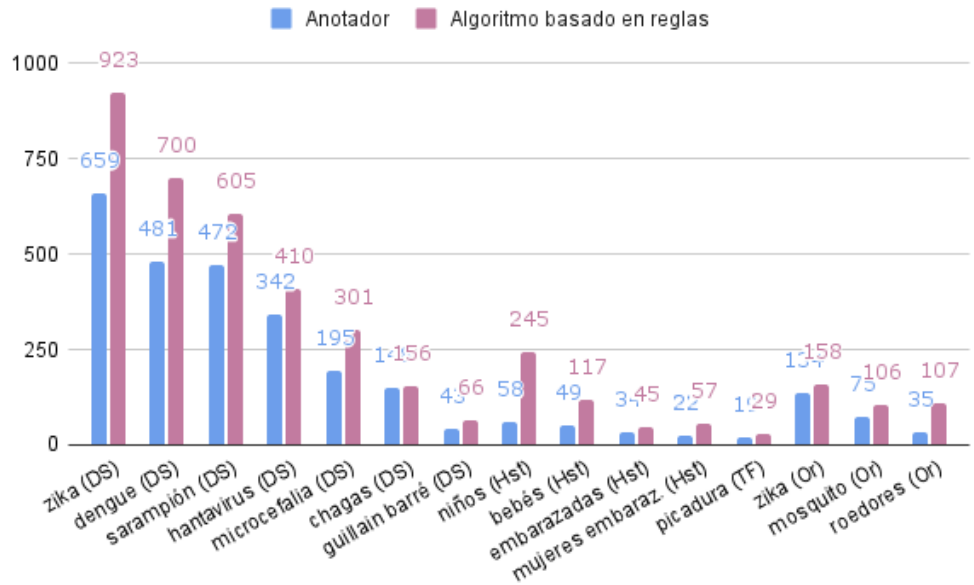


Fig. 5.3: Términos frecuentes hallados en los artículos enteros entre los anotadores y el algoritmo basado en reglas.

En la tabla 5.7 se pueden observar los resultados de *precision*, *recall* y *F1-score micro average* (match exacto y parcial) para los títulos y los artículos enteros entre las entidades halladas por los anotadores contra el algoritmo. Cabe resaltar que, debido a un error al generar la partición, 21 de los artículos del conjunto de test (sobre los 513 totales) se encontraban dentro del conjunto de artículos que se analizaron para la generación del presente método. Esto constituye un error, ya que no debería haber sido parte del conjunto de test. Por esta razón, los resultados aquí presentados pueden tener una variación respecto al valor real. Estimamos que esta es muy pequeña dado que el porcentaje del conjunto de test que se utilizó para el armado de reglas es del 4 %.

Entidad	**Match exacto**			**Match parcial**		
	P	**R**	**F1**	**P**	**R**	**F1**
Date	0.76	0.84	0.80	0.88	0.92	0.90
Disease	0.94	0.94	0.94	0.96	0.95	0.96
Host	0.52	0.67	0.59	0.61	0.70	0.65
Location	0.58	0.59	0.59	0.79	0.74	0.76
Number of cases	0.08	0.80	0.15	0.09	0.80	0.16
Origin	0.41	0.21	0.28	0.26	0.24	0.25
Transmission form	0.68	0.52	0.59	0.87	0.57	0.69
Negation	0.25	1.00	0.40	0.25	1.00	0.40
Past	0.00	0.00	0.00	0.00	0.00	0.00
Uncertainty	0.25	0.44	0.32	0.33	0.53	0.40
Micro-average	**0.70**	**0.76**	**0.73**	**0.81**	**0.84**	**0.82**

Precision, recall y *F1-score* en los títulos.

Entidad	**Match exacto**			**Match parcial**		
	P	**R**	**F1**	**P**	**R**	**F1**
Date	0.55	0.70	0.61	0.65	0.78	0.71
Disease	0.70	0.79	0.74	0.73	0.80	0.77
Host	0.26	0.33	0.29	0.32	0.40	0.35
Location	0.62	0.38	0.47	0.71	0.43	0.53
Number of cases	0.45	0.64	0.53	0.46	0.64	0.53
Origin	0.28	0.32	0.30	0.36	0.43	0.39
Transmission form	0.56	0.25	0.34	0.52	0.28	0.37
Conditional	0.00	0.00	0.00	0.00	0.00	0.00
Negation	0.08	0.41	0.14	0.10	0.49	0.17
Past	0.16	0.18	0.17	0.24	0.28	0.26
Uncertainty	0.21	0.25	0.23	0.30	0.35	0.32
Micro-average	**0.49**	**0.54**	**0.51**	**0.53**	**0.58**	**0.55**

Precision, recall y *F1-score* en los artículos enteros.

Tab. 5.7: Resultados del match exacto y parcial del método basado en reglas para los títulos y los artículos enteros.

5.4.4. Análisis de los resultados

En cuanto a los resultados de las entidades halladas por el algoritmo contra las anotadas manualmente, se puede observar lo siguiente:

- **Conditional:** Como se pudo observar en la sección 4.5, no hubo muchas anotaciones realizadas por los anotadores sobre esta entidad (solo 3). Inicialmente, se utilizó la regla de detección de términos escritos en forma condicional (ver sección 5.4.1), pero dado que encontraba muchas entidades (115 totales y 15 distintas) en relación a las anotadas y que eran todas erróneas, se decidió únicamente utilizar como expresión regular el término *'si'*. Con este enfoque, el algoritmo halló en total 73 entidades totales, como podemos ver en la tabla 5.4. Aun así, no hubo ningún tipo de coincidencias entre las halladas por el algoritmo y los anotadores. Para ver ejemplos de anotaciones frecuentes anotadas con el enfoque de detectar términos escritos en forma condicional, ver sección 8.3 del apéndice.
- **Host, origin, transmission form, negation, uncertainty, past y conditional:** En el caso de estas entidades que se reconocían utilizando expresiones regulares, podemos ver que los resultados no fueron tan óptimos en comparación con las demás entidades con reglas más complejas. Dentro de estas entidades, en los títulos se obtuvo un *F1* exacto mínimo de 0 % para la entidad *Past* y un máximo de 59 % para *transmission form* y *host* (siendo el *micro average* de 73 %), y en el artículo entero se obtuvo un mínimo de 0 % para *conditional* y un máximo de 34 % para *transmission form* (con un *micro average* de 51 %). Esto se debe a que la lista realizada no abarca todas las posibilidades que el anotador sí puede reconocer.
- **Disease:** dado que la lista de términos era mucho más acotada, sí podemos observar mejores resultados contra las demás entidades que estaban basadas en expresiones regulares, tanto para el artículo entero (74 % el *F1* exacto y 77 % el parcial) como los títulos (94 % el exacto y 96 % el parcial). Como se mencionó en la sección 4.5, los anotadores han anotado enfermedades no válidas (por ejemplo, *ceguera*, *glaucoma* y *rubéola*, entre otras), por lo que si se corrigiesen dichas anotaciones, nos encontraríamos con mejores resultados.
- **Date:** Para esta entidad se obtuvieron resultados altos. Sin embargo, se encontraron los siguientes problemas:
  - *Diferencias entre lo definido en el criterio de anotación y las anotaciones manuales:* el algoritmo anota correctamente aquellas fechas anteriores (ver sección 5.4.1) a las del título como *Past*, cuando el anotador las anota como *Date*. Esto provoca una disminición en los resultados tanto para la entidad *Past* como *Date*.
  - *Fechas aisladas:* los anotadores anotaron fechas aisladas (es decir, que no se encuentran relacionadas con alguna otra entidad), cuando el algoritmo sigue el criterio de anotación y solo las anota si co-ocurre con otra entidad.
  - *Normalización de las fechas:* como no existe un formato estándar de las fechas en los artículos, el anotador es capaz de reconocer fechas como parte de una sola entidad, cuando el algoritmo las separa en dos o más (por ejemplo, si tenemos la fecha *[25 de Marzo, 2000*, el anotador anota la entidad *"[25 de Marzo, 2000]"(Date)*. Por el contrario, el algoritmo utiliza *Freeling* para detectar fechas, el cual las reconoce por separado: *"[25 de Marzo]"(Date)* y *"[2000]"(Date)*). Si se observa el resultado del *F1 parcial*, se puede ver que provee mejores resultados que el exacto (de 80 % a 90 % en los títulos, y de 61 % a 71 % en los artículos enteros).

- **Number of cases:** en cuanto a los títulos, los anotadores encontraron 10 entidades de tipo *Number of cases* totales dentro del título, mientras que el algoritmo 95. Esto se debe principalmente a que en los títulos suelen mencionarse números, pero rara vez se encuentra información sobre a qué se refiere (por ejemplo, en el título *"SARAMPION - ARGENTINA: (BA) (02) AUMENTO DE CASOS AUTOCTONOS, VACUNACION INCOMPLETA.", el número* 02 *no se sabe si referencia a muertos, a casos u otra cosa*). Por esta razón, los anotadores no suelen anotar este tipo de entidades en el título, pero el algoritmo encuentra un número y una palabra de interés cercana (en el caso del ejemplo sería *"CASOS"*, pero otras comúnmente halladas en el título son *muerte* y *brotes*) y las anota como entidad. Si se observa el valor de *recall* (80 %), se puede ver que halló gran parte de las que debía. Sin embargo, según el valor de *precision* (8 %), pocos de los términos hallados eran correctos. En cuanto a los artículos enteros, el algoritmo tuvo un valor exacto y parcial de 64 % para *recall*, encontrándose por encima del *micro-average* exacto de 54 % y parcial de 58 %. En cuanto a *precision*, se puede observar que el algoritmo etiqueta más términos de los que debe, ya que el exacto fue de 45 % y el parcial de 46 %, encontrándose por debajo del *micro average*. Esto indica que hay que poner énfasis en mejorar la *precision*.

- **Location:** los valores de *precision* en los artículos enteros fueron elevados (por encima del *micro average*), indicando que gran parte de los términos detectados son correctos. Sin embargo, el valor de *recall* fue bajo, demostrando no ser capaz de detectar gran parte de los que debe. En cuanto a los títulos, ambos valores fueron similares, encontrándose por debajo del *micro average*. Algunos de los problemas fueron:

  - En gran parte de los títulos, se exhibe el país junto con la abreviatura de una ciudad o región (por ejemplo, *"Brasil: (SP)"*). En este caso, el anotador es capaz de relacionar *SP* con la abreviatura de una ubicación geográfica (por ejemplo, *Sao Paulo*), pero el algoritmo no ya que no poseemos una lista con las abreviaturas de las ubicaciones geográficas. Además, estas no suelen seguir un estándar, sino que varían según el artículo.
  - Similar al caso anterior, en los títulos se indica entre paréntesis el nombre de una ciudad o región de un país, solo que sin abreviar (por ejemplo, *Argentina (Chubut)*). Al obtener las ubicaciones geográficas con *Freeling*, obtenemos dos entidades de tipo ubicación: (*Argentina* y *Chubut*). Sin embargo, según el criterio de anotación, el anotador debe anotar todo como una sola entidad (*Argentina (Chubut)*). Para este caso, en los títulos, se puede observar una mejoría en el F1 parcial ya que cambia de 59 % a 76 %.
  - El algoritmo solo anota, en el cuerpo de los artículos, aquellas ubicaciones que co-ocurren en una misma oración con una entidad de tipo *Number of cases*, lo que genera que muchas ubicaciones no hayan sido tenidas en cuenta. Esto se ve reflejado en el bajo valor de *recall*.

- **Past:** inicialmente se probó utilizando el método para detectar términos escritos en pasado (ver sección 5.4.1), con un total de 105 entidades totales y 45 distintas en los títulos, y 3068 totales y 914 distintas en los artículos enteros. Dado que para los títulos se obtenía un *F1* (exacto y parcial) de 0 %, y para los artículos enteros un *F1* exacto de 5 % y un parcial de 9 %, se decidió utilizar únicamente las expresiones

regulares. Con este método, se obtuvieron mejores resultados en el artículo entero (17 % el match exacto y 26 % el parcial). Para ver ejemplos de anotaciones frecuentes anotadas con el enfoque de detectar términos escritos en tiempo pasado, ver sección 8.3 del apéndice.

- **Origin y transmission form:** los resultados para estas entidades dieron por debajo de los *micro-average* debido a inconsistencias en las anotaciones manuales. Si bien se intentó establecer ejemplos en los criterios de anotación para evitar confusiones, como se puede observar en la tabla de las entidades frecuentes anotadas por los anotadores 4.8 en la sección 4.5, hubo problemas para su comprensión ya que se han anotado como formas de transmisión los términos: *'aedes aegypti'*, *'mosquito/s'*, *'raton/es'* y las variaciones con las que se escriben cada uno de ellos. En cuanto a términos de tipo *Origin* podemos encontrar: *'consumo'* y *'picadura/s*, entre otros. En cambio, el algoritmo tiene una lista fija establecida sobre los términos de las expresiones regulares que son de tipo *transmission form* y *origin*, por lo que no van a haber términos relacionados a *mosquitos* o *ratones* como formas de transmisión, ni *consumo* o *picaduras* como causa. Si se observan los resultados en los artículos enteros, más de la mitad de las entidades halladas de tipo *Transmission form* son correctas (con una *precision* exacta de 56 % en los artículos enteros), mientras que falla en hallar todas las que debe, lo cual se refleja en el valor de *recall*. En cuanto a *Origin*, ambos valores son bajos (*precision* exacta de 28 % y *recall* de 32 %).

Para concluir, se pudo observar que para la mayoría de las entidades para las cuales se utilizaron únicamente expresiones regulares para su detección (*Host*, *Origin*, *Transmission form*, *Conditional*, *Negation*, *Past* y *Uncertainty*), los resultados estuvieron por debajo de los *micro averages*. El único caso que resultó con resultados elevados fue la entidad *Disease*, lo cual se debe a que el conjunto de entidades de interés es reducido y ya se encuentra establecido. En cambio, aquellas para las cuales se combinó *FreeLing* con reglas más complejas, se observaron resultados por encima o cercanos al *micro average* (*Date*, *Location* y *Number of cases*). Alguna de las mejoras que podrían aplicarse son:

- Analizar un mayor número de artículos para poder expandir las expresiones regulares utilizadas y mejorar las reglas.
- Aplicar normalizaciones sobre los artículos para que *FreeLing* sea capaz de detectar las entidades de tipo *Date* y *Location* como parte de una sola entidad en vez de múltiples. Otra opción podría ser detectar entidades de tipo *Date* y *Location* que se encuentran contiguas y anotarlas como parte de una misma entidad, si corresponde.
- No anotar las entidades de tipo *Number of cases* que se encuentran en los títulos o evaluar alguna regla más compleja para su detección allí.

5.5. Método basado en redes neuronales profundas

Para entrenar un algoritmo supervisado de redes neuronales profundas que realice anotaciones automáticas, debíamos contar con un corpus anotado. Nos basamos en el trabajo realizado por Akbik, Blythe y Vollgraf [2], utilizando su librería *Flair*[38]. Este es

[38] `https://github.com/flairNLP/flair`

un framework que permite realizar distintos tipos de procesamiento del lenguaje natural, como *NER* y *part-of-speech tagging*, y tiene soporte para una gran cantidad de idiomas, como el español. Al estar construido sobre *pytorch*[39], permite fácilmente acceder a todas sus funcionalidades para entrenar modelos.

Existen diversas tareas dentro del procesamiento del lenguaje natural, siendo una de ellas el NER, que se pueden formular como un problema de *etiquetado de secuencias*. En estas, el texto es tratado como una secuencia de palabras, a las cuales se les debe asignar una etiqueta (dentro de una lista de categorías predefinidas). Un recurso muy importante para realizar esta tarea son los *word embeddings* (ver sección 3.2). En el trabajo de Akbik, Blythe y Vollgraf, se propone un *contextual string embeddings* que tienen la habilidad de: (1) ser entrenados en grandes corpora no etiquetadas, (2) capturar el significado de la palabra según el contexto (por lo que habrá distintos *embeddings* según el contexto) y (3) modelar las palabras y contexto como una secuencia de caracteres (lo cual permite manejar palabras raras o que están mal escritas, y también modelar estructuras de subpalabras, como prefijos y terminaciones). Utilizan un modelo de lenguaje que le permite modelarlo como una distribución de secuencias de caracteres en vez de palabras. Luego, para su módulo etiquetador de secuencias de NER, utilizan una arquitectura BiLSTM (ver sección 3.10), combinado posteriormente con una capa que utiliza CRF (ver sección 3.2).

En la sección 5.5.1, explicamos de manera más detallada cómo funciona la librería y cómo fue utilizada para entrenar nuestro corpus para realizar NER. Luego, en la sección 5.5.4, presentamos los resultados preliminares que obtuvimos entrenando sobre nuestro corpus anotado y, finalmente, en la sección 5.5.5, realizamos un análisis sobre los resultados obtenidos.

5.5.1. Explicación del método

Para generar los *embeddings* a nivel de palabra, el enfoque procesa cada oración como una secuencia de caracteres utilizando un modelo de lenguaje a nivel de caracter. El objetivo de este modelo es predecir el siguiente caracter dado los caracteres previos. Para esto, se calcula la probabilidad de que aparezca el caracter x_t dada la aparición de los caracteres anteriores $x_0, ..., x_{t-1}$: $P(x_t|x_0, ..., x_{t-1})$. La probabilidad de una oración entera se puede descomponer como el producto de la probabilidad de los caracteres condicionados a los caracteres anteriores:

$$P(x_{0:T}) = \prod_{t=0}^{T} P(x_t|x_{0:t-1}) \tag{5.3}$$

donde $x_0, ..., x_T$ son caracteres, $x_{0:t} =: (x_0, ..., x_t)$ y T es el índice del último caracter de la oración.

Se utilizan las capas ocultas de una red neuronal recurrente *forward-backward* para crear los *word embeddings* contextualizados. Luego, para cada palabra, se utiliza el *fLM* (*forward language model*) para extraer el estado de la capa oculta luego del último caracter de la palabra (obteniendo la información sintáctica y semántica de la oración desde el inicio hasta la palabra seleccionada, incluyéndola) y del *bLM* (*backward language model*) se extrae el estado hasta antes del primer caracter de la palabra (conteniendo la información

[39] Librería gratuita y open-source de inteligencia artificial en Python (lenguaje de programación) que se utiliza, entre otras cosas, para el procesamiento del lenguaje natural.

desde el final de la oración hasta el inicio de la palabra seleccionada). Ambos estados se concatenan para formar el *embedding* de la palabra final, teniendo la información semántica y sintáctica de la palabra desde el comienzo de la oración hasta el final y viceversa, como también su contexto.

Formalmente, supongamos que tenemos palabras cuyos caracteres comienzan en los índices $t_0, t_1, ..., t_n$. Luego, el *word embedding* contextualizado para cada una de ellas se define como:

$$w_i^{CharLM} := \left[\frac{h^f_{t_{i+1}-1}}{h^b_{t_i-1}} \right] \tag{5.4}$$

donde *CharLM* es el modelo de lenguaje a nivel de caracter, el superíndice f indica que es la capa *forward* y el superíndice b la capa *backward*, y h_k representa el pasado de la secuencia del caracter k.

En la figura 5.4, se puede observar un ejemplo de un word *embedding* dentro de una oración.

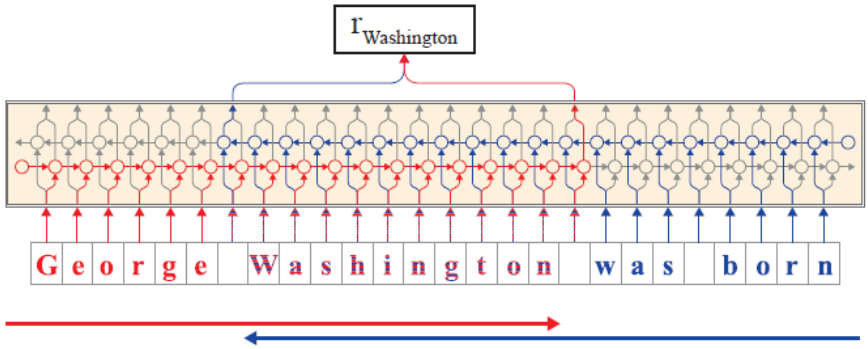


Fig. 5.4: Ejemplo de extracción de un *embedding* para la palabra "Washington" junto a su contexto en una oración. El modelo forward está marcado en rojo y el backward en azul. Gráfico tomado de Akbik, Konomi y Melnikov [3].

Una vez obtenidos los *embeddings* de cada una de las palabras de las oraciones, estos son pasados a un módulo etiquetador de secuencias BiLSTM-CRF que permitirá realizar NER (ver en la figura 5.5 un gráfico que describe a alto nivel la propuesta final). La librería ofrece la posibilidad de utilizar *stacked embeddings*, que resultan de combinar distintos *embeddings* concatenando cada vector de *embeddings* en un vector final. Por ejemplo, si se combinase el *word embedding* propuesto por los autores junto con un *GloVe embedding* [49], el *word embedding* contextualizado para la palabra cuyo primer caracter se encuentra en la posición i estaría dado por:

$$w_i = \left[\frac{w_i^{CharLM}}{w_i^{GloVe}} \right] \tag{5.5}$$

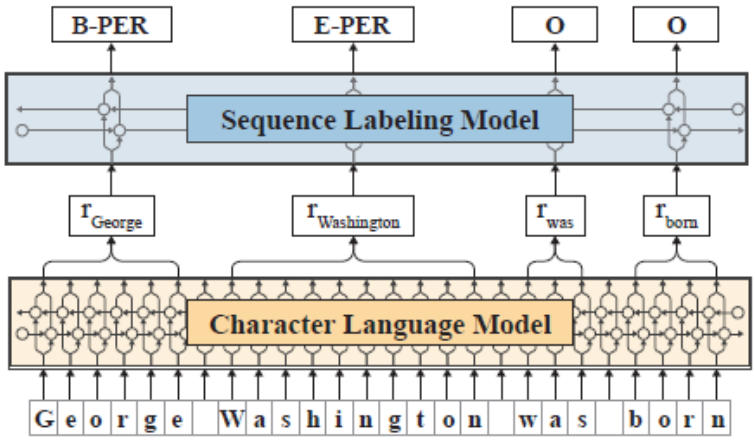


Fig. 5.5: Descripción a alto nivel del enfoque propuesto por Flair. La oración es ingresada como una secuencia de caracteres dentro de un modelo de lenguaje bidireccional a nivel de caracteres. De este LM, se obtienen los *word embeddings* contextualizados de cada palabra, que son pasados al etiquetador de secuencias BiLSTM-CRF, que realiza NER en el caso del ejemplo. El formato de etiquetado del gráfico es el *BIOES*, en donde *B-PER* indica el inicio (*begin*) de la entidad persona, *E-PER* indica el fin (*end*) de la entidad y *O* indica que el *token* no pertenece a ninguna entidad. Gráfico tomado de Akbik, Konomi y Melnikov [3].

5.5.2. Detalles de implementación

La propuesta descripta anteriormente cuenta con una librería en python[40] que utilizamos para realizar NER. Esta provee la implementación de distintos tipos de *embeddings* que se encuentran disponibles para su uso y la posibilidad de utilizar *embeddings* propios.

Para utilizar el corpus anotado generado en este trabajo, se dividieron los artículos anotados en un 80 % de *training* (410) y un 20 % de *test* (103) de manera estratificada. Esta forma preserva el porcentaje sobre cada clase distinta siendo, en nuestro caso, cada clase una enfermedad, lo cual se realiza para no perder información sobre ninguna de ellas. De esta forma, el conjunto de *test*, está formado por un 20 % del total de enfermedades de tipo Chagas, 20 % de sarampión, etc. En la siguiente tabla se puede observar la división final del conjunto de datos de entrenamiento y test (microcefalia no es tenida en cuenta ya que los artículos se solapan con los de zika):

Set	**Chagas**	**Dengue**	**Guillain**	**Hanta**	**Sarampión**	**Zika**	**Total**
Train	33	69	13	100	97	98	**410**
Test	8	18	3	25	24	25	**103**
Total	**41**	**87**	**16**	**125**	**121**	**123**	**513**

A su vez, el conjunto de entrenamiento con los artículos enteros[41] fue dividido nuevamente en cinco partes de manera estratificada para realizar *five fold cross validation*, lo cual será detallado más precisamente en la sección 5.5.3. Se utilizó dicho conjunto de entrenamiento como conjunto de desarrollo para estudiar la mejor configuración de parámetros.

[40] `https://github.com/flairNLP/flair`

[41] No se realizó *five-fold cross validation* con el corpus basado solo en los títulos por falta de recursos.

Nos basamos en la propuesta de Reimers y Gurevych [53] para la selección de parámetros a variar y sus posibles valores. Los parámetros tenidos en cuenta fueron los siguientes:

- **Optimizador:** Se probó con *SGD* (*Stochastic gradient descent*) y *Adam* (Adaptive Moment Estimation), los cuales serán explicados en la sección 5.5.3. A la vez, para cada uno de ellos, se probaron los valores del *learning rate* (LR): {0.001, 0.05, 0.01, 0.1}.
- **Mini-batch size:** $\{8, 16, 32\}$.
- **Cantidad de capas de la red BiLSTM:** $\{1, 2, 3\}$.
- **Hidden size:** $\{80, 128, 256\}$.
- **Variational droput:** {0, 0.05, 0.1, 0.5}.
- **Embeddings**: Se probaron siete configuraciones distintas de *word embeddings*.

Una vez elegido el conjunto de parámetros con mejores resultados, se procedió a entrenar el algoritmo sobre el total de los datos de entrenamientos separados (80 % sobre el total). Los datos del corpus anotado son dados como parámetro a un objeto llamado *ColumnCorpus*, que se encarga de entrenar un etiquetador de secuencias sobre un conjunto etiquetado propio[42].

En el algoritmo 1 podemos observar cómo es el proceso para entrenar un corpus con datos etiquetados y entrenar un modelo de etiquetado de secuencias. Luego, en el algoritmo 2, se puede observar cómo realizar el etiquetado de oraciones utilizando el modelo entrenado anteriormente.

Algorithm 1: Algoritmo para NER utilizando la librería Flair

```
Cargar corpus con datos de entrenamiento, test y desarrollo
tag_dictionary = Crear diccionario con etiquetas de tipo NER
/* Inicializar embeddings                                    */
embeddings = [embedding_1, ..., embedding_n]
embeddings = StackedEmbeddings(embeddings)
/* Crear el etiquetador de secuencias                        */
tagger = SequenceTagger(hidden_size, embeddings, tag_dictionary,
 tag_type='ner')
/* Crear y entrenar el modelo                                */
trainer = ModelTrainer(tagger, corpus)
trainer.train(learning_rate, mini_batch_size, max_epochs=100)
```

5.5.3. *Five-fold cross validation*

Una vez que se encuentra dividido el conjunto de datos en *training* y *test*, se debe elegir el conjunto de parámetros que mejores resultados provee para la métrica seleccionada (en

[42] La herramienta provee distintos corpora para el caso de que no se necesite utilizar un corpus propio, como el *'CONLL_03_SPANISH'* que contiene cuatro tipos de etiquetas (personas, organizaciones, ubicaciones y nombres varios) o el *'WIKINER_SPANISH'* entrenado sobre Wikipedia.

Algorithm 2: Algoritmo para etiquetar oraciones

```
/* Inicializar oraciones                                    */
sentences = list()
for oración ← oraciones do
  | sentences.add(Sentence(oración));
end
/* Cargar modelo entrenado                                  */
tagger = SequenceTagger.load(trainer)
/* Etiquetar oraciones                                      */
tagger.predict(sentences)
```

lo que concierne a este trabajo, será el *F1 micro average*). Para esto, se subdivide el conjunto de entrenamiento nuevamente en *training* y *test*, y se evalúan los resultados para distintos parámetros. El problema que esto trae es que los resultados para un subconjunto de *test* pueden dar distinto que para otro. Para evitarlo, se utiliza una técnica llamada *five-fold cross validation*, que consiste en dividir el conjunto de entrenamiento en cinco *folds* y cada uno de ellos se divide, a su vez, en entrenamiento y test, asegurando que los datos de este último sean disjuntos (ver figura 5.6). Luego, por cada conjunto de datos separados (cada *split* del ejemplo), se entrena el algoritmo y se calcula el F1-score sobre el *fold* de test, obteniendo como resultado final el promedio de los F1-score de cada *fold*. Para finalizar, se entrena todo el conjunto original de entrenamiento con los parámetros óptimos seleccionados y se utiliza el conjunto de test separado inicialmente para evaluar el desempeño del algoritmo. En la tabla 5.8 se puede observar la distribución de artículos para cada fold.

Para cada distinto parámetro que se quería probar, se dejaron fijos los demás valores (estos varían según la instancia, dejando en cada una los mejores hallados hasta el momento[43]) y se compararon los resultados de *precision*, *recall* y *F1 micro-average* sobre el conjunto de test. Se utilizó el F1-score para elegir los valores óptimos de cada iteración, los cuales serán señalizados en negrita.

A continuación, se presentan los distintos parámetros modificados, junto a una explicación de los mismos y los resultados obtenidos.

Optimizador

Los optimizadores son los algoritmos que se utilizan en la red neuronal para ajustar los valores de los pesos y el *learning rate* de forma tal de minimizar el error (diferencia entre el resultado obtenido y el esperado). Una de las formas que se mencionó en la sección 3.4 es la del *descenso por gradiente*. Una variación es el *SGD* que actualiza los parámetros más frecuentemente, calculándolo sobre cada uno o un subconjunto de los ejemplos de entrenamiento en vez de sobre todo el conjunto entero. Otro tipo es el *Adam* [42], en el cual distintos parámetros tienen un *learning rate* adaptativo individual que se va modificando en base a los valores del gradiente de la función de error, distinto al SGD que mantiene el *learning rate* fijo para todos los parámetros y para cada ciclo.

[43] Cuando era posible, es decir cuando ya se tenían los resultados de las corridas anteriores en esa instancia.

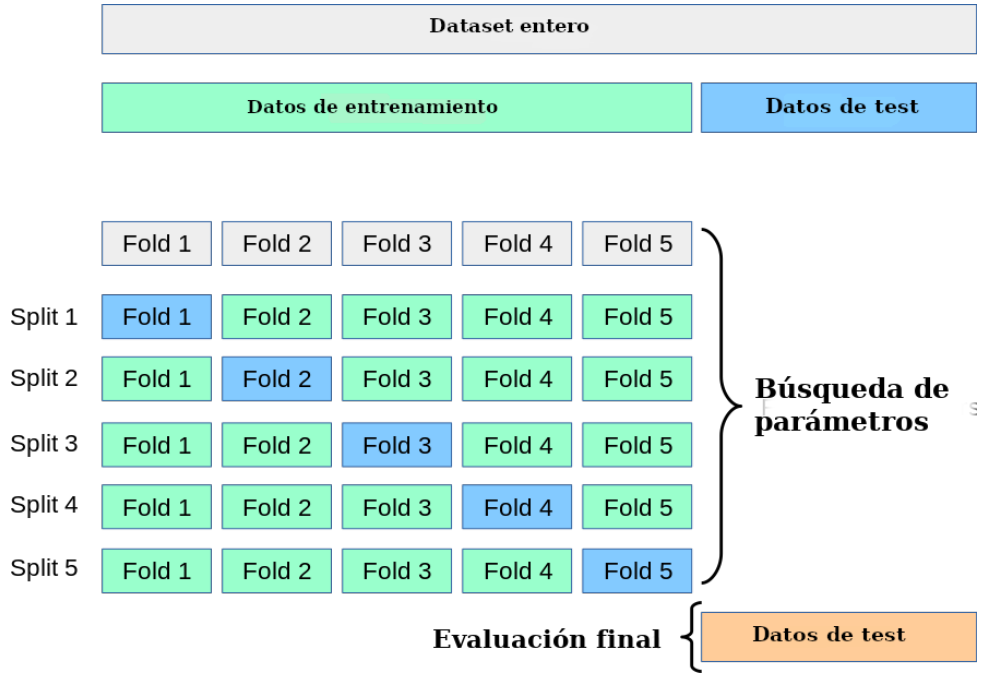


Fig. 5.6: Ejemplo de *Five-fold cross validation*. Los datos señalizados en verde representan al conjunto de entrenamiento y los azules al de test. Gráfico adaptado de `https://scikit-learn.org/stable/modules/cross_validation.html`.

Fold	Set	Chagas	Dengue	Guillain-barré	Hantav.	Saramp.	Zika	Total
1	Train	26	55	11	80	77	79	**328**
	Test	7	14	2	20	20	19	**82**
2	Train	26	55	11	80	77	79	**328**
	Test	7	14	2	20	20	19	**82**
3	Train	27	55	10	80	78	78	**328**
	Test	6	14	3	20	19	20	**82**
4	Train	27	55	10	80	78	78	**328**
	Test	6	14	3	20	19	20	**82**
5	Train	26	56	10	80	78	78	**328**
	Test	7	13	3	20	19	20	**82**

Tab. 5.8: Distribución de los artículos por enfermedad en cada fold del five-fold cross validation.

Los parámetros fijos para este experimento fueron[44]:

# layers	Hidden size	Dropout	Mini batch size	Embeddings
1	256	0.5	32	Config 1

Se utilizaron como optimizadores el *SGD* y el *Adam*. Para cada uno de ellos, se fue variando el valor del *learning rate* (LR). Los resultados pueden observarse en el gráfico 5.7 y fueron los siguientes:

[44] La *'Config 1'* de los *embeddings* será explicada posteriormente en el apartado de *Embeddings*.

Optimizer	LR	P	R	F1
SGD	0.001	0.621	0.440	0.514
	0.05	0.641	0.630	0.635
	0.01	0.656	0.535	0.589
	0.1	0.645	0.631	**0.638**
Adam	0.001	0.631	0.619	0.624
	0.05	0.469	0.176	0.237
	0.01	0.631	0.410	0.496
	0.1	0.240	0.120	0.151

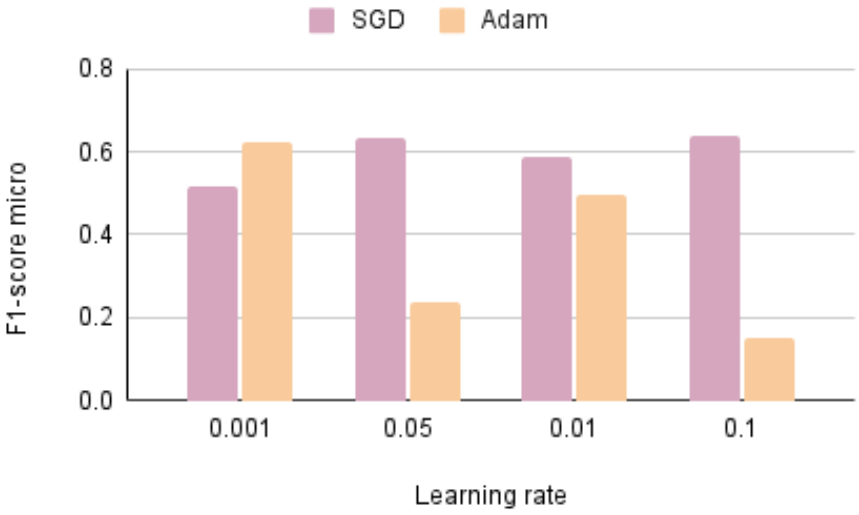


Fig. 5.7: Resultados del algoritmo de RNN utilizando distintos optimizadores y *learning rates*.

Como se puede observar en los resultados, es importante variar los valores del *learning rate* al iterar distintos tipos de optimizador, ya que el desempeño puede diferir drásticamente (como en el caso del algoritmo *Adam* que presenta una gran diferencia entre los resultados utilizando *learning rate* de 0.1 y 0.001). El algoritmo que mejor resultados obtuvo fue, con F1-score de 0.638, el SGD con *learning rate* de 0.1. Estos valores serán los fijos en los siguientes experimentos.

Mini-batch learning

En un *mini-batch learning*, los datos de entrenamiento se dividen en distintos *batches* (según el valor del *mini-batch size*). Para cada época de entrenamiento de la red neuronal, se utilizan los datos de cada *batch* por separado para entrenar la red, computar el gradiente y modificar los pesos correspondientes.

Los parámetros fijos para este experimento fueron:

# layers	Hidden size	Dropout	Optimizador	LR	Embeddings
1	256	0.5	SGD	0.1	Config 1

Se utilizaron los valores $\{8, 16, 32\}$ para los distintos *mini-batch size*. Los resultados fueron:

Mini-batch size	P	R	F1
8	0.644	0.635	**0.639**
16	0.644	0.631	0.637
32	0.645	0.631	0.638

Como puede observarse, si bien no hay gran diferencia entre los valores obtenidos, los mejores se obtuvieron con un *mini-batch size* de 8, el cual se dejará fijo para los próximos experimentos.

RNN layers

Otro parámetro sobre el cual se puede iterar es la cantidad de capas BiLSTM que tiene la red. En caso de existir más de una, en la librería utilizada, se crea una *stacked BiLSTM*, en la cual la salida de una de las capas pasa a ser parte de la entrada de la siguiente, permitiendo representaciones más complejas del modelo. En la figura 5.8 se puede observar un ejemplo del comportamiento de una *stacked* red neuronal recurrente bidireccional con dos *layers* en *tensor flow*[45].

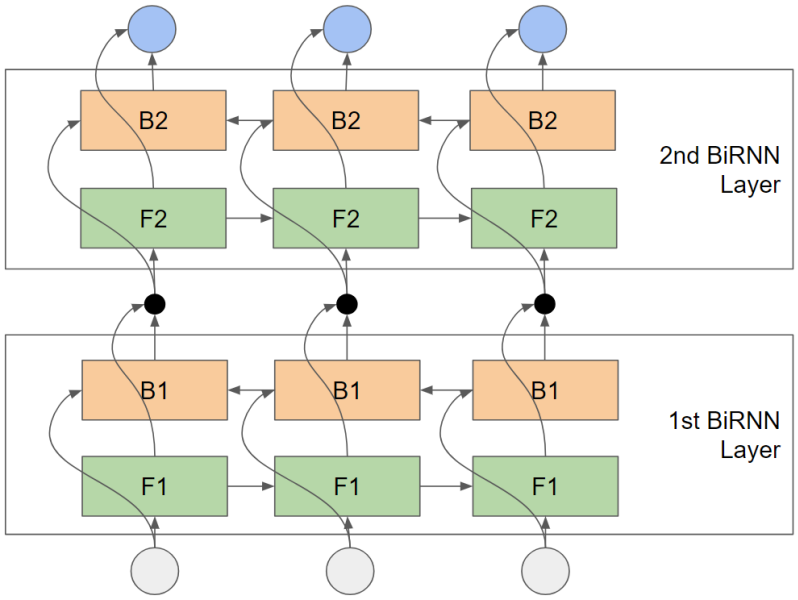


Fig. 5.8: Ejemplo del funcionamiento de una *stacked* red neuronal recurrente bidireccional de dos *layers* en *tensor flow*. $F1$ y $F2$ representan cada una de las capas *forward* de la red, y $B1$ y $B2$ representan las capas *backward*. Además, cada caja rectangular representa una celda de la red neuronal. Gráfico tomado de `https://stackoverflow.com/questions/49242266/difference-between-bidirectional-dynamic-rnn-and-stack-bidirectional-dynamic-rnn`.

Los parámetros fijos para este experimento fueron:

Hidden size	Dropout	Optimizador	LR	Mini batch size	Embeddings
256	0.5	SGD	0.1	8	Config 1

[45] Tensor flow es una librería de aprendizaje automático. Para más información visitar `https://www.tensorflow.org/`.

Se probaron los valores $\{1, 2, 3\}$ para la cantidad de capas de la red neuronal BiLSTM. Los resultados fueron:

# layers	P	R	F1
1	0.644	0.635	0.639
2	0.614	0.664	0.637
3	0.622	0.664	**0.642**

Los mejores resultados se obtuvieron utilizando tres capas de redes BiLSTM. La misma no fue fijada en los siguientes experimentos ya que el tiempo de cómputo de este experimento fue elevado y se ejecutó en paralelo mientras se iteraban los demás parámetros.

Hidden size

El parámetro *hidden size* determina el tamaño del vector de cada estado oculto en cada paso de la red neuronal recurrente (en el caso de la figura 3.7, sería el tamaño del vector h). Como fue explicado en la sección 3.4, estos vectores son utilizados para que la red tenga memoria de los estados previos.

Los parámetros fijos para este experimento fueron:

# layers	Dropout	Optimizador	LR	Mini batch size	Embeddings
1	0.5	SGD	0.1	8	Config 1

Se utilizaron los valores $\{80, 128, 256\}$ para el valor del *hidden size*. Los resultados fueron:

Hidden size	P	R	F1
80	0.647	0.629	0.638
128	0.653	0.625	0.638
256	0.644	0.635	**0.639**

Como se puede observar en los resultados, el tamaño del vector no generó grandes cambios. El mejor F1-score obtenido fue utilizando un *hidden size* de 256, lo cual no modifica los parámetros fijos que se venían utilizando.

Variational dropout

El *dropout* es una técnica de regularización de las redes neuronales que sirve para reducir el *overfitting* (ver sección 3.4) y mejorar el desempeño de la red. Esta consiste en ignorar algunas de las unidades de la red (ya sean ocultas o no) poniéndole un valor de 0 (ver figura 5.9). El valor seleccionado p establece, para cada unidad, la probabilidad de retenerla, lo cual puede observarse también como la probabilidad $1 - p$ de ignorarla. En el *dropout naive*, se aplica una nueva máscara en cada paso de la capa y en un *variational dropout* (propuesto por Gal y Ghahramani [29]) se utiliza la misma máscara en cada capa. Se optó por este último tipo siguiendo la recomendación de la propuesta de Reimers y Gurevych [53].

Los parámetros fijos para este experimento fueron:

# layers	Hidden size	Optimizador	LR	Mini batch size	Embeddings
1	256	SGD	0.1	8	Config 1

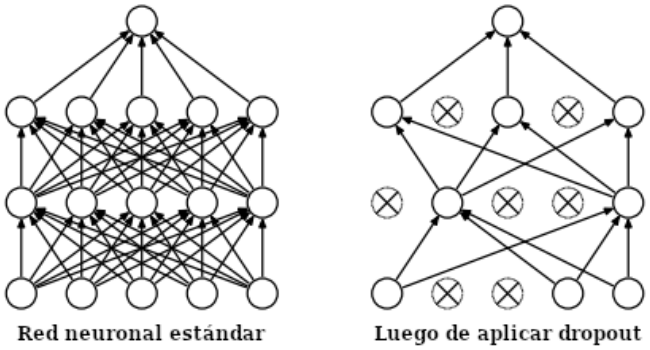


Fig. 5.9: Ejemplo de *dropout* en una red neuronal. Gráfico adaptado de Srivastava y col. [61]

Se probaron los valores {0, 0.05, 0.1, 0.5} para el valor del *variational dropout*. Los resultados fueron:

Variational dropout	**P**	**R**	**F1**
0.00	0.627	0.6015	0.614
0.05	0.630	0.605	0.617
0.1	0.625	0.615	0.618
0.5	0.644	0.635	**0.639**

En este caso, los mejores resultados se obtuvieron utilizando el *variational dropout* de 0.5 que ya se venía utilizando.

Embeddings

Como fue explicado en la sección 3.2, los *embeddings* son formas para representar textos en forma de vectores numéricos. La librería de *Flair* ofrece distintos tipos de *embeddings* para el idioma español, tanto a nivel de palabras como a nivel de caracter. Los siguientes fueron utilizados:

- **Embeddings a nivel de palabra:** *FastText*[46] entrenados sobre Wikipedia (*wiki dumps* del año 2018). Estos son *embeddings* clásicos sin contexto, por lo que cada palabra distinta cuenta con una única representación.

- **Embeddings a nivel de caracter contextuales:** Son a nivel de caracter y tienen en cuenta el contexto en el que se encuentran. Existe una versión para el paso *forward* y otra para el *backward*. Se encuentran entrenados sobre *Wikipedia* (*Wiki dumps*[47]). Estos son el *Flair Embedding* y el *Pooled Flair Embeddings*. Se diferencian en que los *embeddings* del segundo de ellos van evolucionando en el tiempo (es decir, son dinámicos), por lo que una palabra en una misma oración en diferentes puntos en el tiempo puede tener representaciones distintas.

[46] Se pueden encontrar en `https://fasttext.cc/docs/en/crawl-vectors.html`. [Accedido en Mayo de 2021].
[47] `https://archive.org/details/eswiki-20181201` [Accedido en Mayo de 2021]

También se trabajó con los *embeddings* de **Uchile**[48] que utilizan el corpus *Spanish Billion Word Corpus* [15], el cual contiene 1400 millones de palabras en español obtenidas de la *web*. Se utilizaron los *word embeddings* entrenados con los algoritmos *FastText*, *Glove* y *Word2Vec*, los cuales llamamos *SBWC-FastText*, *SBWC-Glove* y *SBWC-Word2Vec* respectivamente.

Otros de los conceptos introducidos por *Flair* es el de *Stacked Embeddings*, lo cual permite combinar distintos tipos de *embeddings* en uno solo. Esto es útil para combinar los *Flair embeddings* en su versión *forward* y *backward*, y también para combinar otros tipos, como los clásicos.

Los parámetros fijos para este experimento fueron:

# layers	Hidden size	Optimizador	LR	Mini batch size	Dropout
1	256	SGD	0.1	8	0.5

Se probaron las siguientes configuraciones de *embeddings*:

- **Configuración 1:** *stacked embedding* compuesto de [Embedding clásico wikidumps-FastText, Flair embedding forward, Flair embedding backward].
- **Configuración 2:** *stacked embedding* compuesto de [Embedding clásico wikidumps-FastText, Pooled flair embedding forward, Pooled flair embedding backward].
- **Configuración 3:** *stacked embedding* compuesto de [SBWC-FastText, Flair embedding forward, Flair embedding backward].
- **Configuración 4:** *stacked embedding* compuesto de [SBWC-Glove, Flair embedding forward, Flair embedding backward].
- **Configuración 5:** *stacked embedding* compuesto de [SBWC-Word2Vec, Flair embedding forward, Flair embedding backward].
- **Configuración 6:** *stacked embedding* compuesto de [Flair embedding forward, Flair embedding backward].
- **Configuración 7:** SBWC-FastText (sin encontrarse en un *StackedEmbedding*).

Los resultados fueron:

Embedding	P	R	F1
Config 1	0.644	0.635	**0.639**
Config 2	0.658	0.617	0.637
Config 3	0.655	0.631	**0.642**
Config 4	0.643	0.633	0.638
Config 5	0.643	0.624	0.634
Config 6	0.640	0.635	0.637
Config 7	0.657	0.591	0.621

[48] Disponible en `https://github.com/dccuchile/spanish-word-embeddings` [Accedido en Mayo de 2021].

En estos casos, al comparar utilizar *PooledFlairEmbeddings* (0.637) contra *FlairEmbeddings* (0.639), el que mejor resultados ofreció fue el segundo de ellos. En cuanto a la variación de los *embeddings* clásicos en español utilizados en conjunto con los de tipo *flair* dentro de un *stacked embedding*, el que mejor resultados ofreció fue la configuración 3 (0.642), el cual utiliza el corpus *SBWC-FastText*.

5.5.4. Resultados

Se entrenó el algoritmo basado en redes neuronales sobre el conjunto de entrenamiento, tanto para el artículo entero como para los títulos, y el testing se realizó sobre el conjunto de test. Los parámetros utilizados fueron los que mejores resultados dieron según el *five-fold cross validation* y fueron los siguientes:

# layers	**Hidden size**	**Optimizador**	**LR**
3	256	SGD	0.1

Mini batch size	**Dropout**	**Embeddings**
8	0.5	Config. 3

Para obtener estos parámetros, se realizaron en total 23 iteraciones a través del método *five-fold cross validation*. Si bien se podrían haber iterado sobre otras configuraciones (por ejemplo, al ver que para el *variational dropout* el mejor valor fue el más alto, se podría haber probado con valores mayores), nos limitamos a las propuestas por Reimers y Gurevych [53]. Una de las desventajas de dejar fijo un conjunto de parámetros y variar otros es que algunos de los que se encuentran fijos puede influir en los demás. Para evitar esto, se podría haber utilizado el enfoque *grid search*, el cual itera sobre todas las combinaciones posibles de parámetros. Sin embargo, no se optó por el mismo debido a falta de recursos para iterar sobre la gran cantidad de parámetros que se poseían, a diferencia de la metodología elegida que nos permitió probar una amplia gama de los mismos.

A continuación se presentan una serie de tablas con los resultados del algoritmo, tanto sobre los títulos como los artículos enteros, sobre el conjunto de test. Inicialmente, se muestra la cantidad total y diferente de términos hallados por entidad nombrada. Luego, algunos de los términos más frecuentemente hallados para ciertas entidades. Finalmente, los resultados de *precision*, *recall* y *F1-score*.

En la tabla 5.9 se pueden observar la cantidad de entidades y modificadores encontrados por el algoritmo basado en redes neuronales y los anotadores sobre el conjunto de datos de test, tanto para los títulos como para los artículos enteros.

Entidad	**Algoritmo**		**Anotadores**	
	Total	**Diferentes**	**Total**	**Diferentes**
Abbreviation	0	0	2	1
Date	103	103	99	99
Disease	118	13	112	12
Host	2	2	8	8
Location	111	62	117	61
Number of cases	0	0	2	1
Origin	2	2	11	9
Transmission form	6	4	5	4
Conditional	-	-	0	0
Negation	-	-	0	0
Past	-	-	0	0
Uncertainty	0	0	3	2
Total	**342**	**186**	**359**	**197**

Entidades en los títulos.

Entidad	**Algoritmo**		**Anotadores**	
	Total	**Diferentes**	**Total**	**Diferentes**
Abbreviation	0	0	2	1
Date	109	109	147	134
Disease	655	51	596	70
Host	132	80	131	85
Location	561	273	494	246
Number of cases	513	250	436	239
Origin	141	60	180	79
Transmission form	40	29	51	40
Conditional	-	-	0	0
Negation	9	5	11	7
Past	124	87	91	67
Uncertainty	49	21	67	42
Total	**2333**	**965**	**2206**	**1010**

Entidades en los artículos enteros.

Tab. 5.9: Tipos y cantidad de entidades y modificadores encontrados por el algoritmo basado en redes neuronales y los anotadores sobre el conjunto de datos de test.

En estas tablas, se puede observar que las cantidades de términos hallados entre el algoritmo y los anotadores fueron similares (tanto los totales como los diferentes). En el título, los anotadores hallaron un 55 % de términos distintos, mientras que el algoritmo un 54 %, y en los artículos enteros los anotadores hallaron un 48 % distintos contra un 41 % por el algoritmo. Sin embargo, esto no significa que los términos hallados sean los mismos, por lo que dicha información se debe complementar con los valores de *precision* y *recall*, que serán mostrados posteriormente.

En las tablas 5.10, 5.11 y 5.12 se pueden observar ejemplos de términos anotados con mayor frecuencia, para ciertas entidades, por el algoritmo de redes neuronales en los títulos y los artículos enteros sobre el conjunto de datos de test.

Location	**# ocur.**
Brasil	13
Colombia	7
Puerto Rico	5
Venezuela	4
Argentina	4
Bolivia	3
México	3
Costa Rica	3
Chile	3
El Salvador	2
Perú (Iquitos)	2
Paraguay	2
Perú: (Ica)	2
Brasil (Cea)	2
Perú : (múltiples departamentos)	2

Date	**# ocur.**
18 de enero de 2016	1
19 de mayo de 2011	1
18 de diciembre de 2016	1
29 de agosto de 2011	1
06 de junio de 2013	1
15 de agosto de 2011	1
18 de abril de 2012	1
29 de julio de 2013	1

Disease	**# ocur.**
zika	27
hantavirus	25
sarampión	24
dengue	20
chagas	8
microcefalia	6
guillain barré	2
sífilis	1
congénitas	1
sindrome pulmonar	1
chikungunya	1
síndrome neurologico	1
guillain - barré	1

Host	**# ocur.**
strongyloides	1
niños	1

Transmission form	**# ocur.**
transmisión vectorial	2
transmisión oral	2
trasmisión oral	1
embarazadas	1

Origin	**# ocur.**
no - vacunados	1
acai	1

Tab. 5.10: Ejemplos de términos frecuentes hallados por el algoritmo de RNN en los títulos sobre el conjunto de datos de test, junto a la cantidad de ocurrencias, para ciertas entidades.

Disease	**# ocur.**
dengue	139
zika	136
sarampión	102
hantavirus	91
chagas	38
microcefalia	35
enfermedad	23
sph	10
virus	9
guillain - barré	7
guillain barré	6
hanta	5
chikungunya	4
virus hanta	4
sífilis	3
malformación congenita	3
chikunguna	3
hantavirosis	2
hanta virus	2
chagas congénito	2

Host	**# ocur.**
bebés	17
mujeres embarazadas	10
niños	9
bebés recién nacidos	4
recién nacidos	4
fetos	4
embarazadas	3
adulto	3
mujer de 27 años	2
gestantes	2
mujeres	2
mujer de 44 años	2
niño de 5 meses	2
hombre de 38 años	2
menores de 15 años de edad	1
ciudadano ruso, de 39 años	1
hombre de 65 años	1
infantes	1
turista	1
madres	1

Transmission form	**# ocur.**
transmisión oral	4
transmisión sexual	3
picadura	2
transmisión vectorial	2
orina	2
persona a persona	2
trasmisión oral	2
consumo	2
vía sexual	1
congénita	1
abstinencia sexual	1
roedores _ calomys laucha	1
heces	1
sexualmente entre seres humanos	1
inhalación, en ambientes cerrados donde las heces o la orina	1
ratones silvestres	1
contacto entre especies simpáticas	1
contacto con orina, heces, saliva o sangre de ratones	1
vía respiratoria	1

Origin	**# ocur.**
roedores	19
mosquito	17
zika	11
_ aedes aegypti _	10
mosquito _ aedes aegypti _	8
zancudo	7
mosquito aedes aegypti	5
mosquitos	4
vinchucas	3
_ aedes _	2
vinchuca	2
ratones silvestres	2
ratón colilargo	2
jugo de caña de azucar	2
_ t. cruzi _	2
colilargo	1
roedores selvaticos	1
insectos	1
contagio de mujeres embarazadas	1
trypanosoma cruzi _	1

Tab. 5.11: Ejemplos de términos frecuentes hallados por el algoritmo de RNN en los artículos enteros sobre el conjunto de datos de test, junto a la cantidad de ocurrencias, para ciertas entidades.

Location	**# ocur.**
Brasil	47
Colombia	21
Venezuela	16
México	16
Puerto Rico	13
Bolivia	10
Honduras	10
Perú	9
Argentina	9
Piura	8
Bahía	8
Paraguay	7
Chile	7
Costa Rica	6
Ica	6
Guatemala	6
Buenos Aires	6
Ceará	6
República Dominicana	5
Cuba	5

Uncertainty	**# ocur.**
sospechosos	15
sospecha	7
casos probables	3
casos sospechosos	3
posibles casos	2
bajo investigación	2
sospechoso	2
en observación	2
puede	1
en estudio	1
probables	1
confirmar	1
fuertemente sospechosos	1
diagnóstico no concluyente	1
resultado sospechoso	1
posible brote	1
sospechas	1
esperan de los resultados	1
indicios sugieren	1
probable	1

Tab. 5.12: Ejemplos de términos frecuentes hallados por el algoritmo basado en redes neuronales en los artículos enteros sobre el conjunto de datos de test, junto a la cantidad de ocurrencias, para ciertas entidades.

En la tabla 5.13 se pueden observar los resultados de *precision*, *recall* y F1-score (match exacto y parcial) del método basado en redes neuronales para los títulos y los artículos enteros sobre el conjunto de test[49].

5.5.5. Análisis de los resultados

Para poder analizar los resultados obtenidos por el algoritmo de redes neuronales profundas contra los archivos anotados manualmente, debemos tener las tablas comparativas de estos últimos sobre el mismo conjunto de test.

En las tablas 5.14, 5.15 y 5.16 se pueden observar ejemplos de términos con mayor frecuencia anotados por los anotadores en los títulos y en los artículos enteros sobre el conjunto de datos de test, para ciertas entidades.

[49] Se señalan con guión (-) aquellas entidades que el algoritmo no es capaz de detectar ya que no se encontraban datos en el conjunto de entrenamiento. Esto es distinto a obtener un resultado de 0, ya que esto indica que encontró entidades, pero no hubo coincidencias.

Entidad	Match exacto			Match parcial		
	P	**R**	**F1**	**P**	**R**	**F1**
Abbreviation	0.00	0.00	0.00	0.00	0.00	0.00
Date	0.92	0.96	0.94	0.94	0.98	0.96
Disease	0.91	0.96	0.94	0.94	0.98	0.96
Host	0.50	0.12	0.20	0.50	0.12	0.20
Location	0.68	0.65	0.67	0.84	0.84	0.84
Number of cases	0.00	0.00	0.00	0.00	0.00	0.00
Origin	1.00	0.20	0.33	0.40	0.20	0.27
Transmission form	0.33	0.40	0.36	0.67	0.50	0.57
Conditional	-	-	-	-	-	-
Negation	-	-	-	-	-	-
Past	-	-	-	-	-	-
Uncertainty	0.00	0.00	0.00	0.00	0.00	0.00
Micro-average	**0.83**	**0.79**	**0.81**	**0.89**	**0.88**	**0.88**

Precision, recall y *F1-score* en los títulos.

Entidad	Match exacto			Match parcial		
	P	**R**	**F1**	**P**	**R**	**F1**
Abbreviation	0.00	0.00	0.00	0.00	0.00	0.00
Date	0.88	0.65	0.75	0.78	0.68	0.73
Disease	0.70	0.77	0.73	0.73	0.79	0.76
Host	0.53	0.53	0.53	0.59	0.61	0.60
Location	0.59	0.67	0.63	0.64	0.73	0.68
Number of cases	0.68	0.80	0.74	0.69	0.81	0.75
Origin	0.51	0.43	0.47	0.54	0.47	0.50
Transmission form	0.27	0.22	0.24	0.41	0.38	0.40
Conditional	-	-	-	-	-	-
Negation	0.11	0.09	0.10	0.12	0.09	0.11
Past	0.31	0.43	0.36	0.46	0.53	0.49
Uncertainty	0.39	0.28	0.33	0.50	0.38	0.43
Micro-average	**0.62**	**0.66**	**0.64**	**0.66**	**0.70**	**0.68**

Precision, recall y *F1-score* en los artículos enteros.

Tab. 5.13: Resultados del match exacto y parcial del método basado en redes neuronales profundas para títulos y artículos enteros sobre conjunto de datos de test.

Location	**# ocur.**
Brasil	12
Perú	8
Colombia	7
Argentina	6
Puerto Rico	5
Venezuela	4
Paraguay	3
Bolivia	3
Costa Rica	3
México	3
Honduras	2
Ecuador	2
Brasil (cea)	2
Brasil (mg)	2
República Dominicana	2

Date	**# ocur.**
17 de junio del 2016	1
24 de febrero del 2018	1
19 de enero, 2016	1
206 de mayo de 2017	1
mier 14 de abril de 2004	1
16 de octubre de 2015	1
30 de julio, 2015	1
13 de octubre de 2011	1

Disease	**# ocur.**
zika	25
hantavirus	25
sarampión	24
dengue	19
chagas	8
microcefalia	5
guillain barré	1
chikungunya	1
sídrome de guillain-barré	1
lesiones congénitas	1
síndrome de guillain barré	1
strongyloides	1

Host	**# ocur.**
embarazadas	1
recién nacidos	1
niños	1
comunidades vulnerables	1
inmigrantes	1
donantes de sangre	1
gestantes	1
embarazada	1

Origin	**# ocur.**
zika	3
brote familiar	1
roedores	1
no-vacunados	1
transmisión oral	1
importado	1
ratones	1
acai	1
gullain-barré	1

Transmission form	**# ocur.**
vectorial	2
transmisión oral	1
trasmisión oral	1
consumo	1

Tab. 5.14: Entidades comúnmente halladas por los anotadores en los títulos sobre el conjunto de datos de test, junto a la cantidad de ocurrencias.

Disease	**# ocur.**
zika	113
dengue	108
sarampión	88
hantavirus	80
chagas	34
microcefalia	31
enfermedad	19
virus	12
chikungunya	10
virus hanta	6
guillain barré	5
guillain-barré	5
estrongiloidiasis	5
denv-1	4
denv-2	4
sindrome guillain barré	3
denv-4	3
mal	3
guillain - barré	2
chikunguna	2

Host	**# ocur.**
bebés	11
embarazadas	9
niños	7
mujeres embarazadas	4
bebé	4
bebés recién nacidos	3
recién nacidos	3
mujeres	3
fetos	3
mujer	2
mujer de 27 años	2
hombre de 38 años	2
niño de 10 años	2
niño de 5 meses	2
niña de 6 meses	2
adultos latinoamericanos que viven en Europa	2
gestantes	2
mujer de 79 años	1
lactante de 10 meses de edad	1
turista extranjero	1

Transmission form	**# ocur.**
vectorial	4
aedes aegypti	3
transmisión oral	3
mosquito _aedes aegypti_	2
trasmisión oral	2
consumo	2
aedes aegypti	2
transmisión vertical (de madre a feto)	1
penetra en el organismo humano a traves de las vías respiratorias o de heridas	1
picado	1
mosquito	1
sexual	1
infección de la madre durante el embarazo	1
picadura	1
ingestión	1
ingesta	1
consume	1
sexualmente	1

Origin	**# ocur.**
zika	31
mosquito	23
roedores	22
virus	5
aedes aegypti	5
mosquitos	4
zancudo	4
importado	3
mosquito aedes aegypti	2
t. cruzi	2
vinchucas	2
vinchuca	2
aedes aegypti	2
jugo de caña de azucar	2
calomys laucha	2
mosquito _aedes aegypti_	2
ratones	2
ratas	2
parásito	2
roedor	2

Tab. 5.15: Entidades comúnmente halladas por los anotadores en los artículos enteros sobre el conjunto de datos de test, junto a la cantidad de ocurrencias.

Location	**# ocur.**
Brasil	31
Colombia	17
Perú	14
Puerto Rico	14
Paraguay	10
Argentina	10
Honduras	9
Venezuela	9
México	9
Buenos Aires	9
Bolivia	8
Guatemala	7
Costa Rica	7
Ceará	6
Bahía	5
Piura	5
República Dominicana	5
Uruguay	5
Panaa	4
Chubut	4

Uncertainty	**# ocur.**
sospechosos	9
casos sospechosos	6
probables	4
sospecha	4
probable	3
más de	2
posible	2
estan en observación	2
sospechoso	2
unas	1
notificaciones	1
bajo investigación	1
casos	1
había confianza	1
sospechosas	1
con sospecha	1
aguardan la confirmación	1
posiblemente asociada	1
riesgo de casos	1
a confirmar	1

Tab. 5.16: Entidades comúnmente halladas por los anotadores en los artículos enteros sobre el conjunto de datos de test, junto a la cantidad de ocurrencias.

En cuanto a los resultados de los títulos, se observó lo siguiente:

- **Location:** el algoritmo halló 111 entidades (con 62 distintas) contra 117 por el anotador (con 61 distintas). El *F1* exacto fue de 67 % y el parcial de 84 %. Una de las razones por las cuales el *match* parcial ofrece resultados considerablemente mayores es debido a inconsistencias en las anotaciones manuales. Algunos ejemplos de esto son:

  - Ante el texto *"Brasil (Sao Paulo)"*, a veces los anotadores marcaban dos entidades separadas *Brasil* (LOC) y *Sao Paulo* (LOC), o solo la entidad *Brasil* (LOC) sin anotar *Sao Paulo*. Sin embargo, en la mayoría de los casos, el algoritmo anotó correctamente una sola entidad: *Brasil (Sao Paulo)* (LOC).
  - El anotador agregó erróneamente signos ortográficos como parte de la entidad, como *puntos*, *comas* y *dos puntos* (por ejemplo, *"Brasil (Sao Paulo)."*).

- **Transmission form:** para este caso, el algoritmo halló en total 6 entidades contra 5 del anotador, con un match exacto de 36 % y parcial de 57 %. En las tablas 5.10 y 5.14 pueden observarse las diferencias en los términos anotados. Por ejemplo, en dos de ellos, el algoritmo anotó adicionalmente la palabra *transmisión* y el anotador no (el anotador anotó *vectorial* (TF) y el algoritmo *transmisión vectorial* (TF)). Algo a destacar es que el algoritmo fue capaz de detectar la palabra *transmisión* y su variante mal escrita *trasmisión*.

- **Negation, Past y Conditional:** No se encontraron entidades de este tipo en el conjunto de entrenamiento, por lo que el algoritmo no detectó ninguna y no se muestran resultados de los mismos en las tablas.
- **Date:** el match exacto fue de 94 % contra el parcial de 96 %. El algoritmo fue capaz de reconocer el apartado de la fecha en los títulos correctamente, por más que las mismas no se encontrasen en un formato estándar.
- **Disease:** En cuanto a los términos frecuentes para esta entidad, hubo cierta similitud entre los hallados por los anotadores y el algoritmo, como se puede observar en la figura 5.10. Además, los resultados del *F1* fueron altos (94 % para el exacto y 96 % para el parcial), por lo que el algoritmo las detecta de manera eficiente.
- **Number of cases y Uncertainty**: El algoritmo no fue capaz de encontrar ninguna de las apariones de estas entidades, por más que eran pocas las halladas por los anotadores (2 de tipo *Number of cases* y 3 de tipo *Uncertainty*).
- **Host y Origin:** El algoritmo halló muy pocos términos de tipo *host* (2 contra 8 del anotador) y *origin* (2 contra 11). En cuanto al *F1* exacto, los resultados fueron bajos: *host* 20 % y *origin* 33 %. Sin embargo, si se observa el valor de *precision*, los hallados de tipo *Host* fueron la mitad correctos (50 %) y los de *Origin* todos (100 %). Esto indica que los que halló fueron correctos, pero falla en encontrarlos todos.

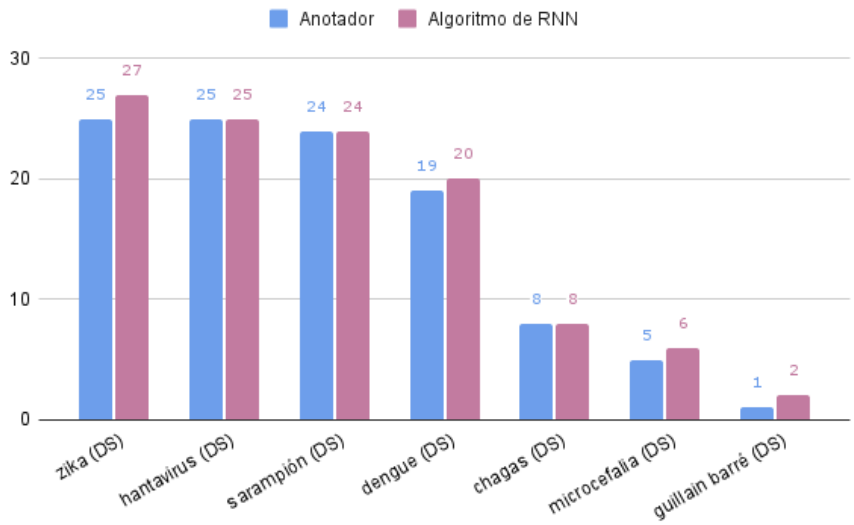


Fig. 5.10: Términos frecuentes de tipo *Disease* hallados en los títulos entre los anotadores y el algoritmo de RNN.

En cuanto a los artículos enteros:

- **Transmission form y origin:** como ya fue mencionado anteriormente, se presentaron durante el proceso de anotación dificultades para detectar estas entidades. Si bien se intentó definir de manera más precisa la misma a través de las distintas iteraciones y se realizaron correcciones posteriores, los resultados demostraron que

no fue suficiente. Este tipo de inconsistencias genera problemas en la generalización del algoritmo. Algunos problemas encontrados fueron:

- Los anotadores anotaron términos erróneos como *aedes aegypti* (TF) y *mosquitos* (TF), que deberían ser de tipo *Origin*. Además, fallaron en detectar otras como *'persona a persona'* (TF).
- El algoritmo detectó correctamente: *aedes aegypti* (Or), *mosquitos* (Or) y *'persona a persona'* (TF).
- El algoritmo etiquetó erróneamente *roedores _ calomys laucha* (TF) y *ratones silvestres* (TF). Estas deberían haber sido anotadas como entidades de tipo *Origin*.
- Las entidades de tipo *Transmission form* tienen la particularidad de que en algunos casos son inusualmente largas, a diferencia de otras como *Disease*. Por esta razón, hubo casos en los que el algoritmo detectó parcialmente entidades complejas, como la entidad *'contacto con orina, heces, saliva o sangre de ratones'* y *'respirar el aire contaminado con orina, excreta o saliva del roedor'*, en las cuales el anotador etiquetó parte de ellas y no su totalidad. Por esta razón, se puede observar una gran diferencia entre el resultado del match exacto y el parcial (21 % de diferencia).

- **Disease:** el algoritmo halló 655 totales y 51 distintas, contra las 596 totales y 70 distintas de los anotadores, con *F1* exacto de 73 % y parcial de 76 %. Algunas observaciones sobre los resultados fueron:

  - Los anotadores anotaron entidades erróneas como *rubéola* o *poliomielitis*, pero el algoritmo no.
  - El algoritmo anotó entidades erróneas como *paludismo* y *sífilis*, que fueron también anotadas por los anotadores.
  - El algoritmo fue capaz de encontrar distintas variantes para referirse a una misma enfermedad como, en el caso de guillain barré: guillain - barre, guillain barre, sindrome guillain - barre, guillaim - barre, guillain barre; o en el caso de dengue: dengue, dengue grave, denv 1, denv 2, dengue severo.

- **Origin y Disease:** Si bien llegó a detectar *zika* como causa, para el artículo entero solo encontró 11 ocurrencias contra las 31 halladas por los anotadores. Tras un análisis de las anotaciones, se pudo observar que los demás casos que deberían haber sido anotados como *Origin* fueron anotados como *Disease*. Esto genera una disminución de los resultados tanto para *Origin* como para *Disease*.

- **Date y Past:** el algoritmo anotó como *Date* aquellas fechas que se encontraban mayormente en el apartado del título, por lo que es razonable haber obtenido una *precision* de 88 % (lo cual indica que la mayoría de las que encontró fueron correctas). Sin embargo, cuando una fecha se encontraba en el texto, hubo tendencia a que la anotara como *Past*, por más que la fecha fuese la misma del encabezado (por ejemplo, si la fecha del artículo databa del año *2014* y en el cuerpo del artículo se encontraba la fecha *2014*, el algoritmo la anotó como *Past* en vez de *Date*). Esto se refleja en el valor de *recall* para la entidad *Date* (65 %) indicando que no encuentra todas las

que debería. Esto implica, a la vez, una disminución en el valor de *precision* para la entidad de tipo *Past* (con un exacto de 31 % y parcial de 46 %), ya que muchas de las que etiqueta como *Past* deberían ser de tipo *Date*. Otros factores que afectan el resultado de la entidad *Past* son los errores en el corpus anotado, habiendo entidades de tipo *Past* que fueron anotadas como *Date*.

- **Abbreviation, Host, Negation, Origin, Past, Transmission Form y Uncertainty:** Estas entidades tuvieron resultados por debajo del *micro average*. Esto se debe tanto a inconsistencias en las anotaciones, como al amplio espectro que abarcan las entidades de estos tipos.

- **Conditional:** No se encontraron entidades de este tipo en el conjunto de entrenamiento, por lo cual el algoritmo no detecta este tipo de entidad en los artículos enteros.

- **Number of cases:** el *F1* exacto fue de 74 % y el parcial de 75 %, siendo una de las entidades con mejores resultados (junto a *Disease* y *Date*). Si bien fue capaz de detectar gran parte de los términos (con un valor de *recall* exacto de 80 %), tuvo un mayor problema en los términos que detectó como tales (con *precision* exacto de 68 %). Esto se debió a inconsistencias en las anotaciones. Algunos ejemplos de lo mismo fueron:

  - En los casos de infectados por cantidad de habitantes, como por ejemplo *'116 casos por cada 100.000 habitantes'*, el anotador etiquetó erróneamente solo 116 (NoC), mientras que el algoritmo etiquetó toda la frase. En otros casos similares (en los que se mencionan infectados por cada tantos habitantes), tanto los anotadores como el algoritmo etiquetaron las frases completas.
  - El algoritmo etiquetó erróneamente casos referentes a otras enfermedades que no son de interés (por ejemplo, casos de *hepatitis*) que no fueron anotadas como tales por los anotadores.
  - Hubo términos que fueron anotados correctamente como *Number of cases* por el algoritmo y no por los anotadores.

- Si se compara el *F1* exacto contra el parcial, aquellas entidades de definición un tanto ambiguas, tuvieron mejores resultados si se observa el parcial: *transmission form* pasó de 24 % a 40 %, *past* de 36 % a 49 % y *uncertainty* de 33 % a 43 %. Una de las razones para lo mismo es debido a inconsistencia en las anotaciones que dificultan el aprendizaje del algoritmo. Por ejemplo, para el caso de la entidad *Uncertainty*, se detectaron casos en los que para la frase *'casos sospechosos'*, se anotó a veces la frase entera y otras veces solo *sospechosos*.

- En cuanto a algunos de los términos hallados para ciertas entidades con mayores frecuencias, se puede observar que el algoritmo en general etiquetó más ocurrencias de ciertos términos que los anotadores, como se puede observar en la figura 5.11.

Como conclusión, en los títulos, los mejores resultados se obtuvieron en entidades como *Disease*, *Date* y *Location*, que son aquellas más frecuentes en los títulos, por lo que el algoritmo tiene más datos para el entrenamiento. Sin embargo, para las demás entidades, los resultados del *F1* exacto se encontró por debajo del 36 %. Para el caso de los artículos

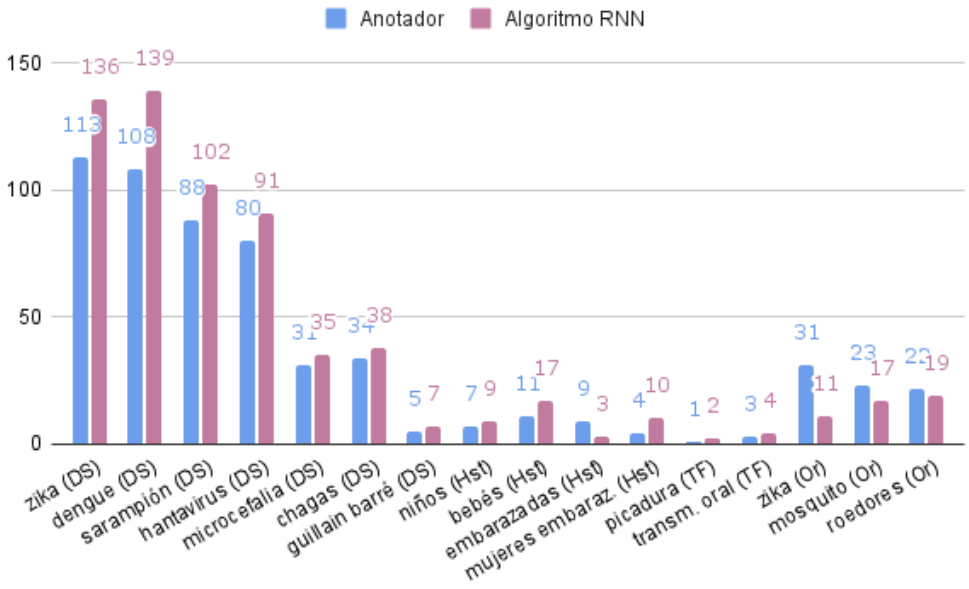


Fig. 5.11: Términos frecuentes hallados en los artículos enteros entre los anotadores y el algoritmo de RNN.

enteros, hubo ciertas entidades para las que se obtuvieron mejores resultados de *F1* exacto en comparación con los títulos, como la entidad *Number of cases* (de 0 % en los títulos a 74 % en los artículos enteros), *Origin* (de 33 % a 47 %) y *Host* (de 20 % a 53 %). Otras entidades, como *Negation*, *Uncertainty*, *Past* y *Transmission form*, tuvieron resultados bajos. Esto se condiciona con que son las entidades para las cuales más se dificultó la anotación, demostrado en los resultados del *inter-annotator agreement* presentados en la sección 4.4.4.1.2. Por esta razón, una mejora en el corpus anotado y en los criterios de anotación debería resultar en una mejora a este algoritmo.

5.6. Comparación de los métodos

Para poder comparar el algoritmo basado en redes neuronales profundas contra el basado en reglas, debemos tener los resultados sobre el mismo conjunto de datos. Por lo tanto, iniciamos presentando tablas de los resultados del algoritmo basado en reglas sobre el mismo conjunto de datos de test utilizado en el algoritmo de redes neuronales, del cual ya se tienen las tablas en la sección anterior. Mostramos inicialmente la cantidad de términos totales y diferentes hallados. Luego, algunos de los términos más frecuentemente hallados para ciertas entidades. Finalmente, los resultados de *precision*, *recall* y *F1-score*.

En la tabla 5.17 se pueden observar la cantidad de entidades y modificadores encontrados por el algoritmo basado en reglas (*rule-based*) contra el de redes neuronales (*RNN*) sobre el conjunto de datos de test, tanto en los títulos como en los artículos enteros.

Entidad	Algoritmo RNN		Algoritmo rule-based	
	Total	**Diferentes**	**Total**	**Diferentes**
Abbreviation	0	0	-	-
Date	103	103	114	109
Disease	118	13	114	9
Host	2	2	9	8
Location	111	62	110	33
Number of cases	0	0	19	7
Origin	2	2	13	3
Transmission form	6	4	4	2
Conditional	-	-	0	0
Negation	-	-	2	1
Past	-	-	19	13
Uncertainty	0	0	6	4
Total	**342**	**186**	**410**	**189**

Entidades en los títulos.

Entidad	Algoritmo RNN		Algoritmo rule-based	
	Total	**Diferentes**	**Total**	**Diferentes**
Abbreviation	0	0	-	-
Date	109	109	178	140
Disease	655	51	671	18
Host	132	80	162	29
Location	561	273	288	99
Number of cases	513	239	625	225
Origin	141	60	207	39
Transmission form	40	29	18	7
Conditional	-	-	13	1
Negation	9	5	62	8
Past	124	87	146	91
Uncertainty	49	21	85	27
Total	**2333**	**965**	**2455**	**684**

Entidades en los artículos enteros.

Tab. 5.17: Tipos y cantidad de entidades y modificadores encontrados por el algoritmo basado en redes neuronales contra el de reglas para los títulos y los artículos enteros sobre el conjunto de datos de test.

En la tabla 5.18, 5.19 se pueden observar algunos de los términos más frecuentes para ciertas entidades hallados por el algoritmo basado en reglas sobre el conjunto de datos de test de los títulos y los artículos enteros, respectivamente.

Location	**# ocur.**
Brasil	21
Argentina	12
Perú	8
Venezuela	7
Colombia	7
Chile	6
México	5
Puerto Rico	5
Paraguay	4
Bolivia	4
Honduras	2
Ecuador	2
Ica	2
Uruguay	2
Iquitos	2

Date	**# ocur.**
2019	3
2016	2
2015	2
2018	2
17 de junio del 2016	1
24 de febrero del 2018	1
19 de enero	1
mayo de 2017	1
14 de abril de 2004	1
16 de octubre de 2015	1
30 de julio	1
13 de octubre de 2011	1
11 de enero de 2007	1
19 de mayo de 2011	1
05 de abril del 2017	1

Disease	**# ocur.**
zika	27
hantavirus	25
sarampión	24
dengue	20
chagas	8
microcefalia	6
guillain barré	2
chikungunya	1
guillain-barré	1

Host	**# ocur.**
gestantes	2
embarazadas	1
recién nacidos	1
no-vacunados	1
ninos	1
inmigrantes	1
donantes de sangre	1
embarazada	1

Transmission form	**# ocur.**
transmisión oral	2
transmisión vectorial	2

Origin	**# ocur.**
zika	6
vectorial	4
roedores	3

Tab. 5.18: Entidades comúnmente halladas por el algoritmo basado en reglas en los títulos, junto a la cantidad de ocurrencias, sobre el conjunto de datos de test.

Disease	**# ocur.**
dengue	173
zika	155
sarampión	108
hantavirus	97
microcefalia	41
chagas	40
chikungunya	21
guillain barré	9
guillain-barré	8
hanta	6
enfermedad de chagas	3
hanta virus	2
microcefalias	2
hantavirosis	2
dengue hemorrágico	1

Host	**# ocur.**
bebés	31
niños	30
mujeres embarazadas	14
bebé	11
recién nacidos	10
niño	10
niña	8
embarazadas	7
latinoamericanos	5
embarazada	4
gestantes	4
turista	3
viajeros	3
nñas	3
no vacunarse	2

Location	**# ocur.**
Brasil	33
Colombia	16
Argentina	14
Perú	12
Puerto Rico	11
Bolivia	10
Venezuela	9
México	9
Chile	7
Honduras	6
Iquitos	6
Buenos Aires	6
Paraguay	5
Uruguay	5
Costa Rica	5

Origin	**# ocur.**
roedores	37
mosquito	26
zika	21
aedes aegypti	15
mosquitos	13
mosquito _aedes aegypti	10
zancudo	7
vectorial	7
vector	6
mosquito aedes aegypti	5
estrongiloidiasis	5
trypanosoma cruzi	4
t. cruzi	4
insectos	4
roedor	4

Transmission form	**# ocur.**
transmisión oral	4
transmisión vectorial	4
transmisión sexual	4
picadura	2
contaminación	2
picado	1
vía sexual	1

Tab. 5.19: Entidades comúnmente halladas por el algoritmo basado en reglas en los artículos enteros, junto a la cantidad de ocurrencias, sobre el conjunto de datos de test.

En las tablas 5.20 y 5.21 se pueden observar los resultados de *precision*, *recall* y *F1-score* exacto y parcial entre el algoritmo basado en redes neuronales (RNN) y el de reglas (*rule-based)*, tanto para los títulos como para los artículos enteros. Esto se realizó sobre los artículos del conjunto de test utilizado en el algoritmo de redes neuronales profundas. Cabe resaltar que, debido a un error al generar la partición, uno de los artículos del conjunto de test (sobre los 103 totales) se encontraba dentro del conjunto de artículos que se analizaron para la generación del método basado en reglas. Esto constituye un error, ya que no debería haber sido parte del conjunto de test. Por esta razón, los resultados aquí presentados sobre el método de reglas pueden tener una variación respecto al valor real. Estimamos que esta es muy pequeña dado que el porcentaje del conjunto de test que se utilizó para el armado de reglas es solamente del 1 %.

Entidad	Algoritmo RNN			Algoritmo rule-based		
	P	**R**	**F1**	**P**	**R**	**F1**
Abbreviation	0.00	0.00	0.00	-	-	-
Date	0.92	0.96	0.94	0.68	0.78	0.73
Disease	0.91	0.96	0.94	0.96	0.95	0.96
Host	0.50	0.12	0.20	0.78	0.88	0.82
Location	0.68	0.65	0.67	0.64	0.60	0.62
Number of cases	0.00	0.00	0.00	0.11	1.00	0.19
Origin	1.00	0.20	0.33	0.20	0.10	0.13
Transmission form	0.33	0.40	0.36	0.25	0.20	0.22
Conditional	-	-	-	-	-	-
Negation	-	-	-	0.00	0.00	0.00
Past	-	-	-	0.00	0.00	0.00
Uncertainty	0.00	0.00	0.00	0.33	0.67	0.44
Micro-average	**0.83**	**0.79**	**0.81**	**0.70**	**0.75**	**0.73**

Precision, recall y *F1-score* en los títulos.

Entidad	Algoritmo RNN			Algoritmo rule-based		
	P	**R**	**F1**	**P**	**R**	**F1**
Abbreviation	0.00	0.00	0.00	-	-	-
Date	0.88	0.65	0.75	0.48	0.54	0.51
Disease	0.70	0.77	0.73	0.71	0.79	0.75
Host	0.53	0.53	0.53	0.28	0.34	0.31
Location	0.59	0.67	0.63	0.69	0.40	0.51
Number of cases	0.68	0.80	0.74	0.45	0.62	0.52
Origin	0.51	0.43	0.47	0.34	0.38	0.36
Transmission form	0.27	0.22	0.24	0.28	0.10	0.14
Conditional	-	-	-	0.00	0.00	0.00
Negation	0.11	0.09	0.10	0.08	0.45	0.14
Past	0.31	0.43	0.36	0.17	0.21	0.18
Uncertainty	0.39	0.28	0.33	0.19	0.24	0.21
Micro-average	**0.62**	**0.66**	**0.64**	**0.50**	**0.54**	**0.52**

Precision, recall y *F1-score* en los artículos enteros.

Tab. 5.20: Resultados del match exacto del método basado en redes neuronales y el de reglas para los títulos y los artículos enteros sobre el conjunto de datos de test.

Entidad	Algoritmo RNN			Algoritmo rule-based		
	P	**R**	**F1**	**P**	**R**	**F1**
Abbreviation	0.00	0.00	0.00	-	-	-
Date	0.94	0.98	0.96	0.85	0.89	0.87
Disease	0.94	0.98	0.96	0.97	0.97	0.97
Host	0.50	0.12	0.20	0.88	0.88	0.88
Location	0.84	0.84	0.84	0.81	0.73	0.77
Number of cases	0.00	0.00	0.00	0.11	1.00	0.20
Origin	0.40	0.20	0.27	0.11	0.10	0.11
Transmission form	0.67	0.50	0.57	0.67	0.40	0.50
Conditional	-	-	-	0.00	0.00	0.00
Negation	-	-	-	0.00	0.00	0.00
Past	-	-	-	0.00	0.00	0.00
Uncertainty	0.00	0.00	0.00	0.33	0.67	0.44
Micro-average	**0.89**	**0.89**	**0.89**	**0.81**	**0.83**	**0.82**

Precision, recall y *F1-score* en los títulos.

Entidad	Algoritmo RNN			Algoritmo rule-based		
	P	**R**	**F1**	**P**	**R**	**F1**
Abbreviation	0.00	0.00	0.00	-	-	-
Date	0.78	0.68	0.73	0.58	0.65	0.62
Disease	0.73	0.79	0.76	0.73	0.81	0.77
Host	0.59	0.61	0.60	0.37	0.44	0.40
Location	0.64	0.73	0.68	0.77	0.44	0.56
Number of cases	0.69	0.81	0.75	0.46	0.63	0.53
Origin	0.54	0.47	0.50	0.43	0.47	0.45
Transmission form	0.41	0.38	0.40	0.30	0.17	0.21
Conditional	-	-	-	0.00	0.00	0.00
Negation	0.12	0.09	0.11	0.10	0.60	0.17
Past	0.46	0.53	0.49	0.24	0.29	0.26
Uncertainty	0.50	0.38	0.43	0.28	0.35	0.31
Micro-average	**0.66**	**0.70**	**0.68**	**0.54**	**0.58**	**0.56**

Precision, recall y *F1-score* en los artículos enteros.

Tab. 5.21: Resultados del match parcial del método basado en redes neuronales y el de reglas para los títulos y los artículos enteros sobre el conjunto de datos de test.

Al comparar los resultados en los títulos, el algoritmo de redes neuronales tuvo mejores resultados de *F1 micro-average* que el basado en reglas, tanto para el título (81 % contra 73 % en match exacto, y 89 % contra 82 % en match parcial) como para los artículos enteros (64 % contra 52 % en match exacto, y 68 % contra 56 % en match parcial).

Algunas observaciones sobre los resultados sobre los títulos son:

- **Number of cases:** el algoritmo de reglas halló 19 entidades contra las 0 del de redes neuronales. Si bien este encontró todas las que debía (*recall* de 100 %), etiquetó muchos términos erróneos (*precision* de 11 %). Aun así, tuvo mejores resultados que

el de redes neuronales profundas.

- **Date:** el algoritmo de reglas tuvo un *F1* exacto de 73 % contra el 94 % del algoritmo de RNN (21 % de diferencia). Una de las razones de dicha diferencia se debe al problema de que las mismas no se encuentran escritas en un formato estándar. En este caso, *Freeling* detecta múltiples fechas (cuando se encuentra separadas por comas, por ejemplo) en vez de una sola. Si se observa el match parcial, la diferencia fue menor, con 87 % del de reglas contra 96 % (9 % de diferencia).

- **Location:** el algoritmo de RNN tuvo un *F1* exacto de 67 % contra el 62 % del de reglas, y parcial de 84 % contra 77 %. Una de las razones por las cuales es mejor se debe a que es capaz de detectar aquellas ubicaciones que contienen entre paréntesis una ciudad o región (por ejemplo, *"Brasil (Sao Paulo)"*) como una sola entidad, a diferencia del algoritmo de reglas que detecta: *Brasil* (LOC) y *'Sao Paulo'* (LOC). Esto genera que el algoritmo de reglas detecte más instancias de países, a diferencia del de reglas que detecta el país junto con la región o ciudad correspondiente. Esto puede observarse en la figura 5.12.

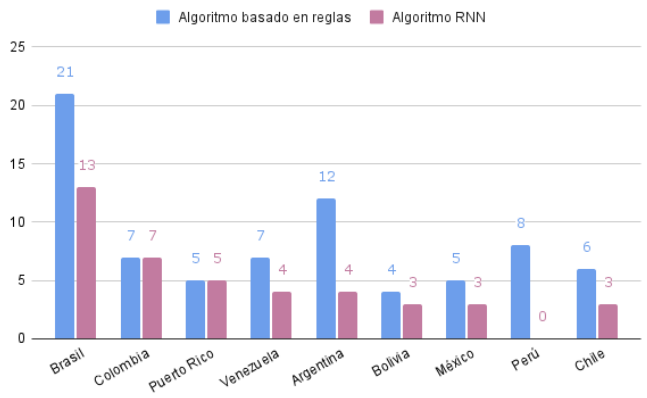


Fig. 5.12: Términos de ubicaciones geográficas comúnmente halladas en los títulos por ambos algoritmos.

- **Conditional, Negation y Past:** para estas entidades, el algoritmo de RNN no tuvo ocurrencias en el conjunto de entrenamiento, por lo cual no fue capaz de detectar ninguna. En cambio, el algoritmo de reglas sí fue capaz de detectar algunas, aunque todas incorrectas.

- **Origin:** Si bien el algoritmo de redes neuronales etiquetó solamente 2 entidades de este tipo, ambas son correctas (con un *precision* de 100 %). El de reglas, por el contrario, etiquetó 13, pero mayoritariamente incorrectas (*precision* de 20 %). En cuanto al valor de *recall* y *F1*, el primero de ellos obtuvo mejores resultados.

- **Host, Transmission form y Uncertainty:** Dichas entidades fueron poco frecuentes en los títulos (esto puede observarse en la tabla 5.9 en base a la cantidad de términos etiquetados por los anotadores). Para las mismas, ambos algoritmos tuvieron resultados, tanto de *precision*, *recall* y *F1*, por debajo de los *micro-averages*.

- **Disease:** Ambos algoritmos obtuvieron los mejores resultados de *precision*, *recall* y *F1*. Si bien el de reglas obtuvo resultados un poco mejores que el de redes neuronales, la diferencia es baja.

- En cuanto a términos frecuentes de ciertas entidades, se encontraron ocurrencias similares entre distintos términos, como se puede observar en la figura 5.13.

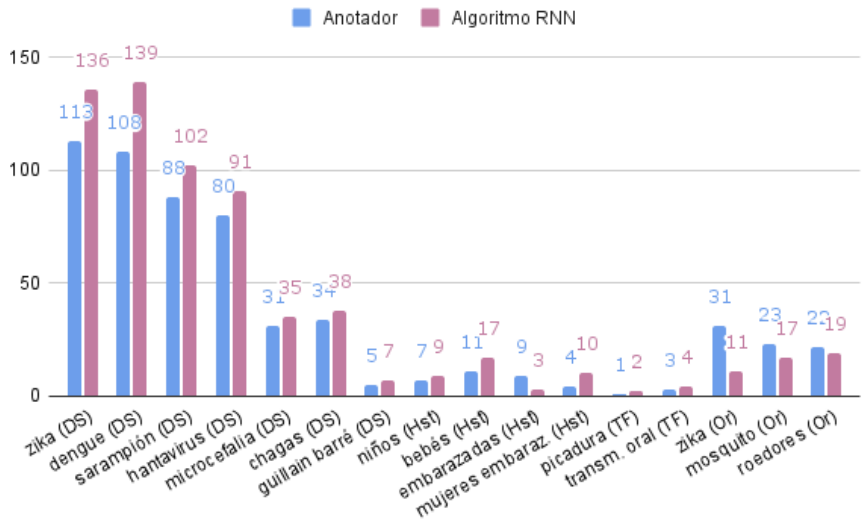


Fig. 5.13: Términos frecuentes hallados en los títulos entre el algoritmo basado en reglas y el de redes neuronales.

En cuanto a los artículos enteros, se observó lo siguiente:

- **Number of cases:** El algoritmo de RNN fue capaz de detectar mejor dicha entidad (*F1* exacto de 74 % contra 52 % del de reglas), reconociendo entidades como *"10 casos / 100 mil habitantes"*, caso que el de reglas no fue capaz de detectar ya que solo considera términos numéricos.

- **Negation:** Si bien para ambos algoritmos el *F1* fue bajo, el algoritmo de reglas fue capaz de detectar más de las que debía, con un valor de *recall* exacto de 45 % contra el 9 % del de redes, y parcial de 60 % contra 11 %.

- **Disease:** si bien el de reglas tuvo mejores resultados (*F1* exacto de 75 % contra 73 % del de RNN, y parcial de 77 % contra 76 %), la diferencia es ínfima, ofreciendo ambos algoritmos buenos resultados. Algo a destacar del algoritmo de RNN, es que al basarse en una mayor cantidad de artículos para su entrenamiento, es capaz de detectar más variaciones para referirse a una misma enfermedad o reconocer aquellas con errores ortográficos. Por el lado contrario, el de redes neuronales arrastra los errores del corpus anotado (por ejemplo, detecta enfermedades erróneas como *sífilis*).

- Para la mayor parte de entidades en las que el algoritmo de reglas solo utilizó expresiones regulares, el de RNN obtuvo mejores resultados, con *F1* exactos respectivos de: *Host* (53 % contra 31 %), *Origin* (47 % contra 36 %), *Past* (36 % contra 18 %)

y *Uncertainty* (33 % contra 21 %). Esto se debe a que el algoritmo de RNN se entrenó con una mayor cantidad de artículos que aquellos considerados para crear las expresiones regulares utilizadas en el algoritmo de reglas.

- Para aquellas entidades en las cuales el algoritmo de reglas tenía reglas más complejas, el de RNN igualmente tuvo mejores resultados, con matches exactos de: *Date* (51 % del de reglas contra 75 %), *Location* (51 % contra 63 %) y *Number of cases* (52 % contra 74 %).
- En cuanto a términos frecuentes hallados para ciertas entidades, se puede observar que el algoritmo de reglas encontró más ocurrencias que el de RNN. Pueden observarse algunos ejemplos de estos términos en la figura 5.14.

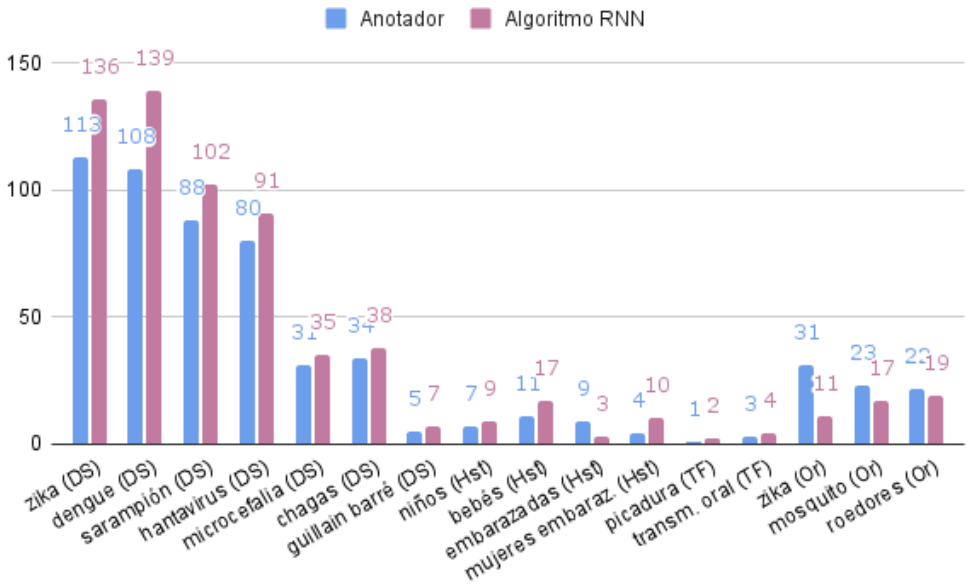


Fig. 5.14: Términos frecuentes hallados en los artículos enteros por ambos algoritmos.

Como conclusión de ambos métodos, en general los resultados del algoritmo de redes neuronales fueron mejores. Si bien el costo de generar el corpus anotado y el entrenamiento son altos (en tiempos y recursos), generaliza mejor ya que utiliza más artículos para entrenar y es más fácil de adaptar a nueva información, ya que basta solamente entrenar el algoritmo con los nuevos datos. Además, muchas de las entidades etiquetadas erróneamente se debieron a inconsistencias en las anotaciones, por lo que si se corrigiesen se obtendrían mejores resultados. En cambio, el algoritmo de reglas requiere crearlas manualmente en base a artículos, y se pierde información ya que no se es capaz de utilizar la misma cantidad que las que utiliza el algoritmo de RNN para el entrenamiento, y no es fácil de adaptar, por lo que si aparecen nuevos términos (por ejemplo, nuevas enfermedades) hay que agregar nuevas reglas manualmente. Una desventaja del algoritmo de RNN es que depende de las entidades que se encuentren en el conjunto de entrenamiento y, para el caso de los títulos, no hubo instancias de los modificadores de tipo *Conditional*, *Past* y *Negation*.

6. EXTRACCIÓN DE RELACIONES

En este capítulo presentamos el trabajo realizado para la extracción de relaciones. Iniciamos con una introducción sobre el tema y trabajos previos. Luego, hablamos de medidas de calidad. Finalmente, explicamos el algoritmo basado en co-ocurrencia desarrollado junto con los resultados obtenidos.

6.1. Introducción

La extracción de relaciones es una tarea dentro del área de extracción de información que consiste en reconocer relaciones semánticas en un texto entre dos o más entidades. Supongamos que tenemos el texto *"Se detectaron 15 casos de hantavirus"*, en el cual se tienen las entidades nombradas *15* (NoC) y *hantavirus* (Disease); luego, se podría establecer una relación de tipo *"casos de enfermedad"(CasesOfDis)* entre *15* y *hantavirus* (Ver figura 6.1).

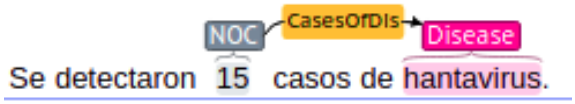


Fig. 6.1: Ejemplo de entidades nombradas (NoC y Disease) con una relación entre ellas (CasesOfDis).

El resto del capítulo está organizado de la siguiente manera. En la sección 6.2, mencionamos trabajos previos sobre extracción de relaciones. Luego, en la sección 6.3, presentamos la medida de calidad utilizada para medir el desempeño de la tarea. Finalmente, en la sección 6.4, desarrollamos el algoritmo basado en co-ocurrencia desarrollado, presentando el método, los resultados y el análisis de los resultados.

6.2. Trabajos previos

Una de las formas de realizar la tarea de extracción de relaciones es utilizando **métodos basados en reglas**. Por ejemplo, se podrían establecer patrones hechos a mano de la forma de tuplas (X, α, Y) siendo X e Y entidades, y α las palabras entre ambas entidades. Para evitar falsos positivos, se podría requerir que X e Y pertenezcan a ciertas entidades, y también se podría utilizar el *PoS-tagging* para realizar otro tipo de filtrado (por ejemplo, que sean sustantivos). Además, se podrían utilizar caminos de dependencias[50] para generar reglas. Sin embargo, estos enfoques conllevan mucho trabajo manual para definir las reglas (teniendo una para cada tipo de relación que puede existir) y varían según el idioma y el dominio del problema.

Otro tipo de formas de realizar RE es utilizando **métodos débilmente supervisados**, que inician con un conjunto de datos etiquetados de ejemplos de relaciones que se utilizan para crear automáticamente patrones de extracción, que a la vez extraen más

[50] Tarea que consiste en extraer dependencias gramaticales entre las palabras de una oración.

ejemplos de relaciones. Esto se realiza a través de un proceso iterativo. Un algoritmo utilizado de este tipo es el *Snowball* [1]. Si bien este método puede encontrar más relaciones que el anterior y requiere menos esfuerzo, nuevos tipos de relaciones que se quieran extraer requieren un nuevo conjunto de ejemplos.

Una forma más común de realizar RE es a través de **métodos supervisados**, que consisten en entrenar un clasificador binario que determina si existe una relación entre dos o más entidades. Este tipo de enfoque requiere que se cuente con datos con etiquetas de entidades y relaciones, que serán utilizados por el clasificador para extraer características de los mismos y determinar el tipo de relación. Un trabajo realizado en este tipo de enfoque es el de Zhou y col. [73]. El problema de este tipo de enfoques es que es costoso proveer los ejemplos etiquetados y para agregar nuevas relaciones hay que agregar nuevos datos.

Otro enfoque es utilizando **métodos no supervisados**, que se basan en extraer relaciones del texto sin la necesidad de tener datos de entrenamiento o reglas hechas a mano, basándose en una serie de restricciones generales y heurísticas. Un ejemplo en este tipo de algoritmos es el llamado *TextRunner* [71], el cual utiliza rasgos como el *PoS-tag* de las palabras para evaluar si dos términos pueden estar relacionados o no. Algunos sistemas que utilizan este enfoque son el *Stanford OpenIE*[51] y el *OpenIE 5.0*[52]. Si bien no requieren datos de entrenamiento y no hay que especificar ninguna relación de interés ya que considera todas las posibles, el desempeño del sistema depende mucho de las heurísticas y restricciones establecidas.

También existen los **métodos supervisados a distancia**, que combinan métodos supervisados con otros enfoques para extraer relaciones. Por ejemplo, en el trabajo de Mintz y col. [48], utilizan una base de datos que contiene miles de ejemplos de relaciones, y a través de métodos no supervisados extraen de un corpus grande sin etiquetar relaciones entre entidades, que son utilizados posteriormente para entrenar un clasificador (usando la técnica de métodos supervisados). Si bien este método requiere menos esfuerzo manual, depende de una base de datos con ejemplos de las relaciones.

6.3. Medidas de calidad

Para evaluar el desempeño del algoritmo, se utilizó la medida F1-score. Se implementó un algoritmo que compara las relaciones (binarias y ternarias) anotadas por el anotador y el algoritmo a través del cálculo de los TP, TN, FN y FP. Posteriormente, se calcula la precision, recall y F1-score con los mismos.

El algoritmo se basa en identificar, por cada oración, los pares de entidades que pueden estar relacionadas, guardando las tuplas de entidades que existen con dicho tipo de relación y el orden en el que se encuentran ya que indican el sentido de la misma. Por ejemplo, si tenemos la entidad de tipo *Disease* con identificador $T1$ y la entidad de tipo *Number of cases* con identificador $T2$, tenemos la tupla $(T2, T1)$ para la relación *cases of disease*, indicando que dicha relación ocurre desde la entidad T2 hacia la entidad T1. Luego, supongamos que tenemos en la oración i la lista de relaciones de tipo R realizadas por el anotador (que llamaremos $anotador_i$) y sean $pred_i$ las relaciones detectadas por el algoritmo en la oración i. Entonces, realizamos lo siguiente:

- Los TP están constituidos por las tuplas que se encuentran en $pred_i$ y también se encuentran en $anotador_i$.

[51] `https://nlp.stanford.edu/software/openie.html` [Accedido en Abril de 2020]
[52] `https://github.com/dair-iitd/OpenIE-standalone`. [Accedido en Abril de 2020]

- Los FN están constituidos por las tuplas que se encuentran en $anotador_i$ pero no en $pred_i$.
- Los FP están constituidos por las tuplas que se encuentran en $pred_i$ pero no en $anotador_i$.
- Los TN están constituidos por cada tupla de entidades $(t_k, ..., t_j)$ ($j = k + 2$ en caso de R relación binaria y $j = k + 1$ en caso de relación ternaria) que se encuentran en la oración i tal que puedan tener una relación de tipo R entre ellas y que no se encuentra en $anotador_i$ ni en $pred_i$.

Luego, para cada relación, se computan los valores de *precision*, *recall F1-score micro-average* en base a los valores de TP, FN, FP y TN.

En el algoritmo 3 se puede observar lo realizado para calcular las métricas generales para el conjunto de anotaciones del anotador y el algoritmo para un artículo particular, y en el algoritmo 4 se puede observar lo realizado para el cálculo de las métricas para una relación particular.

Algorithm 3: Algoritmo para calcular F1-score sobre todas las relaciones.

```
overall_tp, overall_fn, overall_fp, overall_tn = 0, 0, 0, 0
metricas_relaciones = diccionario() ;
for relación ← relaciones do
    tp, fn, fp, tn = obtener_metricas_por_relación(relación);
    metricas_relaciones[relacion] = (tp, fn, fp, tn);
    overall_tp += tp;
    overall_fn += fn;
    overall_fp += fp;
    overall_tn += fn;
end
precision = overall_tp / (overall_tp+overall_fp) ;
recall = overall_tp / (overall_tp+overall_fn) ;
f1 = 2 * precision*recall / (precision+recall) ;
Output: precision, recall, f1, metricas_relaciones
```

Algorithm 4: obtener_metricas_por_relación(R) - Algoritmo para calcular F1-score sobre la relación R.

Input: R = relación actual
anotador = relaciones anotadas por el anotador por oración;
pred = relaciones anotadas por el algoritmo por oración;
tp, tn, fp, fn = 0, 0, 0, 0;
for *oración* ← *oraciones* **do**
  relaciones_anotador = anotador[oracion][R];
  relaciones_pred = pred[oracion][R];
  for $k \leftarrow$ *relaciones_pred* **do**
    if $k \in$ *relaciones_anotador* **then**
      tp++
    end
  end
  for $k \leftarrow$ *relaciones_anotador* **do**
    if $k \notin$ *relaciones_pred* **then**
      fn++
    end
  end
  for $k \leftarrow$ *relaciones_pred* **do**
    if $k \notin$ *relaciones_anotador* **then**
      fp++
    end
  end
  for ent_i, $ent_j \leftarrow$ *entidades* **do**
    if $ent_i \neq ent_j$ ***and*** $ent_i \in$ *oración* ***and*** $ent_j \in$ *oración* ***and***
     $\exists r \in relaciones : r == R$ ***and*** $(ent_i, ent_j) \notin relaciones\_anotador$ ***and***
     $(ent_i, ent_j) \notin relaciones\_pred$ **then**
      tn++
    end
  end
end
Output: tp, tn, fp, fn

6.4. Método basado en co-ocurrencia

En esta sección se desarrolla el método basado en co-ocurrencia para la detección de relaciones.

6.4.1. Explicación del método

Se implementó un algoritmo basado en la co-ocurrencia de entidades para detectar relaciones binarias y ternarias de manera automática, con algunas reglas adicionales. Este algoritmo puede ser utilizado como baseline para comparar sus resultados con otro más sofisticado. Nos basamos en detectar las relaciones entre las entidades anotadas manualmente por los anotadores dentro de una misma oración, es decir que no se anotan relaciones entre entidades que pertenecen a distintas oraciones. Para saber qué entidades pueden estar relacionadas con otras, se generó un diccionario el cual contiene, para cada entidad,

cuáles son las entidades con las cuales puede estar relacionada (según el criterio de anotación), junto con el nombre de dicha relación (esto se realizó tanto para entidades binarias como ternarias). Luego, se itera sobre todas las entidades anotadas en cada oración y se evalúa si existe, en la misma oración, otra entidad para la cual existe una relación en el criterio de anotación entre ambas entidades. En caso de que así sea, se considera dicha relación como posible, que será agregada a la lista si cumple además alguna de las reglas propuestas en el siguiente párrafo.

En base a un subconjunto de artículos analizados, se generaron reglas adicionales para la detección de relaciones binarias[53], además de la co-ocurrencia de entidades. Estas fueron:

- Se notó que aquellas entidades que se encontraban a grandes distancias no solían estar relacionadas. Por esta razón, se estableció una distancia máxima entre las entidades a relacionar y, en caso de excederlas, no se establece la relación. Las distancias se establecieron en base a los siguientes casos:
  - Si una de las entidades es un modificador (*Negation*, *Conditional*, *Past* o *Uncertainty*), luego la distancia máxima entre ambas entidades es de 50 caracteres. En base al análisis del corpus anotado (ver sección 4.5), la longitud promedio de las palabras es de cinco caracteres, por lo que 50 caracteres equivale a aproximadamente diez palabras.
  - Si ninguna de las entidades pertenece a un modificador (por ejemplo, una entidad de tipo *Location* con una de tipo *Disease*), luego la distancia máxima es de 100 caracteres, lo que equivale a aproximadamente veinte palabras.
- No establecer relaciones entre la misma entidad. Existen casos en los que un término posee múltiples etiquetas (por ejemplo, *zika* puede ser origen y enfermedad al mismo tiempo) y, en base a las posibles relaciones que poseemos, no es correcto que se relacione un término consigo mismo.

6.4.2. Resultados

A continuación se presentan una serie de tablas con los resultados del algoritmo sobre todo el corpus anotado, tanto para los títulos como para los artículos enteros. Inicialmente, se muestra la cantidad total y diferente de términos hallados por relación. Luego, ejemplos de los términos más frecuentes hallados para ciertas relaciones. Finalmente, los resultados de *precision*, *recall* y *F1-score*.

En las tablas 6.1 y 6.2 se pueden observar las relaciones y cantidad de ocurrencias por relación, de las relaciones extraídas por el algoritmo y los anotadores sobre los títulos y los artículos enteros.

[53] En el caso de las relaciones ternarias, solo nos basamos en la co-ocurrencia dentro de la misma oración.

Relación	Entidades	Algoritmo		Anotadores	
		Total	**Diferentes**	**Total**	**Diferentes**
DsOccIn	DS-LOC	620	278	573	254
NegDs	NT-DS	3	2	1	1
NegLoc	NT-LOC	3	3	-	-
NegOr	NT-OR	2	1	2	1
NoCDs	NoC-DS	10	6	8	5
NoCLoc	NoC-LOC	8	8	4	4
OccTo	DS-HT	40	29	36	26
OrDs	OR-DS	48	29	38	25
OrOccIn	OR-LOC	50	37	14	13
TfDs	TF-DS	31	17	14	8
TfOccIn	TF-LOC	30	25	7	6
TfOr	TF-OR	11	9	8	6
UcDs	UT-DS	18	13	6	5
UcNoC	UT-NoC	2	2	-	-
UcOr	UT-OR	7	7	5	5
UcTf	UT-TF	7	5	4	4
UcOrDs	UcTfOr-DS-OR	4	4	4	4
UcTfDs	UcTfDs-DS-TF	4	3	3	3
Total		**898**	**478**	**727**	**370**

Tab. 6.1: Relaciones y ocurrencias encontradas por el algoritmo extractor de relaciones y las de los anotadores en los títulos. En la segunda columna se mencionan las entidades involucradas en la relación mencionada en la primera columna.

Relación	Entidades	Algoritmo		Anotadores	
		Total	Diferentes	Total	Diferentes
DsOccIn	DS-LOC	1876	912	1838	924
DtDs	DT-DS	87	70	89	75
NegDs	NT-DS	49	43	30	28
NegLoc	NT-LOC	31	30	1	1
NegNoC	NT-NoC	63	59	28	26
NegOr	NT-OR	24	22	18	16
NegTf	NT-TF	25	24	13	12
NoCDs	NoC-DS	1330	922	1164	817
NoCHt	NoC-HT	469	422	214	198
NoCLoc	NoC-LOC	2343	1925	810	773
OccTo	DS-HT	433	254	408	254
OrDs	OR-DS	597	316	596	300
OrOccIn	OR-LOC	198	152	66	60
PtDs	PT-DS	322	285	203	189
PtNoC	PT-NoC	606	577	389	379
TfDs	TF-DS	172	123	105	77
TfOccIn	TF-LOC	83	76	18	1
TfOr	TF-OR	179	163	149	134
UcDs	UT-DS	303	219	89	75
UcNoC	UT-NoC	478	419	271	249
UcOr	UT-OR	88	83	58	54
UcTf	UT-TF	24	21	18	17
NegTfOr	NegTfOr-TF-OR	7	7	3	3
UcNoCDs	UcNoCDs-NoC-DS	98	91	60	59
UcOrDs	UcTfOr-DS-OR	44	43	36	36
UcTfDs	UcTfDs-DS-TF	6	5	5	5
Total		**9935**	**7263**	**6679**	**4778**

Tab. 6.2: Relaciones y ocurrencias encontradas por el algoritmo extractor de relaciones y las de los anotadores en los artículos enteros. En la segunda columna se mencionan las entidades involucradas en la relación mencionada en la primera columna.

En las tablas 6.3 y 6.4 se pueden observar ejemplos de algunos de los términos más frecuentes de las entidades involucradas en distintas relaciones, tanto para los títulos como para los artículos enteros, respectivamente.

OrDs	**# ocur.**
zika-microcefalia	11
no-vacunados-sarampión	4
vectores-chagas	2
acai contaminado-chagas	2
importado-sarampión	2
acai-chagas	2
brote-hantavirus	2
mosquitos-zika	2
vectorial-chagas	1
brote familiar-chagas	1

TfDs	**# ocur.**
transmisión oral-chagas	8
vía oral-chagas	5
vectorial-chagas	3
transmisión interpersonal-hantavirus	2
adultos y machos-hantavirus	1
brote oral-chagas	1
transmisión sexual-zika	1
transmisión vertical-microcefalia	1
transmisión vertical-zika	1
trasmisión sexual-zika	1

OccTo	**# ocur.**
sarampion-niños	6
zika-embarazadas	5
hantavirus-embarazada	2
zika-adultos	2
zika-recién nacidos	1
zika-embarazada	1
microcefalia-embarazada	1
microcefalia-niños	1
dengue-embarazada	1
hantavirus-trabajadores	1

TfOr	**# ocur.**
transmisión oral-acai contaminado	2
transmisión interpersonal-brote	2
transmisión oral-brote familiar	1
brote oral-vino de acai artesanal	1
transmisión vertical-zika	1
trasmisión oral-acai	1
consumo-acai	1
glándulas salivales-mosquitos	1
vía oral-acai	1

Tab. 6.3: Relaciones comúnmente halladas en los títulos por el algoritmo, junto a la cantidad de ocurrencias.

OrDs	# ocur.
zika-microcefalia	69
aedes aegypti-dengue	21
mosquito-dengue	20
mosquito-zika	17
mosquito _aedes aegypti_-zika	13
aedes aegypti-zika	12
mosquito _aedes aegypti_-dengue	10
zika-guillain-barré	10
mosquitos-zika	10
mosquito _aedes aegypti_-chikungunya	8
mosquito-enfermedad	7
roedores-hantavirus	7
mosquito aedes aegypti-dengue	6
mosquitos-dengue	6
virus-microcefalia	5

TfDs	# ocur.
transmisión oral-chagas	10
via oral-chagas	8
vectorial-chagas	6
transmisión sexual-zika	6
congénito-chagas	5
picadura-dengue	4
picadura-zika	4
transmisión directa de sus madres-chagas	2
ingestión-chagas	2
consumo-enfermedad	2
relaciones sexuales-zika	2
consumo-chagas	2
trasmisión oral-chagas	2
ratones silvestres-hantavirus	2
transmisión interpersonal-hantavirus	2

OccTo	# ocur.
microcefalia-bebés	24
zika-bebés	20
zika-embarazadas	19
microcefalia-recién nacidos	14
zika-mujeres embarazadas	13
microcefalia-niños	13
sarampión-niños	12
microcefalia-bebé	9
zika-madres	6
zika-niños	6
zika-adultos	5
zika-fetos	5
microcefalia-fetos	5
zika-recién nacidos	4
chagas-niños	3

TfOr	# ocur.
picadura-mosquito	6
picadura-_aedes aegypti_	4
mosquitos-zika	2
transmisión oral-acai contaminado	2
orina-roedores	2
congenitos-zika	2
ingestión-jugo de caña de azucar	2
transmisión interpersonal-brote	2
glandulas salivales-mosquitos	2
picadura-vector	2
penetra en el organismo humano a traves de las vías respiratorias o de heridas-raton	1
picado-mosquito	1
transmisión oral-brote familiar	1
vía sexual-zika	1
respirar-lugares muy cerrados como galpones	1

Tab. 6.4: Relaciones comúnmente halladas en los artículos enteros por el algoritmo, junto a la cantidad de ocurrencias.

En la tabla 6.5 se pueden observar los resultados de *precision*, *recall* y *F1-score* exacto de las relaciones (binarias y ternarias) para los títulos y los artículos enteros entre las relaciones extraídas por el algoritmo contra las anotadas manualmente.

Relación	**P**	**R**	**F1**
DsOccIn	0.92	1.00	0.96
NegDs	0.33	1.00	0.50
NegLoc	0.00	0.00	0.00
NegOr	1.00	1.00	1.00
NoCDs	0.80	1.00	0.89
NoCLoc	0.50	1.00	0.67
OccTo	0.90	1.00	0.95
OrDs	0.79	1.00	0.88
OrOccIn	0.28	1.00	0.44
TfDs	0.45	1.00	0.62
TfOccIn	0.23	1.00	0.38
TfOr	0.73	1.00	0.84
UcDs	0.33	1.00	0.50
UcNoC	0.00	0.00	0.00
UcOr	0.71	0.83	0.77
UcTf	0.43	1.00	0.60
UcOrDs	0.25	0.25	0.25
UcTfDs	0.50	0.67	0.57
Micro-avg	**0.80**	**0.99**	**0.89**

Resultados en los títulos.

Relación	**P**	**R**	**F1**
DsOccIn	0.87	0.89	0.88
DtDs	0.87	0.80	0.84
NegDs	0.59	0.97	0.73
NegLoc	0.03	1.00	0.06
NegNoC	0.43	1.00	0.60
NegOr	0.71	0.94	0.81
NegTf	0.52	1.00	0.68
NoCDs	0.79	0.90	0.84
NoCHt	0.44	0.97	0.61
NoCLoc	0.32	0.92	0.47
OccTo	0.83	0.88	0.86
OrDs	0.85	0.85	0.85
OrOccIn	0.26	0.78	0.39
PtDs	0.56	0.92	0.70
PtNoC	0.60	0.93	0.73
TfDs	0.51	0.84	0.64
TfOccIn	0.22	1.00	0.36
TfOr	0.78	0.93	0.85
UcDs	0.27	0.94	0.42
UcNoC	0.55	0.97	0.70
UcOr	0.62	0.95	0.75
UcTf	0.67	0.89	0.76
NegTfOr	0.29	0.67	0.40
UcNoCDs	0.30	0.48	0.37
UcOrDs	0.23	0.29	0.25
UcTfDs	0.50	0.60	0.55
Micro-avg	**0.60**	**0.89**	**0.72**

Resultados en los artículos enteros.

Tab. 6.5: *Precision*, *recall* y *F1-score exacto micro averages* del algoritmo extractor de relaciones.

6.4.3. Análisis de los resultados

Como se mencionó anteriormente, este algoritmo solo se encarga de encontrar relaciones y no entidades. Por lo tanto, los resultados solo se enfocan en evaluar las relaciones halladas por el algoritmo contra las anotadas manualmente.

Si se comparan las cantidades de relaciones halladas por los anotadores contra las del algoritmo (ver tablas 6.1 y 6.2), podemos observar que el algoritmo halló en total más que el anotador: 898 contra 727 en los títulos, y 9935 contra 4778, información que debe complementarse con los resultados para evaluar si las que encuentra son correctas o no.

Algunas observaciones sobre los resultados son:

- En cuanto a los términos de las entidades involucradas en las relaciones *OrDs*, *TfDs*, *OccTo* y *TfOr* encontrados por los anotadores (ver tablas 4.11 y 4.12) contra las del algoritmo (ver tablas 6.3 y 6.4), se pueden notar similitudes. Por ejemplo, en la figura 6.2 se pueden observar ejemplos de los términos frecuentes en los títulos y en la figura 6.3 en los artículos enteros.

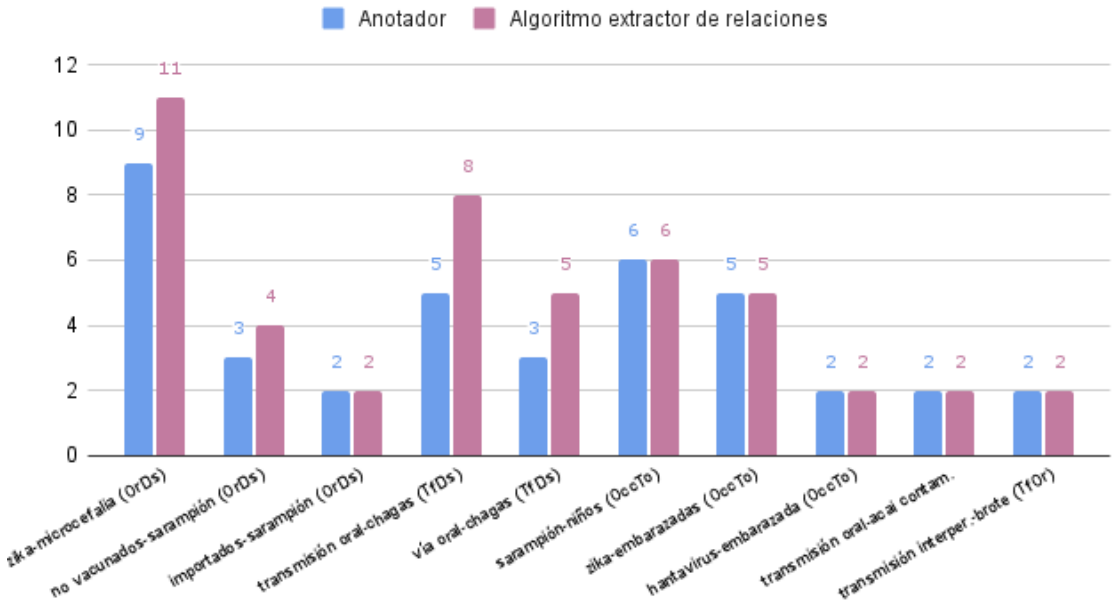


Fig. 6.2: Relaciones frecuentes halladas en los títulos entre los anotadores y el algoritmo extractor de relaciones.

- En los títulos se obtuvo un *recall micro average* de 99 %, con valores específicos de 99 % y 100 % para gran parte de las relaciones, lo cual refleja que el algoritmo fue capaz de detectar casi todas las relaciones que debía. Sin embargo, los valores de *precision* demuestran que tiende a detectar relaciones erróneas, con un *micro average* de 80 %.
- Similar a los títulos, en los artículos enteros se puede observar que el algoritmo es mejor detectando las relaciones que debe (con un *recall micro average* de 89 %), pero tiende a detectar relaciones erróneas (con un *precision micro average* de 60 %).
- **Number of cases:** Para las relaciones que involucran cantidad de casos (salvo cuando se relaciona con la enfermedad), hubo más ocurrencias encontradas por el algoritmo que por el anotador (por ejemplo, en el artículo entero, para *NoCLoc* se encuentran 2343 contra 810 y para *NoCHt* tenemos 469 contra 198). Según los valores de *recall*, se puede observar que encuentra la mayoría de los que debe (92 % para *NoCLoc* y 97 % para *NoCHt*), pero el problema es que detecta muchas relaciones erróneas, lo cual puede verse en los valores de *precision* (32 % para *NoCLoc* y 44 % para *NoCHt*).
- **NegLoc:** se obtuvo un *F1* de 6 % en los artículos enteros. El anotador encontró solamente 1 relación de ese tipo, mientras que el algoritmo 31 (siendo capaz de

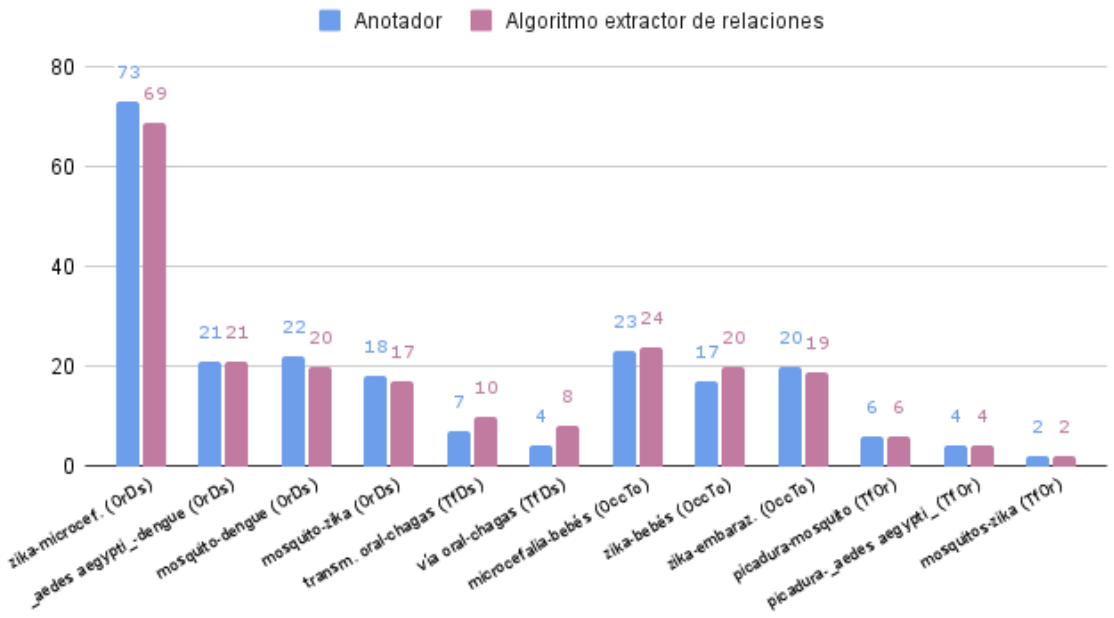


Fig. 6.3: Relaciones frecuentes halladas en los artículos enteros entre los anotadores y el algoritmo extractor de relaciones.

detectar la única correcta, puesto que el valor de *recall* es de 100 %).

- **TfOccIn y OrOccIn:** Los resultados de dichas relaciones se encontraron por debajo del *F1 micro average*, con un 36 % en los artículos enteros y 38 % en los títulos para *TfOccIn*, y 39 % en los artículos enteros y 44 % en los títulos para *OrOccIn*. También, podemos observar que en los artículos enteros, el anotador encontró 18 relaciones de tipo *TfOccIn*, mientras que el algoritmo halló 83 (con un *recall* de 100 %); y para *OrOccIn* tenemos 66 contra 298 (con un*recall* de 78 %). Luego, la co-ocurrencia de entidades para estas relaciones no es suficiente ya que el algoritmo encuentra muchas erróneas. Esto podría implicar que deberían generarse nuevas reglas para su detección.

- **DsOccIn, OccTo y TfOr:** Estas relaciones tuvueron resultados por encima del *F1 micro average*: *DsOccIn* con 88 % en los artículos enteros y 96 % en los títulos, *OccTo* con 86 % y 95 %, y *TfOr* con 85 % y 84 %.

Como conclusión, en general, el algoritmo tuvo mejores valores de *recall* que de *precision*, por lo que es capaz de encontrar las relaciones que debe, aunque encuentra muchas de más. Existen diversos tipos de relaciones para las cuales el algoritmo implementado ofreció altos valores de *F1*, por lo que este algoritmo sencillo es una buena aproximación. En cambio, existen otros casos, como *TfOccIn* y *OrOccIn* para los cuales no fue suficiente, por lo que se podrían considerar generar otro tipo de reglas. Además, dado que poseemos un corpus anotado con relaciones, se podrían entrenar algoritmos de redes neuronales para realizar esta tarea, o generar un método híbrido

7. CONCLUSIONES Y TRABAJO A FUTURO

Este trabajo tenía como objetivo generar herramientas que permitan ayudar, en adición a la información brindada por los sistemas de vigilancia, en la detección temprana de brotes de enfermedades para tomar las medidas necesarias con el fin de disminuir su propagación. Para realizar esto, debíamos contar inicialmente con información sobre brotes de enfermedades prevalentes en Latinoamérica y, a falta del mismo, decidimos generar un corpus anotado. Una vez creado, se pudo proceder a extraer información de interés a través de algoritmos para reconocer entidades nombradas y relaciones.

En cuanto a la información sobre brotes de enfermedades, se generó un corpus anotado basado en 513 artículos periodísticos de ProMED-mail. Este puede ser utilizado para contestar una serie de preguntas que conciernen a temas de salud, ayudando a entender el curso de epidemias en desarrollo. Esta etapa del trabajo fue demandante en cuanto a tiempo y recursos, requiriendo la preparación del corpus (selección y descarga de los artículos, seguido de una limpieza y normalización de los textos), la definición del esquema y los criterios de anotación, y la anotación. Se realizó un proceso iterativo de definición-anotación-evaluación-revisión con tres tandas de anotaciones en las que se anotaron artículos superpuestos para evaluar la consistencia entre las anotaciones. Si bien se intentó definir de manera precisa el criterio de anotación, se encontraron ciertos problemas:

- Entidades difíciles de definir, como *Origin* y *Transmission form*, las cuales fueron confundidas entre sí.
- Dificultad al establecer los términos abarcados por las entidades pese a encontrarse definido en los criterios. Por ejemplo, ante el texto *'mujeres embarazadas'*, se encontraron las anotaciones: *'embarazadas'* y *'mujeres embarazadas'*.
- Términos erróneos anotados, como enfermedades fuera de la lista de enfermedades de interés.
- Términos anotados como *Date* en vez de *Past*.

Si bien estos no son todos los problemas que se hallaron, son suficientes para demostrar la dificultad de la tarea de anotación y, pese a que el criterio de anotación se fue redefiniendo, no fue fácil de comprender por todos los anotadores, habiendo muchas inconsistencias entre las anotaciones realizadas.

Siete tandas de anotaciones fueron llevadas a cabo por ocho personas, quienes eran hablantes nativas de español peruano (variante andino y limeño) y español argentino (variante rioplatense). Se contó con un lingüista, cinco estudiantes de máster en ciencias de la computación y dos doctores en ciencias de la computación que son, a la vez, investigadores en el área de procesamiento del lenguaje natural con experiencia en realizar anotaciones en diversas áreas. La consistencia entre las anotaciones se midió a través del coeficiente κ de Cohen. A la hora de su implementación, nos encontramos con distintas decisiones a tomar, siendo una de ellas delimitar el inicio y fin de cada entidad a través del formato IOB2, lo cual permite diferenciar etiquetas asignadas a términos contiguos. El segundo problema se encontró cuando un término tenía asociada más de una etiqueta, para el

cual se desarrollaron dos variantes: (1) considerar dos anotaciones como iguales si y solo si ambos anotadores asignaron el mismo conjunto de etiquetas al mismo término, y (2) crear una lista con cada etiqueta asociada a un término y, a la hora de generar el coeficiente κ, evaluar cada etiqueta por separado y retornar un promedio entre todas. Si bien se implementaron todas las variantes, se decidió utilizar la implementación más simple: sin formato IOB2 y sin considerar múltiples etiquetas asignadas a un mismo término por separado. El valor final de κ sobre el corpus anotado fue de 0.53.

Para extraer información de los textos se trabajó con los artículos enteros y con una versión reducida de los mismos, la cual consistía solamente en el título y la fecha de cada artículo. En cuanto a estos últimos, se notó que son útiles para extraer información sobre la enfermedad, la fecha y la ubicación geográfica que trata el artículo. Esto se debe a que dichas tres entidades ocupan cerca del 92 % de las totales de los títulos y, además, son aquellas cuyos resultados de *F1-score* se encontraron dentro de los más elevados para los distintos algoritmos implementados de NER. Si se necesita extraer información más detallada, como las causas y formas de transmisión de las enfermedades, se debe recurrir al artículo entero.

En cuanto a la detección de entidades nombradas, se desarrolló un algoritmo basado en reglas y otro en redes neuronales profundas. Para evaluar el desempeño de ambos, se implementó un *F1-score micro average* exacto y uno parcial. Este último nos permitió contabilizar aquellas entidades que no tenían una coincidencia total en las anotaciones. Para extraer relaciones, se desarrolló un algoritmo basado en la co-ocurrencia de entidades nombradas con ciertas reglas adicionales, implementado además un *F1-score micro average* exacto para su evaluación.

El algoritmo de reglas se basó en el análisis de un subconjunto de artículos para generar las reglas. La generación de las mismas fue costosa en recursos. En los títulos se obtuvo un *F1 micro average* exacto de 73 %, y las entidades con mejor *F1* allí fueron *Date* (80 %) y *Disease* (94 %). Sin embargo, los resultados para el resto de las entidades estuvieron por debajo del *micro average*, por lo que los títulos no resultan útiles para extraer cierto tipo de información. En el caso de los artículos enteros, los resultados del *F1 micro average* exacto fueron de 51 %. Los resultados para aquellas entidades en las cuales solo se utilizaban expresiones regulares para su detección: *host*, *transmission form*, *origin* y los modificadores (*conditional*, *negation*, *past* y *uncertainty*) estuvieron por debajo del *micro average*. Para aquellas entidades para cuya detección se utilizaron reglas más complejas y *Freeling* (*date*, *disease*, *location* y *number of cases*), los resultados se encontraron por encima del *micro average*. Además, para el caso de la entidad *number of cases*, en los títulos se obtuvo un resultado bajo (*F1* exacto de 15 %), mientras que para los artículos enteros se encontró por encima del *micro average* (*F1* exacto de 53 %). Esto se debió a que en los títulos se encontraban valores numéricos, pero no se establecía a qué hacían referencia (si cantidad de casos, muertes u otra razón), por lo cual los anotadores no solían etiquetarlos como *NoC* pero el algoritmo sí.

El algoritmo de redes neuronales profundas se basó en la propuesta de Akbik, Blythe y Vollgraf [2], que utiliza un módulo etiquetador de secuencias con una arquitectura de red BiLSTM combinado posteriormente con una capa CRF, entrenado con *embeddings* contextualizados a nivel de caracter. Se realizaron 23 iteraciones de entrenamiento sobre los artículos enteros con distintos parámetros, utilizando la técnica de *five-fold cross validation*. En la misma, se varió el tipo de optimizador, tamaño de los *mini-batch*, cantidad de capas de la red BiLSTM, *hidden size*, *variational dropout* y distintos *embeddings* (*wiki-*

dumps y *Spanish Billion Word Corpus*) entrenados con distintas técnicas (*FastText*, *Glove* y *Word2Vec*). Si bien esto fue costoso en tiempo de tiempo y recursos, una vez entrenado sobre el corpus anotado, se obtuvieron mejores resultados (*F1-score micro average* exacto de 81 % para los títulos y 64 % para los artículos enteros) que en el de reglas (73 % y 52 %, respectivamente).

También se desarrolló un algoritmo de extracción de relaciones basado en la co-ocurrencia de entidades nombradas con algunas reglas adicionales (distancia máxima entre entidades y el no establecimiento de relaciones entre un mismo término), el cual no requirió un gran esfuerzo comparado con los de NER. Los resultados fueron de un *F1-score micro average* exacto de 89 % para los títulos y 72 % para los artículos enteros.

7.1. Trabajo a futuro

Una de las dificultades de crear un corpus anotado fue, además del esfuerzo requerido para llevar adelante las anotaciones, establecer un esquema y criterio de anotación lo suficientemente claro para que sea consistente. Como se observó en el gráfico del κ por entidades (ver figura 4.3), existen varias de ellas (*host*, *negation*, *past*, *transmission form*, *uncertainty origin* y *conditional*) que fueron difíciles de definir, pese a que fueron revisadas y modificadas luego de cada una de las iteraciones. Como trabajo a futuro, se podrían eliminar alguna de las inconsistencias del corpus. También, sería útil incrementar el tamaño del corpus anotado para mejorar el entrenamiento del algoritmo de redes neuronales. Para esto, se deberían revisar los criterios y esquemas de anotación, y verificar que los anotadores los comprenden.

En cuanto al algoritmo de extracción de entidades nombradas basado en reglas, teniendo en cuenta los resultados obtenidos, se podría realizar un trabajo posterior para mejorar las reglas, poniendo énfasis en aquellas cuyos resultados se encontraron por debajo del *F1 micro-average*, o mejorando aquellas con *precision* o *recall* bajos (por ejemplo, en los títulos, la entidad *Number of cases* tuvo una *precision* de 8 % y un *recall* de 80 %, por lo que se podría invertir en mejorar la *precision*). Para esto, se debería utilizar un conjunto mayor de notas periodísticas para mejorar las expresiones regulares existentes y generar nuevas reglas. Algunas de las mejoras para las entidades de tipo *Date* y *Location* podrían ser para aquellos casos en los que deberían formar parte de una misma entidad y actualmente son anotadas por separado (por ejemplo, *'Brasil (Sao Paulo)'* es anotado como dos entidades de tipo *Location* en vez de una sola).

En cuanto al algoritmo basado en redes neuronales profundas, se podría realizar la experimentación para la búsqueda de parámetros óptimos para los títulos. En cuanto a los resultados obtenidos, se podría mejorar el corpus anotado para que sea más consistente y el algoritmo tenga un mejor entrenamiento (por ejemplo, arreglar las inconsistencias que se encuentran en las anotaciones para las entidades *Transmission form* y *Origin*).

En cuanto al algoritmo de extracción de relaciones, se podría mejorar el método estableciendo reglas adicionales para aquellas relaciones cuyos resultados fueron bajos (tanto de *F1 micro average* como de *recall* y *precision*). También se podría desarrollar un método basado en redes neuronales profundas utilizando las relaciones del corpus anotado, o crear un método híbrido.

Sobre cada método implementado para extraer información, se podrían evaluar los resultados de cada enfermedad por separado (por ejemplo, los resultados de cada entidad para los artículos que tratan la enfermedad *Chagas*). De esta forma, se podría saber sobre

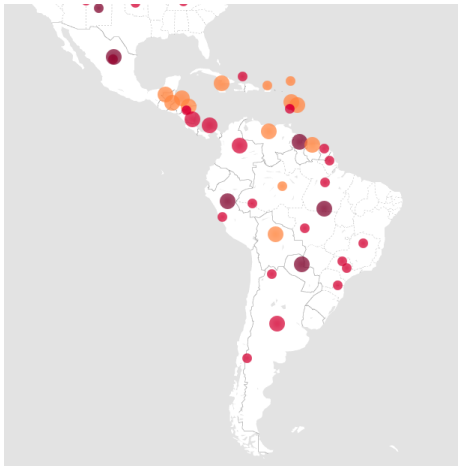


Fig. 7.1: Resultados en Healthmaps para enfermedades "Chagas, zika, dengue, Guillain-barré, hantavirus y sarampión" entre Diciembre 2020 y Junio 2021.

qué enfermedad se obtienen mejores resultados y si hay alguna en particular sobre la cual no se extrae información valiosa.

Otro trabajo a futuro podría ser analizar las entidades y relaciones de cada artículo para poder responder preguntas sobre cada uno de ellos. De esta forma, se podrían extraer datos particulares sobre cada artículo, como la enfermedad mencionada, la cantidad de casos, la ubicación geográfica, los portadores de la enfermedad, las causas y las formas de transmisión, etc. Para esto, no basta con analizar las entidades por separado, ya que hay que tener en cuenta los modificadores de las mismas (por ejemplo, si una entidad está negada o no). Luego, se podría evaluar si en cierta región se está desarrollando un brote de una enfermedad, generando un mapa detallando las enfermedades presentes en ciertas regiones en cierto rango de fechas, como realiza actualmente *Healthmaps*[54] (ver ejemplo en figura 7.1).

También, se podría trabajar en un problema de clasificación en vez de extracción de información. Este es más acotado y se basa en asignar categorías o etiquetas predefinidas a un texto. De esta forma, en vez de buscar entidades nombradas y relaciones, se buscaría clasificar los artículos para que respondan preguntas, como por ejemplo: *'¿sobre qué enfermedad trata el artículo?'*, o *'¿en qué país hubo casos de cierta enfermedad?'*. Luego, se podría determinar cuántos artículos hay sobre una enfermedad particular en determinado país en un rango de fechas.

7.2. Difusión de resultados

Como parte del desarrollo de este trabajo, se escribió una publicación en la *Conference on Computational Natural Language Learning (CoNLL)* de la *Special Interest Group on Natural Language Learning (SIGNLL)*, Association of Computational Linguistics (ACL) [24].

[54] Disponible en `https://healthmap.org/es/` [Accedido en Junio 2021].

REFERENCIAS BIBLIOGRÁFICAS

[1] Eugene Agichtein y Luis Gravano. "Snowball: Extracting relations from large plain-text collections". En: *Proceedings of the fifth ACM conference on Digital libraries*. 2000, págs. 85-94.

[2] Alan Akbik, Duncan Blythe y Roland Vollgraf. "Contextual String Embeddings for Sequence Labeling". En: *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, ago. de 2018, págs. 1638-1649. URL: https://aclanthology.org/C18-1139.

[3] Alan Akbik, Oresti Konomi y Michail Melnikov. "Propminer: A Workflow for Interactive Information Extraction and Exploration using Dependency Trees". En: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Sofia, Bulgaria: Association for Computational Linguistics, ago. de 2013, págs. 157-162. URL: https://www.aclweb.org/anthology/P13-4027.

[4] Elena Arsevska y col. "Web monitoring of emerging animal infectious diseases integrated in the French Animal Health Epidemic Intelligence System". En: *PloS one* 13.8 (2018).

[5] Lalit R Bahl, Frederick Jelinek y Robert L Mercer. "A maximum likelihood approach to continuous speech recognition". En: *IEEE transactions on pattern analysis and machine intelligence* 2 (1983), págs. 179-190.

[6] Miguel Ballesteros, Chris Dyer y Noah A Smith. "Improved transition-based parsing by modeling characters instead of words with LSTMs". En: *arXiv preprint arXiv:1508.00657* (2015).

[7] Frédéric Béchet, Alexis Nasr y Franck Genet. "Tagging unknown proper names using decision trees". En: *proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*. 2000, págs. 77-84.

[8] Emily M. Bender y Dan Flickinger. "Rapid Prototyping of Scalable Grammars: Towards Modularity in Extensions to a Language-Independent Core". En: *Companion Volume to the Proceedings of Conference including Posters/Demos and tutorial abstracts*. 2005. URL: https://www.aclweb.org/anthology/I05-2035.

[9] Daniel M Bikel y col. "Nymble: a high-performance learning name-finder". En: *arXiv preprint cmp-lg/9803003* (1998).

[10] D. K. Bonilla-Aldana y col. "Coronavirus infections reported by ProMED, February 2000-January 2020". En: *Travel medicine and infectious disease* 35.101575 (2020).

[11] Andrew Borthwick y col. "Exploiting diverse knowledge sources via maximum entropy in named entity recognition". En: *Sixth Workshop on Very Large Corpora*. 1998.

[12] Peter F Brown y col. "A statistical approach to machine translation". En: *Computational linguistics* 16.2 (1990), págs. 79-85.

[13] Peter F Brown y col. "Class-based n-gram models of natural language". En: *Computational linguistics* 18.4 (1992), págs. 467-480.

[14] Indra Budi y Stephane Bressan. "Association rules mining for name entity recognition". En: *Proceedings of the Fourth International Conference on Web Information Systems Engineering, 2003. WISE 2003.* IEEE. 2003, págs. 325-328.

[15] Cristian Cardellino. *Spanish Billion Words Corpus and Embeddings*. Ago. de 2019. URL: `https://crscardellino.github.io/SBWCE/`.

[16] M. Carrion y L. C. Madoff. "ProMED-mail: 22 years of digital surveillance of emerging infectious diseases". En: *International health* 9.3 (2017), págs. 177-183.

[17] L. E. Charles-Smith y col. "Using Social Media for Actionable Disease Surveillance and Outbreak Management: A Systematic Literature Review". En: *PloS one* 10.10 (2015).

[18] Eugene Charniak y col. "Parsing as language modeling". En: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 2016, págs. 2331-2336.

[19] Nancy Chinchor, David D Lewis y Lynette Hirschman. *Evaluating message understanding systems: an analysis of the third message understanding conference (MUC-3)*. Inf. téc. SCIENCE APPLICATIONS INTERNATIONAL CORP SAN DIEGO CA, 1993.

[20] Kyunghyun Cho y col. "Learning phrase representations using RNN encoder-decoder for statistical machine translation". En: *arXiv preprint arXiv:1406.1078* (2014).

[21] Michael Collins y Yoram Singer. "Unsupervised models for named entity classification". En: *1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*. 1999.

[22] Mike Conway y col. "Developing a disease outbreak event corpus". En: *Journal of medical Internet research* 12.3 (2010), e43.

[23] Louise Deleger y col. "Building gold standard corpora for medical natural language processing tasks". En: *AMIA Annual Symposium Proceedings*. Vol. 2012. American Medical Informatics Association. 2012, pág. 144.

[24] Antonella Dellanzo, Viviana Cotik y Jose Ochoa-Luna. "A Corpus for Outbreak Detection of Diseases Prevalent in Latin America". En: *Proceedings of the 24th Conference on Computational Natural Language Learning*. Online: Association for Computational Linguistics, nov. de 2020, págs. 543-551. DOI: `10.18653/v1/2020.conll-1.44`. URL: `https://www.aclweb.org/anthology/2020.conll-1.44`.

[25] A. N. Desai y col. "Changing epidemiology of Listeria monocytogenes outbreaks, sporadic cases, and recalls globally: A review of ProMED reports from 1996 to 2018". En: *International journal of infectious diseases* 84 (2019), págs. 48-53.

[26] O. Edo-Osagie y col. "A scoping review of the use of Twitter for public health research". En: *Computers in biology and medicine* 122.103770 (2020).

[27] W Nelson Francis. "A standard corpus of edited present-day American English". En: *College English* 26.4 (1965), págs. 267-273.

[28] Batya Friedman y Helen Nissenbaum. "Bias in computer systems". En: *ACM Transactions on Information Systems (TOIS)* 14.3 (1996), págs. 330-347.

[29] Yarin Gal y Zoubin Ghahramani. "A theoretically grounded application of dropout in recurrent neural networks". En: *Advances in neural information processing systems* 29 (2016), págs. 1019-1027.

[30] Alex Graves y Jürgen Schmidhuber. "Framewise phoneme classification with bidirectional LSTM and other neural network architectures". En: *Neural networks* 18.5-6 (2005), págs. 602-610.

[31] Ralph Grishman y Beth M Sundheim. "Message understanding conference-6: A brief history". En: *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*. 1996.

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

[33] Alan L Hodgkin y Andrew F Huxley. "A quantitative description of membrane current and its application to conduction and excitation in nerve". En: *The Journal of physiology* 117.4 (1952), págs. 500-544.

[34] George Hripcsak y Adam S Rothschild. "Agreement, the f-measure, and reliability in information retrieval". En: *Journal of the American medical informatics association* 12.3 (2005), págs. 296-298.

[35] Zhiheng Huang, Wei Xu y Kai Yu. "Bidirectional LSTM-CRF models for sequence tagging". En: *arXiv preprint arXiv:1508.01991* (2015).

[36] Nancy Ide y James Pustejovsky. *Handbook of linguistic annotation*. Springer, 2017.

[37] Rafal Jozefowicz y col. "Exploring the limits of language modeling". En: *arXiv preprint arXiv:1602.02410* (2016).

[38] Alphonse Juilland y Eugenio Chang-Rodrıguez. *Frequency dictionary of Spanish words*. De Gruyter Mouton, 1964.

[39] Daniel Jurafsky y James H Martin. "Speech and Language Processing: An introduction to speech recognition, computational linguistics and natural language processing". En: *Upper Saddle River, NJ: Prentice Hall* (2008).

[40] FW Kaeding. *Häufigkeitswörterbuch der deutschen Sprache. Festgestellt durch einen Arbeitsausschuß der deutschen Stenographie Systeme*. 1897.

[41] Jae-Ho Kim, In-Ho Kang y Key-Sun Choi. "Unsupervised named entity classification models and their ensembles". En: *COLING 2002: The 19th International Conference on Computational Linguistics*. 2002.

[42] Diederik P Kingma y Jimmy Ba. "Adam: A method for stochastic optimization". En: *arXiv preprint arXiv:1412.6980* (2014).

[43] Philipp Koehn. "Europarl: A parallel corpus for statistical machine translation". En: *MT summit*. Vol. 5. Citeseer. 2005, págs. 79-86.

[44] Miroslav Kubat. "Neural networks: a comprehensive foundation by Simon Haykin, Macmillan, 1994, ISBN 0-02-352781-7." En: *The Knowledge Engineering Review* 13.4 (1999), págs. 409-412.

[45] John Lafferty, Andrew McCallum y Fernando CN Pereira. "Conditional random fields: Probabilistic models for segmenting and labeling sequence data". En: (2001).

[46] Jason Lee, Kyunghyun Cho y Thomas Hofmann. "Fully Character-Level Neural Machine Translation without Explicit Segmentation". En: *Transactions of the Association for Computational Linguistics* 5 (2017), págs. 365-378. DOI: 10.1162/tacl_a_00067. URL: https://www.aclweb.org/anthology/Q17-1026.

[47] ChunYang Liu y col. "Convolution neural network for relation extraction". En: *International Conference on Advanced Data Mining and Applications*. Springer. 2013, págs. 231-242.

[48] Mike Mintz y col. "Distant supervision for relation extraction without labeled data". En: *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. 2009, págs. 1003-1011.

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

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

[51] Julien Rabatel, Elena Arsevska y Mathieu Roche. "PADI-web corpus: Labeled textual data in animal health domain". En: *Data in brief* 22 (2019), págs. 643-646.

[52] Alec Radford, Rafal Jozefowicz e Ilya Sutskever. "Learning to generate reviews and discovering sentiment (2017)". En: *arXiv preprint arXiv:1704.01444* (2017).

[53] Nils Reimers e Iryna Gurevych. "Optimal hyperparameters for deep lstm-networks for sequence labeling tasks". En: *arXiv preprint arXiv:1707.06799* (2017).

[54] C. Rolland y col. "Early Detection of Public Health Emergencies of International Concern through Undiagnosed Disease Reports in ProMED-Mail". En: *Emerging infectious diseases* 26.2 (2020), págs. 336-339.

[55] Tara N Sainath y Bo Li. "Modeling time-frequency patterns with LSTM vs. convolutional architectures for LVCSR tasks". En: (2016).

[56] Erik F Sang y Sabine Buchholz. "Introduction to the CoNLL-2000 shared task: Chunking". En: *arXiv preprint cs/0009008* (2000).

[57] Mike Schuster y Kaisuke Nakajima. "Japanese and korean voice search". En: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2012, págs. 5149-5152.

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

[59] Haykin Simon. *Neural networks: a comprehensive foundation*. Prentice hall, 1999.

[60] Rohini K Srihari. "A hybrid approach for named entity and sub-type tagging". En: *Sixth Applied Natural Language Processing Conference*. 2000, págs. 247-254.

[61] Nitish Srivastava y col. "Dropout: a simple way to prevent neural networks from overfitting". En: *The journal of machine learning research* 15.1 (2014), págs. 1929-1958.

[62] Pontus Stenetorp y col. "brat: a Web-based Tool for NLP-Assisted Text Annotation". En: *Proceedings of the Demonstrations Session at EACL 2012*. Avignon, France: Association for Computational Linguistics, abr. de 2012.

[63] Mariona Taulé, M. Antònia Martı y Marta Recasens. "AnCora: Multilevel Annotated Corpora for Catalan and Spanish". En: *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*. Marrakech, Morocco: European Language Resources Association (ELRA), mayo de 2008. URL: http://www.lrec-conf.org/proceedings/lrec2008/pdf/35_paper.pdf.

[64] R. Thiebaut y S. Cossin. "Artificial Intelligence for Surveillance in Public Health". En: *Yearbook of medical informatics* 28.1 (2019), págs. 232-234.

[65] Alan M Turing. "Computing machinery and intelligence". En: *Parsing the turing test*. Springer, 2009, págs. 23-65.

[66] Yequan Wang y col. "Attention-based LSTM for Aspect-level Sentiment Classification". En: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, nov. de 2016, págs. 606-615. DOI: 10.18653/v1/D16-1058. URL: https://www.aclweb.org/anthology/D16-1058.

[67] Joseph Weizenbaum. "ELIZA—a computer program for the study of natural language communication between man and machine". En: *Communications of the ACM* 9.1 (1966), págs. 36-45.

[68] Terry Winograd. *Procedures as a representation for data in a computer program for understanding natural language*. Inf. téc. MASSACHUSETTS INST OF TECH CAMBRIDGE PROJECT MAC, 1971.

[69] Thomas Nelson Winter. "Roberto Busa, SJ, and the invention of the machine-generated concordance". En: *Faculty Publications, Classics and Religious Studies Department* (1999), pág. 70.

[70] Yu-Chieh Wu y col. "Extracting named entities using support vector machines". En: *International workshop on knowledge discovery in life science literature*. Springer. 2006, págs. 91-103.

[71] Alexander Yates y col. "Textrunner: open information extraction on the web". En: *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*. 2007, págs. 25-26.

[72] Mo Yu y col. "Improved neural relation detection for knowledge base question answering". En: *arXiv preprint arXiv:1704.06194* (2017).

[73] GuoDong Zhou y col. "Exploring various knowledge in relation extraction". En: *Proceedings of the 43rd annual meeting of the association for computational linguistics (acl'05)*. 2005, págs. 427-434.

8. APÉNDICE

En este capítulo, se presenta información complementaría al trabajo. Se comienza con siglas y abreviaturas utilizadas en el trabajo. Luego, en la sección 8.2, se explica cómo interpretar un gráfico violín. En la sección 8.3, se muestran ejemplos de términos de entidades de tipo *Past* y *Conditional* obtenidos por el algoritmo de reglas al utilizar los tiempos verbales para su detección. Finalmente, en la sección 8.4, se presentan distintos detalles de implementación realizados durante el desarrollo de este trabajo.

8.1. Siglas y abreviaturas

# ocur.	Cantidad de ocurrencias
Bi-LSTM	Bidirectional long-short term memory
CRF	Conditional random field
ECM	Error cuadrático medio
FP	Falso positivo o *False positive*
FN	Falso negativo o *False negative*
IAA	Inter-annotator agreement
IE	Information Extraction (extracción de información)
IOB	Formato inside–outside–beginning
IOB2	Formato inside–outside–beginning 2
IQR	Interquartile range
LSTM	Long-short term memory
ML	Machine Learning (Aprendizaje automático)
NER	Named Entity Recognition (reconocimiento de entidades nombradas)
NERC	Named Entity Recognition and Classification (Reconocimiento y clasificación de entidade
NLP	Natural Language Processing (procesamiento del lenguaje natural)
RE	Relation Extraction (extracción de relaciones)
Regex	Expresiones regulares
RNN	Recurrent Neural Network (red neuronal recurrente)
TP	*True positive*
TN	*True negative*

8.2. Cómo interpretar un gráfico violín

En la figura 8.2 se puede observar una figura que explica cómo debe interpretarse un gráfico violín. El ancho de cada parte de la figura representa la frecuencia de entradas. El punto blanco que se observa sobre la caja negra representa la mediana[55] sobre todo el conjunto de datos. La caja negra muestra el rango intercuartil (*IQR* por sus siglas del inglés *interquartile range*), que es la distancia entre el tercer y primer cuartil. El primer cuartil indica la mediana de la mitad inferior de los datos, mientras que el tercer cuartil la de la mitad superior.

[55] La mediana es el valor que se encuentra en el centro de un conjunto de datos ordenados.

Las líneas verticales (llamadas bigotes) se extienden desde el rango intercuartil e indican los valores mínimos y máximos adyacentes. Para la parte superior, se calcula una distancia de $1{,}5 * IQR$ desde el tercer cuartil y se dibuja una línea hasta el máximo valor observado del conjunto de datos que se encuentra hasta esa distancia. De manera similar, se calcula una distancia de $1{,}5 * IQR$ desde el primer cuartil y se dibuja una línea hasta el menor valor del conjunto de datos que cumple con la distancia. Todo valor que no se encuentre dentro de estos rangos es considerado un *outlier* (valor atípico).

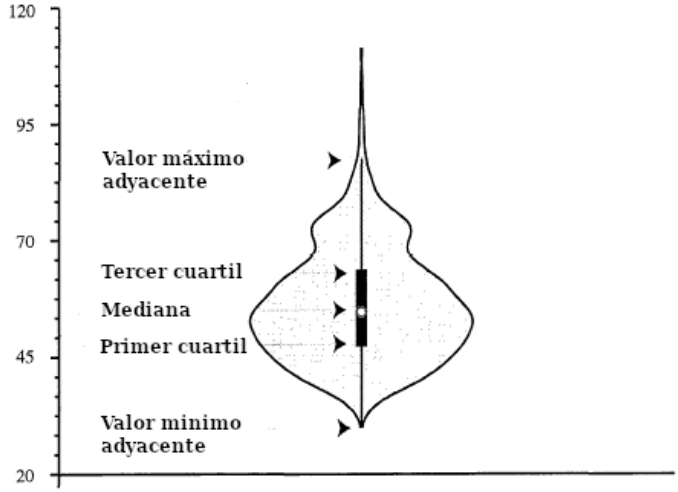


Fig. 8.2: Cómo interpretar un gráfico violín. Gráfico adaptado de `https://towardsdatascience.com/violin-plots-explained-fb1d115e023d`.

8.3. Términos de tipo *Past* y *Conditional* hallados utilizando la técnica de tiempos verbales

En la tabla 8.1 se pueden observar los términos comúnmente hallados por el algoritmo de reglas al utilizar la variante que toma en cuenta los tiempos verbales para reconocer las entidades de tipo *Past* y *Conditional*, y no solamente las expresiones regulares. Esto se reporta sobre los artículos enteros. Para el caso de los títulos, para la entidad de tipo *Past*, solo se encontraron fechas, y no hubo ocurrencias para los tipos condicionales, por lo que no se reportaron los resultados.

8.4. Detalles de implementación

8.4.1. Implementación del κ de Cohen para IAA

En esta sección se presentan los detalles de la implementación del coeficiente κ.

Estructura de los archivos anotados

Cada artículo anotado en *brat* se corresponde con dos archivos: el de texto (con extensión .txt) y el de anotaciones (con extensión .ann). Este último contiene, en cada línea, una anotación realizada en el archivo de texto (ya sea una entidad, o una relación o evento

Past	Ocurrencias
confirmó	107
informó	78
dijo	72
registraron	58
año pasado	53
reportaron	50
registró	49
en lo que va del año	46
confirmaron	44
han registrado	41
han confirmado	34
indicó	32
hasta el momento	32
señaló	31
falleció	30
reporto	28
explicó	28
han reportado	24
murieron	24
presentaron	24

Conditional	Ocurrencias
si	73
podría	22
podrían	6
debería	3
deberíamos	1
plantaría	1
vería	1
habría salido	1
contabilizarían	1
deberían	1
habría experimentado	1
habría facilitado	1
habrían generado	1
sumarían	1
estaría	1

Tab. 8.1: Entidades de tipo pasado y condicional comúnmente halladas por el algoritmo de reglas utilizando reglas de tiempos verbales en los artículos enteros.

que hace referencia a entidades). Como fue mencionado en la sección 4.4.4.1, necesitamos generar tres arreglos distintos: uno con las entidades, otro con las relaciones binarias y otro con las ternarias (eventos).

Para obtener los términos correspondientes a cada entidad anotada, se tuvo que realizar un *mapping* de los términos anotados a los índices que ocupan en el texto original. Por ejemplo, si tuviésemos la oración:

Un niño fue picado por el vector dengue

Tendríamos un diccionario que mapea los índices abarcados por cada *token* a los índices que ocuparía el *token* dentro de cada arreglo. Luego, tendríamos el siguiente diccionario:

$$\begin{aligned} (0,1) &\rightarrow 0 && // \text{ Un} \\ (3,6) &\rightarrow 1 && // \text{ niño} \\ (8,10) &\rightarrow 2 && // \text{ fue} \\ &\ldots. \\ (33,38) &\rightarrow 7 && // \text{ dengue} \end{aligned}$$

De esta forma, al llegar a la anotación de alguna entidad o relación que se encuentra en cierto rango de índices, se buscaría en este diccionario a qué entrada corresponde dentro del arreglo (por ejemplo, *'un'* correspondería con el índice *'0'*).

Para obtener cada *token* que debía ser considerado en el arreglo, se realizó nuevamente la tokenización con FreeLing. Esto se debe a que las tokenizaciones previas no nos servían

puesto que no necesitamos realizar ningún análisis morfológico, ya que esto nos devolvería entidades como fechas o ciudades (por ejemplo, la ciudad *'Buenos Aires'* se transforma en la entidad *'buenos_aires'*) y a nosotros solo nos interesaban los *tokens* por separado.

Anotaciones fragmentadas

Dentro de la herramienta *brat*, existe la posibilidad de realizar anotaciones fragmentadas. Esto permite, para una misma etiqueta, seleccionar un conjunto de *tokens* que no se encuentran consecutivamente en el texto. Por ejemplo, supongamos que se tuviese la oración:

La ciudad de Lima en Perú

En vez de anotar *"Lima en Perú"* con una etiqueta de tipo *LOC*, se podría anotar que la palabra *Lima* y *Perú* forman parte de una misma etiqueta, formando así una anotación fragmentada. Para este caso particular, ya que se había establecido no anotar entidades fragmentadas, para los casos en los que se anotaron entidades fragmentadas, se decidió considerar que la anotación iba de manera consecutiva desde el primer término hasta el último.

IOB2 format

El formato *IOB2* se utiliza para denotar en dónde inicia y termina una anotación (ver sección 4.4.4.1 para más detalle). Supongamos que nos encontramos con dos anotadores distintos tal que el primero realizó la anotación: $token_1$ $token_2$ (X); y el segundo realizó dos anotaciones distintas: $token_1$ (X) y $token_2$ (X). Luego, la anotación con formato IOB2 quedaría así:

- **anotador 1:** $token_1$[B-X] $token_2$[I-X]
- **anotador 2:** $token_1$[B-X] $token_2$[B-X]

En este caso, se indica que la única coincidencia fue en el primer token y en el segundo no hubo *match*.

Por el lado contrario, supongamos para el caso anterior, que el segundo anotador solo anotó: $token_2$ (X). Luego tendríamos lo siguiente:

- **anotador 1:** $token_1$[B-X] $token_2$[I-X]
- **anotador 2:** $token_1$[O] $token_2$[B-X]

En este caso, tendríamos que no hubo ningún *match* entre ambos anotadores.

Entidades

En el archivo con extensión *.ann*, tenemos que cada entidad anotada está formada por el id *T*, seguida por un número que será ascendente con respecto a las siguientes entidades anotadas; luego el tipo de entidad, los índices que ocupa dicha entidad en el archivo de texto (*.txt*) y, finalmente, el texto anotado en sí. Por ejemplo, si tuviésemos una entrada como la siguiente:

T1 Disease 33 38 dengue

Esto indica que la entidad de tipo *Disease* tiene id $T1$ y comprende los índices desde el 33 al 38 del artículo, y la palabra anotada fue *dengue*.

Entonces, dada una entrada con este formato, se busca en el diccionario qué posición debe ocupar la palabra *dengue* en el arreglo de entidades y, finalmente, se agrega en dicha posición del arreglo la etiqueta asignada a dicha entidad, es decir, *Disease*. En el caso en el que la anotación comprenda más de una palabra, se anota para cada palabra en la posición correspondiente al arreglo, que dicha palabra forma parte de la anotación *Disease*.

Si utilizamos el formato IOB, debemos marcar cuál es el *token* que inicia la anotación y cuáles son las subsiguientes. Supongamos que la anotación comprende dos palabras x_1 y x_2 (en la posición i_1 y i_2 correspondientemente). Luego, tenemos en la posición i_1 la anotación *B-Disease* y para la posición i_2 tenemos *I-Disease*.

Además de estas anotaciones, para cada entidad, se genera una entrada en otro diccionario para indicar los índices comprendidos por la entidad T_i. Es decir, para el ejemplo, tenemos una entrada para la entidad *T1* cuyo valor es $(33, 38)$. Esto lo necesitamos posteriormente para las anotaciones de relaciones.

En la figura 8.3, se puede observar un ejemplo del arreglo de entidades en el que se han anotado solamente entidades. En este caso, obtendríamos las siguientes representaciones, dependiendo de si se utiliza formato IOB2 o no:

Sin IOB: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'Disease', 'Disease', 'O', 'O', 'Number_of_cases', 'O', 'Location']

Con IOB: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'B-Disease', 'I-Disease', 'O', 'O', 'B-Number_of_cases', 'O', 'B-Location']

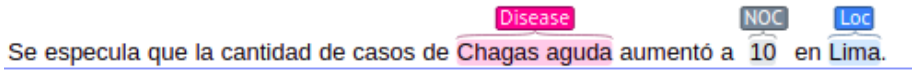


Fig. 8.3: Ejemplo de anotación de entidades

En el mismo se puede observar que existen 3 o 4 entidades (dependiendo de si utilizamos el formato IOB o no) de tipo *Disease*, *Number_of_cases* y *Location*, y se encuentran en la posición 8 (*Chagas*), 9 (*aguda*), 12 (*10*) y 14 (*Lima*).

Relaciones binarias

Las relaciones se encuentran diferenciadas en los archivos de anotaciones por el caracter R más un id ascendente entre relaciones. El mismo se compone de lo siguiente:

$$R_n \ \textit{relation\_name} \ \text{arg1:}T_i \ \text{arg2:}T_j$$

Donde:

- n: índice de la relación entre todas las relaciones binarias.
- *relation_name*: nombre de la relación.

- *arg1* y *arg2*: etiquetas fijas.
- T_i, T_j: referencia a las entidades involucradas en la relación.

Cabe destacar que solamente dos entidades pueden estar relacionadas. Luego, dada una entrada de tipo relación, se obtienen ambas entidades involucradas buscando en el diccionario los índices que abarcan dichas entidades. Las anotaciones sobre la relación se realizan en las posiciones del arreglo que corresponden a la entidad del primer argumento, es decir a T_i y, en dicha posición, se hace referencia a las posiciones que ocupa la entidad del segundo argumento dentro del arreglo. Por ejemplo, si tenemos que la entidad T_1 abarca dos tokens que se encuentran en la posición y_1 e y_2 del arreglo y se relacionan con la entidad $T2$ que abarca los tokens de la posición y_3 e y_4, se anota en las posiciones y_1 e y_2 del arreglo lo siguiente:

$$relation\_name\textbf{:}[IOB\_format]\textbf{:}y_3 - ... - y_4$$

Donde:

- *relation_name:* nombre de la relación.
- *IOB_format:* B si es la primera palabra dentro de la entidad que originó la relación e I en caso de que sea una de las siguientes (siempre y cuando se encuentren trabajando con IOB format).

En la figura 8.4, se puede observar un ejemplo particular para un arreglo de relaciones en el que se han anotado entidades con relaciones entre ellas (el arreglo de entidades es el del ejemplo en la sección 8.4.1). En este caso, obtendríamos las siguientes representaciones, dependiendo de si se utiliza IOB format o no:

Sin IOB: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'DsOcurrs_in:14', 'DsOcurrs_in:14', 'O', 'O', 'nrCasesOcurrsIn:14', 'O', 'O']

Con IOB: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'DsOcurrs_in:B:14', 'DsOcurrs_in:I:14', 'O', 'O', 'nrCasesOcurrsIn:B:14', 'O', 'O']

En el mismo se puede observar que existe la relación *DsOcurrs_in* que parte de los tokens en las posiciones 8 y 9 (*Chagas aguda*) del arreglo de entidades (en el caso con IOB, se puede observar que es la misma relación) e involucra a la entidad en la posición 14 del arreglo (*Lima*), y otra relación de tipo *nrCasesOcurrsIn* que se origina en la entidad en la posición 12 del arreglo (*10*) y se relaciona con la entidad de la posición 14 (*Lima*).

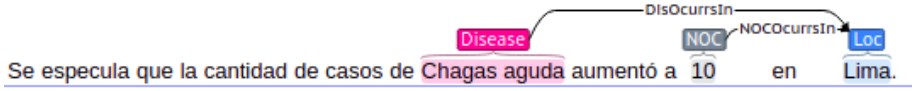


Fig. 8.4: Ejemplo de arreglo de relaciones

Relaciones ternarias

Las entidades ternarias fueron anotadas como eventos en *brat* debido a que la herramienta no tiene soporte para lo mismo. Los eventos son similares a las relaciones, con la diferencia de que pueden involucrar cuantas entidades se hubiesen declarado en el archivo de configuración. La entidad que origina el evento es anotada como una nueva entidad en el archivo de anotación *.ann* y, a cada argumento involucrado, se le puede definir un nombre en vez de tener la etiqueta arg_n. Se diferencian con el caracter E seguido de un id ascendente. El mismo se compone de lo siguiente en el archivo de anotación:

$$E_n \ event\_name{:}T_m \ name\_arg_1 : T_i \ (...) \ name\_arg2 : T_k$$

Donde:

- n: índice del evento dentro de todos los eventos.
- *event_name:* nombre del evento.
- $name\_arg_k$: es el nombre que se le asignó al argumento k.
- T_k: es el id de la entidad involucrada en cada argumento.

Similar a las relaciones, se anota lo siguiente para cada token del arreglo:

$$event\_name{:}[\text{IOB\_format}]{:}índice_i - ... - índice_j{:}índice_n - ... - índice_m$$

Donde:

- *event_name:* nombre del evento.
- *IOB_format:* B si es la primera palabra dentro de la entidad que originó la relación e I en caso de que sea una de las siguientes (siempre y cuando se encuentren trabajando con IOB format).
- $índice_{\{i,n\}} - ... - índice_{\{j,m\}}$: índices que ocupan las n entidades marcadas dentro del arreglo de tokens correspondientemente.

En la figura 8.5, se puede observar un ejemplo en el que se han anotado entidades con un evento. En este caso, obtendríamos las siguientes representaciones, dependiendo de si se utiliza IOB format o no:

> **Sin IOB:** ['O', 'speculatesNrCasesOfDis:8-9:12', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']
>
> **Con IOB:** ['O', 'speculatesNrCasesOfDis:B:8-9:12', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

En el mismo se observa que existe un evento que es originado por el segundo token del texto (*especula*), que su nombre es de tipo *speculatesNrCasesOfDis* e involucra a las entidades que se encuentran entre los índices $8-9$ (*Chagas aguda*) y la que ocupa el token 12 (*10*).

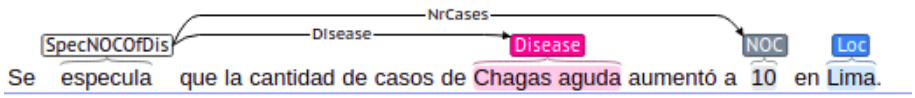


Fig. 8.5: Ejemplo de anotación de entidades con relaciones ternarias (eventos)

Multi-labelling (Múlitples etiquetas)

Al realizar las anotaciones, podría suceder que existan múltiples etiquetas asociadas a un mismo token (múltiples entidades para un token o varias relaciones asociadas a la misma entidad). Hemos considerado tanto manejar estos casos por separado como no realizarlo. Ver figura 8.6 para un ejemplo concreto.

Considerar múltiples etiquetas como una sola

Para este caso, asumiendo que dentro de un mismo token hay dos o más etiquetas, se anota cada una en el mismo índice del arreglo separadas por *;* (punto y coma). Esto implica que dos anotaciones son equivalentes si y sólo si ambos anotadores anotaron el mismo conjunto de etiquetas para ese fragmento de texto. Por ejemplo, supongamos que tenemos las siguientes anotaciones realizadas por dos anotadores distintos sobre el mismo texto en el arreglo de relaciones:

- Anotador 1: [O, T1;T2, O, O]
- Anotador 2: [O, T1, O, O]

Luego, esto resulta en que no hubo ningún match entre ambos anotadores, cuando sí debería haber habido medio match ya que ambos pusieron que la entidad T_1 involucra al token en la posición 2.

Considerar múltiples etiquetas por separado

Para poder evitar el caso anteriormente mencionado, se modifica cómo se encuentra generado cada arreglo de entidades y relaciones. Cada posición es inicializada con una lista vacía en vez del caracter *'O'*, indicando que no hay ningún elemento en dicha posición del arreglo. Luego, a medida que se tienen entradas para agregar en los arreglos, se van insertando en la lista que se encuentra en la posición.

A la hora de calcular el valor de κ, se crea un diccionario que contiene, como claves, las distintas etiquetas que se encuentran en los tres arreglos (entidades, relaciones y eventos). Luego, para cada clave, se guarda un arreglo con valores 1's y 0's, donde 1 indica que el anotador sí colocó dicha etiqueta en esa posición y 0 indica caso contrario. Finalmente, se calcula el coeficiente para cada clave-valor del diccionario y se realiza un promedio de todos los κ obtenidos. Realizar esto genera que los resultados del coeficiente κ sean menores a los obtenidos en los demás casos, ya que nuestros arreglos por clave son esparsos.

En la figura 8.6, se puede observar un ejemplo de anotaciones con relaciones y eventos superpuestos. En este caso, obtendríamos las siguientes representaciones de los arreglos realizando manejo de multi-labelling, dependiendo de si se utiliza formato IOB2 o no:

- **Sin formato IOB**

Entidades: [[], ['speculatesNrCasesOfDis'], [], [], [], [], [], [], ['Disease'], [], [], ['Number_of_cases'], [], ['Location'], [], [], ['Number_of_cases'], [], ['Location']]

Relaciones: [[], [], [], [], [], [], [], [], [], [], [], ['cases_of_disease:8', 'nrCasesOcurrsIn:13', 'nrCasesOcurrsIn:18'], [], [], [], [], ['cases_of_disease:8'], [], []]

Eventos: [[], ['speculatesNrCasesOfDis:11:8', 'speculatesNrCasesOfDis:8:16'], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], []]

- **Con formato IOB**

Entidades: [[], ['B-speculatesNrCasesOfDis'], [], [], [], [], [], [], ['B-Disease'], [], [], ['B-Number_of_cases'], [], ['B-Location'], [], [], ['B-Number_of_cases'], [], ['B-Location']]

Relaciones: [[], [], [], [], [], [], [], [], [], [], [], ['cases_of_disease:B:8', 'nrCasesOcurrsIn:B:13', 'nrCasesOcurrsIn:B:18'], [], [], [], [], ['cases_of_disease:B:8'], [], []]

Eventos: [[], ['speculatesNrCasesOfDis:B:11:8', 'speculatesNrCasesOfDis:B:8:16'], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], [], []]

Para generar posteriormente la parte de diccionarios, supongamos que se tiene como clave la etiqueta *Disease*. Luego, tendríamos la siguiente entrada en el diccionario (de longitud 57 ya que la longitud del texto es de 19 tokens y tendríamos tres arreglos de longitud 19 que equivale a uno de 57), en donde solo para la posición 8 existe una etiqueta de tipo *Disease*:

dict["Disease"]: [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

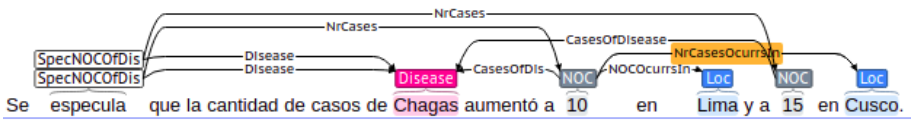


Fig. 8.6: Ejemplo de anotación de entidades con relaciones ternarias (eventos) superpuestas

Generación del coeficiente κ

Luego de armar cada uno de estos tres arreglos por separado (el de las entidades, el de las relaciones y el de los eventos), se prosigue a generar un arreglo final de longitud $3 * N$ formado de la concatenación de cada uno de estos. Se podrán elegir las combinaciones de parámetros deseadas (con multi-labelling o no, y con formato *IOB* o no) y, dependiendo del tipo de multi-labelling que se haya elegido, se calcula directamente el coeficiente sobre los arreglos de ambos anotadores (sin multi-label) o se calcula el promedio de los coeficientes obtenidos por cada anotador y cada etiqueta (con multi-label).

8.4.2. Herramientas de procesamiento de lenguaje natural

Se evaluaron distintas herramientas disponibles para el procesamiento del lenguaje natural de textos en español con el fin de poder realizar las siguientes tareas que facilitan el

análisis de los textos: tokenización, segmentación de oraciones, *part-of-speech tagging* (*PoS-tagging*), lematización y normalización de los textos, entre otras. Utilizando los distintos textos provenientes de ProMED-mail, se realizaron las tareas correspondientes con cada herramienta para evaluar su desempeño y elegir la apropiada.

Las herramientas que se encontraron en español fueron *spaCy*, *Stanford* y *Freeling*. Las primeras dos fueron descartadas, quedándonos con la última. En las siguientes secciones presentamos cada una de ellas.

SpaCy

SpaCy[56], es una herramienta gratuita y *open-source* para el desarrollo avanzado de Procesamiento del Lenguaje Natural en Python. Tiene soporte para más de 50 idiomas, incluyendo una opción multi-idiomas.

Luego de realizar su correspondiente integración, se procedió a evaluar su desempeño en segmentación de oraciones. Allí notamos que no era óptimo ya que muchas palabras abreviadas (por ejemplo, *Dr.* o *Dra.*) eran consideradas como separadores de oraciones y no como abreviaturas. *SpaCy* ofrece la posibilidad de modificar sus archivos de configuración, pero se detectó que para el idioma español, estos archivos contenían muy pocos datos del lenguaje. Por esta razón se decidió descartar la herramienta.

Stanford

Al realizar la segmentación de oraciones con la librería *Stanford NLP*[57], se llegó a una segmentación de oraciones similar a la de *spaCy*, por lo que se decidió descartarla.

Luego, se decidió probar la herramienta *Stanford CoreNLP*[58]. A diferencia de las anteriores, esta ofreció mejores resultados ya que abreviaturas como *Dra.* no implicaban un fin de oración. No obstante, siguieron existiendo problemas, alguno de ellos fueron:

- Otras palabras como *Dras.* eran tomadas como separador de oración, por lo que la lista de palabras que no deben separar oraciones no era muy completa.

- **Segmentación errónea para comillas:** Al encontrarse una comilla seguida de un punto, consideraba la comilla como parte de la oración anterior, y la siguiente comenzaba sin las comillas. Por ejemplo, supongamos que tenemos la oración: *"El Departamento de Salud de Florida señaló en un comunicado que uno de los adultos contrajo la enfermedad en el exterior, sin ofrecer más detalles. "Es muy importante protegerse uno mismo, a la familia y a la comunidad, vacunándose", dijo Miranda Hawker, administradora del Departamento de Salud del condado de Indian River."*. En este caso, al realizar la segmentación de oraciones, la herramienta devolvía que la primera oración era: *"El Departamento de Salud de Florida señaló en un comunicado que uno de los adultos contrajo la enfermedad en el exterior, sin ofrecer más detalles."*. Y la segunda: *"Es muy importante protegerse uno mismo, a la familia y a la comunidad, vacunándose", dijo Miranda Hawker, administradora del Departamento de Salud del condado de Indian River."*.

[56] Disponible en `https://spacy.io/` [Accedido en Junio de 2021].
[57] Disponible en `https://nlp.stanford.edu/software/` [Accedido en Junio de 2021].
[58] Disponible en `https://stanfordnlp.github.io/CoreNLP/` [Accedido en Junio de 2021].

- Otro problema con las comillas era que, al haber varias oraciones dentro de una cita (entre comillas), separa las oraciones como distintas, cuando debería ser todo parte de la misma oración.

Dentro de la documentación de la herramienta, encontramos varias opciones al momento de inicializar el servidor para corregir dichos problemas, pero esto generaba tener un archivo con todas las configuraciones juntas y no por separado. Posteriormente, se decidió descartar esta herramienta.

FreeLing

Otra librería que se utilizó fue FreeLing, que fue diseñada originalmente para el lenguaje español, por lo cual ya se esperaba un mejor manejo del idioma. Inicialmente, se realizó la segmentación de oraciones, observándose mejoras con respecto a las demás herramientas utilizadas. Por ejemplo, dentro del conjunto de palabras que forman parte de las excepciones para la segmentación de oraciones, se encontraba, entre otras, la palabra *"Dras."*. Algo a destacar de esta herramienta, es que la documentación es extensa y contiene diversos archivos de configuración y parámetros para los algoritmos, que permiten adaptar la herramienta a las necesidades de cada proyecto, distinto a las demás herramientas que son más limitadas en esta área. Por ejemplo, al realizar la segmentación de oraciones, se puede agregar en el archivo de configuración el concepto de *markers*, que son pares de caracteres tales que nunca se realiza una segmentación de oración si se encuentra dentro de ellos. Esto quiere decir que, si se tienen los markers:

```
<Markers>
‘‘ ’’
( )
{ }
/* */
‘ ’
</Markers>
```

Nunca se separarán oraciones que se encuentren dentro de comillas (u otro de los pares), problema que se tenía con la librería de Stanford.

También, al momento de realizar la tokenización del texto, se pueden obtener distintos tipos de datos sobre el mismo, como por ejemplo el lema de cada palabra (la entrada en el diccionario) y el *PoS-tagging*, entre otros. Los distintos tipos de *PoS-tagging* que clasifica son: adjetivo, conjunción, determinante, sustantivo, pronombre, adverbio, adposición, verbo, número, fecha, interjección o puntuación, y dentro de cada una de las categorías posee una cierta cantidad de atributos como por ejemplo el género (masculino o femenino), el tipo del que depende la etiqueta (por ejemplo para sustantivo puede ser propio o común), la etiqueta relacionada al NER (si se optó realizarlo), etc.

Además contiene un análisis morfológico del texto que permite extraer, entre otras cosas, fechas y números, y ejecutar NER, que fue utilizado para la parte de detección de entidades nombradas descriptas en la sección 5.4.

8.4.3. Integración de los *gazetteers* en las librerías de NLP

En esta sección, presentamos cómo se generaron los distintos archivos correspondientes para proveer gazetteers a las distintas librerías de NLP utilizadas. Por un lado, la librería

de StanfordNLP requería que los datos estuviesen dentro de un archivo con el formato:

DATO NOMBRE_DEL_TAG
(por ejemplo: Buenos Aires LOCATION)

Para la librería de Freeling, tuvimos que generar distintos tipos de archivos:

- **gazLOC-c:** Nombres de ubicaciones geográficas en minúscula y con espacios reemplazados por guiones bajos (por ejemplo, *buenos_aires*).
- **gazLOC-p:** Palabras que forman parte de ubicaciones geográficas con varias palabras. Esto quiere decir que si el nombre de una ubicación está compuesto por dos palabras, luego se agregan a este archivo ambas palabras por separado (por ejemplo, para *'Buenos Aires'*, 'buenos' estaría en una línea y 'aires' en otra). Aquellas ubicaciones geográficas que están compuestas por solo una palabra no deben ser agregadas aquí.
- **cities:** Lista de ciudades, en minúscula y con los espacios reemplazados por guiones bajos.

Una vez generados los archivos, para que se utilicen los gazetteers en Freeling, se deben actualizar todos los archivos de gazetteers que ya existen y luego ejecutar los módulos de NER (para reconocer todas las entidades nombradas) seguido de NEC (para clasificar las entidades). Para el caso de StanfordNLP, se utilizó el módulo de *regexner* para identificar ubicaciones con nuestros archivos de *gazetteers* utilizando la opción *regexner.mapping*. Luego de cargar los archivos, se prosigue a procesar nuestros textos y extraer las etiquetas correspondientes a las ubicaciones.

8.4.4. Detección de ubicaciones geográficas en los títulos utilizando FreeLing

Para detectar las ubicaciones geográficas en el título, encontramos un problema con la librería de *Freeling* ya que no estaba detectando correctamente las ubicaciones en el título (es decir, asignando las etiquetas correspondientes). Los títulos se encuentran escritos en mayúsculas y, normalmente, son una enumeración de datos (por ejemplo, *HANTAVIRUS - ARGENTINA: (CHU/RN) BROTE, SECUENCIACIÓN GENÉTICA DE MUESTRAS CLÍNICAS, INFORME INRB-USAMRIID*). Debido a la importancia de detectar si el artículo tiene que ser anotado o no, y a que la longitud de los títulos solía ser acotada, se procesó cada *token* por separado y se evaluó si se encontraba dentro de la lista de ubicaciones almacenadas.