Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias de la Computación

Extracción de relaciones semánticas y entidades en documentos del dominio de salud

Tesis presentada para obtener el título de:
Maestro en Ciencias de la Computación

Presenta:
Ing. Erick Barrios González

Dirigida por:
Dr. Guillermo De Ita Luna

Dra. Mireya Tovar Vidal

Agosto 2022

Dedicatoria

A mi mamá, mi papá, mis hermanas que me han apoyado siempre, y a quienes me apoyaron total y parcialmente durante mis estudios.

Agradecimientos

A CONACYT por la beca otorgada durante los años de estudio, para poder obtener este grado académico.

Al proyecto VIEP-BUAP 2021 y al Laboratorio Nacional de Supercómputo del Sureste de México.

A mis asesores de tesis, que me apoyaron constantemente en su elaboración, principalmente a la Dra. Mireya Tovar Vidal por la guía y apoyo brindado durante el curso de mi maestría.

A Dios y a mi familia por el apoyo incondicional que me han dado.

Resumen

Los métodos de procesamiento del lenguaje natural (PLN) se utilizan cada vez más para extraer conocimientos de textos de salud no estructurados. Por ejemplo, analizar información médica, estructurarla en categorías definidas y agruparlas en bases de datos. La organización de la información médica puede ser de utilidad para análisis clínicos, para disminuir el número de errores médicos, o puede ayudar a la toma de decisiones más adecuadas en determinados casos.

En esta tesis se espera extraer automáticamente una gran variedad de conocimientos de documentos de salud redactados en español. Para ello este problema se divide en 2 subtareas:

1. Identificar entidades por documento y sus tipos. Estas entidades son todos los términos relevantes (una palabra o varias palabras) que representan elementos semánticamente importantes en una oración. Se clasificarán estas entidades en 4 categorías: conceptos, acciones, predicados y referencias.

2. Vincular las entidades detectadas y etiquetadas en el documento de entrada. El propósito de esta subtarea es reconocer todas las relaciones semánticas relevantes entre las entidades reconocidas y dividirlas según las siguientes categorías: relaciones generales, relaciones contextuales, roles de acción y roles de predicado.

Esta investigación aborda un escenario, donde para resolver las tareas de identificación de entidades y extracción de relaciones semánticas se utiliza una metodología basada en el uso de redes neuronales recurrentes. Para evaluar la metodología se hará uso de las métricas: precisión, exhaustividad y F_1.

Abstract

Natural language processing (NLP) methods are increasingly used to extract knowledge from unstructured health texts. For example, analyze medical information, structure it into defined categories and group it in databases. The organization of medical information can be useful for clinical analysis, to reduce the number of medical errors, or it can help to make the most appropriate decisions in certain cases.

In this thesis it is expected to automatically extract a great variety of knowledge from Health documents written in spanish. For this, this problem is divided into 2 subtasks:

1. Identify entities by document and their types. These entities are all relevant terms (one word or several words) that represent semantically important elements in a sentence. These entities will be classified into 4 categories: concepts, actions, predicates and references.

2. Link the detected and tagged entities in the input document. The purpose of this subtask is to recognize all relevant semantic relationships between recognized entities and divide them according to the following categories: general relationships, contextual relationships, action roles, and predicate roles.

This research explores a scenario where, using a methodology based on the use of recurrent neural networks, where entities are identified and semantic relationships are extracted. To evaluate the methodology, the metrics will be used: precision, recall and F_1.

Índice general

Dedicatoria I

Agradecimientos II

Resumen III

Abstract IV

Índice de figuras VII

Índice de tablas VIII

1. Introducción **2**

1.1. Planteamiento del problema 2

1.2. Objetivos 6

1.2.1. Objetivo general 6

1.2.2. Objetivos específicos 6

1.3. Antecedentes 6

1.4. Justificación 8

1.5. Metodología 8

1.6. Distribución de la tesis 10

2. Marco teórico **11**

2.1. Procesamiento de lenguaje natural 11

2.1.1. Identificación de entidades 13

2.1.2. Extracción de relaciones semánticas 15

2.1.3. Estilo de la anotación (Identificación de entidades) 16
2.1.4. Dependencias universales (Extracción de relaciones) 16
2.1.5. Marcado descriptivo (Extracción de relaciones) 17
2.2. Aprendizaje automático 17
2.3. Aprendizaje profundo 18
2.3.1. Redes neuronales convolucionales 18
2.3.2. Redes neuronales recurrentes 20
2.3.3. Redes *Transformers* 22
2.3.4. Incrustaciones de palabras en el aprendizaje profundo 24
2.3.5. Campos aleatorios condicionales en el aprendizaje profundo . . 25

3. Estado del arte **26**
3.1. Sistemas para reconocimiento de entidades 26
3.2. Sistemas para extracción de relaciones 31
3.3. Sistemas para ambas subtareas 34

4. Metodología de solución **40**
4.1. Pre-procesamiento de la información 41
4.1.1. Descripción del corpus 41
4.1.2. Estructura del pre-procesamiento 43
4.1.3. Etiquetado 45
4.1.4. Estilo de la anotación (identificación de entidades) 47
4.1.5. Modelo de incrustación de palabras 50
4.1.6. Propuesta para el pre-procesamiento (identificación de entidades) 51
4.1.7. Propuesta para el pre-procesamiento (extracción de relaciones semánticas) 53
4.2. Arquitectura y extracción de información 54
4.2.1. Identificación de entidades 54
4.2.2. Entrada para entrenamiento y predicción de los sistemas de identificación de entidades 56
4.2.3. Extracción de relaciones 58
4.2.4. Entrada para entrenamiento y predicción de los sistemas de extracción de relaciones semánticas 59

4.3. Programación y entrenamiento de la red neuronal de aprendizaje profundo . . . 63
4.3.1. Método para validación cruzada durante el entrenamiento . . 64
4.4. Evaluación de los resultados . . . 65
4.4.1. Métricas de evaluación . . . 65
4.4.2. Método de evaluación . . . 67

5. Resultados **69**
5.1. Resultados de la etapa de pre-procesamiento . . . 69
5.1.1. Resultados: Información del corpus . . . 69
5.1.2. Resultados: Incrustación de palabras . . . 71
5.2. Herramientas utilizadas . . . 73
5.3. Resultados de sistemas para la identificación de entidades . . . 74
5.3.1. Sistema *BiLSTM* . . . 74
5.3.2. Sistema *BiLSTM-CRF* . . . 75
5.3.3. Sistema *BiLSTM-CNN* . . . 76
5.3.4. Evaluación de los sistemas propuestos para identificación de entidades . . . 77
5.4. Resultados de sistemas para la extracción de relaciones . . . 79
5.4.1. Sistema *BiLSTM-CRF* . . . 79
5.4.2. Sistema *BERT* (*BERT multilingual base model (cased)*) . . . 80
5.4.3. Evaluación de los sistemas propuestos . . . 81
5.5. Resultados de los sistemas propuestos en *eHealth-KD Challenge 2021* 85
5.5.1. *Comparación de resultados de los sistemas propuestos para la subtarea A y subtarea B* . . . 85
5.5.2. *Comparación de los resultados en el escenario principal* . . . 87

Conclusiones **89**

Bibliografía **91**

Índice de figuras

1.1. Ejemplo de reconocimiento de entidades y relaciones [50]. 5

2.1. Ejemplo versión simple de *RNN* (*vanilla*) [23]. 20
2.2. Ejemplo etapas en la arquitectura de una neurona *LSTM* [23]. 21
2.3. Ejemplo red *BILSTM* [23]. 22
2.4. Ejemplo de red *Transformer* [57]. 23

4.1. Pasos realizados para el pre-procesamiento de la identificación de entidades. 43
4.2. Pasos realizados para el pre-procesamiento de la extracción de relaciones. 43
4.3. Pasos para la identificación de entidades. 54
4.4. Ejemplo de entrada de ejes para entrenamiento. 57
4.5. Ejemplo de entrada de ejes para predicción. 58
4.6. Pasos para la extracción de relaciones. 59
4.7. Evaluación del entrenamiento de cada modelo. 65
4.8. Fase de evaluación de los resultados. 68

5.1. Arquitectura red *BiLSTM*. 75
5.2. Arquitectura red *BiLSTM-CRF*. 76
5.3. Arquitectura red *BiLSTM-CNN*. 77
5.4. Arquitectura red *BERT*. 80

Índice de tablas

2.1. Etiquetado *POS spacy* 14

3.1. Tabla comparativa entre sistemas para el reconocimiento de entidades entre participantes del *eHealth-KD challenge*. 32
3.2. Tabla comparativa entre sistemas para la extracción de relaciones entre participantes del *eHealth-KD challenge*. 35
3.3. Tabla comparativa entre sistemas para ambas subtareas entre participantes del *eHealth-KD challenge*. 38

4.1. Ejemplo formato corpus proporcionado en [50]. 42
4.2. Ejemplo formato *BRAT standoff*. 42
4.3. Conversión de etiquetado de *Freeling 4.2* 46
4.4. Ejemplo de etiquetado *POS* 47
4.5. Ejemplo de etiquetado de dependencias en *Spacy* 47
4.6. Ejemplo de estilo de anotación *BILOU*. 49
4.7. Ejemplo de estilo de anotación usando "P_" 49
4.8. Ejemplo de estilo de anotación *BIO*. 49
4.9. Frecuencia de aparición de entidades en base al etiquetado *POS*. . . . 51
4.10. Frecuencia de aparición de entidades (conformadas por más de una palabra) en base al etiquetado *POS*. 51
4.11. Ejemplo de secuencia de etiquetas creadas con estilo de anotación *BIO* y *BILOU* 52
4.12. Ejemplo de asignación de valores para sistema *BILSTM* 55
4.13. Ejemplo de estilo de anotación *BIO* y matriz binaria para sistema *BiLSTM-CRF* 55

4.14. Ejemplo de estilo de anotación *BILOU* y matriz binaria para sistema *BiLSTM-CRF* 56
4.15. Entrada para incrustaciones a nivel carácter 56
4.16. Ejemplo entrada sistema (eje “X"), para la identificación de entidades. 57
4.17. Ejemplo entrada sistema (eje “Y"), para identificación de entidades. . 58
4.18. Ejemplo de etiquetado para extracción de relaciones (con múltiples relaciones) en red *BiLSTM-CRF* (Opción 1). 60
4.19. Ejemplo de etiquetado para extracción de relaciones (con una relación) en red *BiLSTM-CRF* (Opción 2). 61
4.20. Ejemplo de etiquetado para extracción de relaciones (con una relación) en red *BiLSTM-CRF* (Opción 3). 61
4.21. Ejemplo de etiquetado base final para extracción de relaciones para red *BiLSTM-CRF* (Opción 4). 61
4.22. Ejemplo entrada con dependencias (*Spacy*) para red *BiLSTM-CRF* para extracción de relaciones 62
4.23. Ejemplo entrada con patrones y dependencias (*Spacy*) para red *BiLSTM-CRF* para extracción de relaciones 63
4.24. Ejemplo de entrada usando etiquetas de marcado descriptivo para dependencias (*Spacy*) para red *BiLSTM-CRF* para extracción de relaciones 63

5.1. Cantidad de relaciones por corpus 71
5.2. Modelos de incrustación de palabras 72
5.3. Resultados de los sistemas para identificación de entidades, comparando estilos de anotación (*Spacy*). 78
5.4. Resultados de los sistemas para identificación de entidades, comparando estilos de anotación (*Freeling*). 78
5.5. Resultados de los sistemas para identificación de entidades, añadiendo la propuesta con diferentes estilos de anotación *Spacy*). 79
5.6. Ejemplo de entrada de etiquetado para red *BERT* 81
5.7. Resultados para cada tipo de relación con opción 2 - Entrada simple (BiLSTM-*CRF*) 82

5.8. Resultados para cada tipo de relación con opción 2 - Entrada con patrones (BiLSTM-*CRF*) 83
5.9. Resultados para cada tipo de relación con opción 2 - Entrada de marcado descriptivo (*BiLSTM-CRF*) 83
5.10. Resultados para cada tipo de relación con opción 4 - (*BiLSTM-CRF*) 84
5.11. Resultados para cada tipo de relación con opción 4 - (*BERT*) 84
5.12. Resultados totales de opción 2 (BiLSTM-*CRF*) con las diferentes entradas propuestas. 84
5.13. Resultados de los mejores resultados en las opciones de etiquetado 2 y 4, para la extracción de relaciones. 85
5.14. Resultados de los mejores sistemas para cada escenario. 85
5.15. Comparación de resultados de la subtarea A 86
5.16. Comparación de resultados de la subtarea B 86
5.17. Comparación de resultados del escenario principal 87

Capítulo 1

Introducción

En este capítulo se muestra el planteamiento del problema que se resuelve en este trabajo de tesis, también se muestran los objetivos, antecedentes, la justificación y la metodología a utilizar en este trabajo, finalmente, se muestra la distribución de este trabajo.

1.1. Planteamiento del problema

Actualmente es más fácil disponer de información médica en formato electrónico e incluso contar con que esa información este estructurada. Sin embargo, la información incluida en medios tradicionales (como publicaciones, manuscritos académicos, informes clínicos, etc.), son poco útiles para seleccionar la información más apropiada en cada caso, sea en el ámbito clínico o en el de la investigación. Es por ello que surge la necesidad de crear nuevas maneras de extraer conocimiento de textos de salud y que la información sea íntegra y confiable para cualquier estudio o análisis que se realice con la información. Este trabajo de tesis se orienta a resolver las tareas planteadas en *eHealth-KD challenge 2021* [50].

El escenario principal de *eHealth-KD challenge 2021* [50] cubre tanto el reconocimiento de entidades (subtarea A) como la extracción de relaciones (subtarea B). El escenario principal se calcula a partir de los resultados de cada subtarea aplicando las fórmulas propuestas por *eHealth-KD challenge 2021*. Para cada subtarea la entrada consta principalmente de varias oraciones sin formato, mientras que la salida

esperada es un archivo *BRAT (.ann)* con todas las entidades (subtarea A) y relaciones (subtarea B) correspondientes encontradas. No es necesario que los archivos de salida esperados y el archivo de evaluación coincidan en el identificador de cada entidad y relación, ni en su orden.

Dada una lista de documentos de *eHealth-KD challenge* redactados en español, el objetivo de la subtarea A es identificar todas las entidades por documento y sus tipos. Estas entidades son todos los términos relevantes (una sola palabra o varias palabras) que representan elementos semánticamente importantes en una oración. Las entidades siempre constarán de una o más palabras completas (es decir, no un prefijo o sufijo de una palabra) y nunca incluirán símbolos de puntuación, paréntesis, etc. Hay cuatro tipos de entidades:

- Concepto: Identifica un término, concepto, idea relevante en el dominio del conocimiento de la oración.
- Acción: Identifica un proceso o modificación de otras entidades. Puede indicarse mediante un verbo o construcción verbal, pero también mediante sustantivos. También se puede utilizar para indicar relaciones funcionales no verbales.
- Predicado: Identifica una función o filtro de otro conjunto de elementos, que tiene una etiqueta semántica en el texto, y se aplica a una entidad con algunos argumentos adicionales.
- Referencia: Identifica un elemento textual que hace referencia a una entidad (de la misma oración o de otra diferente), que se puede indicar mediante pistas textuales.

La subtarea B continúa desde la salida de la subtarea A, vinculando las entidades detectadas y etiquetadas en el documento de entrada. El propósito de esta subtarea es reconocer todas las relaciones semánticas relevantes entre las entidades reconocidas. Las relaciones semánticas se dividen en diferentes categorías:

Relaciones generales: Son relaciones de propósito general entre dos entidades (concepto, acción, predicado o referencia) que tienen una semántica específica. Cuando se aplica una relación de este tipo, se refiere a una relación de dominio (etiquetar

una frase clave como enlace entre dos unidades de información), ya que su semántica es independiente de cualquier etiqueta textual:

- Es-un (hipónimia): Indica que una entidad es un subtipo, instancia o miembro de la clase identificada por la otra.
- Igual-que (sinonimia): Indica que dos entidades son semánticamente iguales.
- Tiene-propiedad: Indica que una entidad tiene una propiedad o característica determinada.
- Parte-de (meronimia): Indica que una entidad es parte constituyente de otra.
- Causa (causa efecto): Indica que una entidad provoca la existencia u ocurrencia de otra.
- Implica: Indica que la existencia de una entidad implica la existencia u ocurrencia de otra.

Relaciones contextuales: Estas relaciones permiten refinar una entidad (concepto, acción, predicado o referencia) adjuntando modificadores. Estos son:

- En-tiempo (relación temporal): Para indicar que algo existe, ocurre o se limita a un marco de tiempo.
- En-lugar (relación espacial): Para indicar que algo existe, ocurre o está confinado a un lugar o ubicación.
- En-contexto: Para indicar un contexto general en el que algo sucede, como un modo, manera o estado.

Roles de acción (relación de caso): Indican qué rol juegan las entidades relacionadas con una Acción:

- Sujeto: Indica quién realiza la acción.
- Objetivo: Indica quién recibe el efecto de la acción. Las acciones pueden tener varios sujetos y objetivos, en cuyo caso la semántica interpretada es que la unión de los sujetos realiza la acción sobre cada uno de los objetivos.

Roles de predicado: Indican qué papel juegan las entidades relacionadas con un predicado:

- Dominio: Indica la entidad principal sobre la que se aplica el predicado.
- Argumento: Indica una entidad adicional que especifica un valor para que el predicado tenga sentido. La semántica exacta de este argumento depende de la semántica de la etiqueta del predicado.

Figura 1.1: Ejemplo de reconocimiento de entidades y relaciones [50].

En la Figura 1.1 se muestran ejemplos de cómo se clasificarían las entidades en conceptos, acciones, predicados y referencias. Por ejemplo, al detectar entidades "asma", "enfermedad", "vías respiratorias", "prolongada", "sol", "verano", "piel", "personas" y "60 años" son conceptos, "afecta", "exposición", "daños", son acciones, "esta" es una referencia y "mayores" es un predicado. También se muestran algunos ejemplos de los tipos de relaciones descritas anteriormente. Por ejemplo, *is-a* (Es-un) señala que el concepto de "asma" es una "enfermedad", *subject* (Sujeto) señala que "asma" tiene como acción "afecta", *target* (Objetivo) donde la acción "afecta" estaría aplicando sobre el concepto de "vias respiratorias", *causes* (Causa) donde la acción "exposición" tiene como efecto la acción "daños", *in-time* (En-tiempo) donde la acción "exposición" es llevada durante el "verano", *in-context* (En-lugar) donde la acción "exposición" se realiza en el contexto "prolongada", *domain* (Dominio) donde el predicado "mayores"

está aplicando sobre el concepto de "personas" y *arg* (Argumento) donde el predicado "mayores" indica la cantidad de "60 años", que restringe la edad mínima para que el predicado sea verdadero.

1.2. Objetivos

El objetivo general define el alcance de la investigación y los objetivos específicos determinan las tareas a realizar.

1.2.1. Objetivo general

Desarrollar una metodología para la detección de entidades y la extracción de relaciones semánticas en textos no estructurados del dominio de salud, a través de modelos basados en redes neuronales recurrentes.

1.2.2. Objetivos específicos

- Construir un método de pre-procesamiento para el corpus utilizado.
- Diseñar una metodología de solución para la identificación de entidades y la extracción de relaciones semánticas usando redes neuronales recurrentes.
- Evaluar los resultados experimentales obtenidos haciendo uso de las métricas precisión, exhaustividad y F_1.

1.3. Antecedentes

En la competencia *eHealth-KD challenge* [50] se propone modelar el lenguaje humano en un escenario en el que los documentos electrónicos de salud (*eHealth*) escritos en español podrían ser legibles por la computadora desde un punto de vista semántico. La finalidad es fomentar el desarrollo de tecnologías de software para extraer automáticamente una gran variedad de conocimientos de los documentos de eSalud (*eHealth*) redactados en idioma español. Pese a que esta problemática esté orientada al dominio de la salud, la forma de extraer la información también podría

aplicarse con un propósito más general y aplicarse en otros dominios. Este evento se ha llevado por varios años. A continuación se describe brevemente los cambios que se han hecho a lo largo de los años que el evento ha estado vigente: La tarea de *eHealth-KD challenge* tiene sus orígenes en 2018 en el TASS - 2018 (Taller de análisis semántico en la SEPLN) y estaba inspirado en tareas de PLN como la tarea 10 de Semeval-2017 "ScienceIE". En esta edición la "subtarea A" correspondía a la identificación de frases clave, la "subtarea B" correspondía a la clasificación de frases clave (donde las frases clave son todos los términos relevantes que representan semánticamente elementos importantes en la oración) y "la subtarea C" correspondía a la asignación de relaciones semánticas. Posteriormente, en 2019 *eHealth-KD challenge* [49] tiene su primera aparición en *IberLEF* (*Iberian Languages Evaluation Forum*). La competencia *IberLEF* es una campaña de evaluación comparativa de sistemas de procesamiento del lenguaje natural en español y otras lenguas ibéricas. Su objetivo es alentar a la comunidad investigadora a organizar tareas competitivas de procesamiento, comprensión y generación de textos para definir nuevos desafíos de investigación y establecer nuevos resultados de vanguardia en esos idiomas La edición de 2019 *eHealth-KD challenge* es más similar a la tarea de 2021 debido a que ya plantea todo en 2 subtareas, "subtarea A" la extracción y clasificación de frases clave (donde las categorías son: concepto, acción, predicado y referencia) y como "subtarea B" a la extracción de relaciones entre frases clave extraídas, donde las relaciones se dividen en 4 categorías (relaciones generales, relaciones contextuales, roles de acción, roles de predicado). En *eHealth-KD challenge 2020* [48] la "subtarea A" y "subtarea B" siguen siendo las mismas respecto al año pasado sin embargo la "subtarea A" cambia su nombre a "Reconocimiento de entidades", las diferencias que se pueden observar respecto a su antecesor son cambios en los formatos de salida de los resultados, se incluyen textos referentes al COVID-19 y en la evaluación de las tareas se agrega una nueva evaluación donde se incluye evaluar el sistema con 100 oraciones que no están relacionadas al tema de la salud. Por último, *eHealth-KD challenge 2021* (la tarea en la que se basa este trabajo) tiene algunos cambios a diferencia de la tarea del año pasado, debido a este año se removió el escenario de evaluación referente a textos que no son de temas referentes al dominio de la salud.

1.4. Justificación

Existe una necesidad de extraer conocimientos de textos de salud, contemplando que la información sea íntegra y confiable para cualquier análisis que se realice con la información, impulsa a la creación de nuevos sistemas que ayuden a realizar estas tareas de extracción, es por eso que son necesarios sistemas para tareas específicas como lo son la identificación de entidades y la extracción de relaciones semánticas. En el sector de la salud es de vital importancia tener información de forma organizada, accesible y que la información sea fiable, es por ello que una adecuada extracción de información es importante para esta área.

1.5. Metodología

Para este trabajo se plantean las siguientes etapas:

1. Pre-procesamiento de la información.
2. Arquitectura de solución para la extracción de información.
3. Programación y entrenamiento de la red neuronal de aprendizaje profundo.
4. Evaluación de los resultados.

A continuación se detallan las etapas:

1. Pre-procesamiento de la información: Para el desarrollo, entrenamiento y evaluación del sistema se usarán los corpus de entrenamiento *(train)* y prueba *(test)* proporcionados por *eHealth-KD challenge 2021* [48] y *eHealth-KD challenge 2020* [50]. Además de usar la información proporcionada por *eHealth-KD challenge* en 2021 y 2020, se definirán que corpus se pueden utilizar para la creación de modelos de incrustación de palabras. Los corpus proporcionados en 2021 [50] y 2020 [48] están compuestos por un archivo con las oraciones originales como se muestra en la Tabla 4.1. También en esta etapa con el objetivo de detectar mejor las entidades y las relaciones semánticas dentro de los sistemas se plantea el análisis de diferentes etiquetadores *Part-of-speech* (*POS*),

etiquetado de dependencias y otras herramientas para la obtención de contexto global y local en una oración. Finalmente es importante probar diferentes herramientas para incrustaciones de palabras como *GloVe* o *Word2Vec*.

2. Arquitectura de solución para la extracción de información: Para esta etapa se ha elegido como propuesta inicial, el uso de redes neuronales recurrentes (más específicamente redes *Long short-term memory* (*LSTM*)), como se explora en el estado del arte, es una de las arquitecturas de redes neuronales con mejores resultados. Es importante resaltar que pese a que es un sistema de *RNN (Recurrent neural networks)* bastante usado, aún quedan métodos para su aplicación que podrían explorarse y mejorar los resultados en las tareas planteadas en esta tesis. También es necesario hacer énfasis que es más importante para este trabajo la precisión de los resultados en lugar de la velocidad con la que se obtienen. Esta arquitectura será complementada con diferentes métodos para la incrustación de palabras y con el algoritmo de *CRF (Conditional random fields)*, que demuestra dar mejoras en los trabajos expuestos en el capítulo referente al estado del campo o del arte.

   Se pretende que de forma general la misma arquitectura (*Bi-LSTM- CRF*) sea usada para la identificación de las entidades y la extracción de relaciones semánticas.

3. Programación y entrenamiento de la red neuronal artificial de aprendizaje profundo: En base a las arquitecturas que fueron seleccionados anteriormente se implementarán soluciones que cumplan con los requisitos propuestos por la tarea de *eHealth-KD challenge 2021*. Para implementación de cada red se usará como lenguaje de programación a *Python* y sus librerías correspondientes.

4. Evaluación de los resultados: La evaluación de los resultados se realizará para cada subtarea y posteriormente para el escenario principal de [50]. Se contempla obtener las medidas de *precisión*, *exhaustividad* y F_1 de cada sistema, para verificar el rendimiento para cada subtarea.

1.6. Distribución de la tesis

El Capítulo 2 de este trabajo de tesis mostrará el marco teórico, el Capítulo 3 mostrará el estado del arte que contiene los trabajos relacionados a este trabajo de tesis, posteriormente el Capítulo 4 nos mostrará la metodología usada para resolver las tareas planteadas, el Capítulo 5 nos mostrará los resultados de la implementación, y finalmente se presentarán las conclusiones.

Capítulo 2

Marco teórico

En este capítulo, se presentan y explican los conceptos teóricos usados en el presente trabajo de tesis.

A continuación, se describe de forma resumida lo que es el procesamiento del lenguaje natural, el reconocimiento de entidades nombradas, relaciones semánticas, estilos de anotación, etiquetado, incrustaciones de palabras y las arquitecturas de aprendizaje profundo más usadas.

2.1. Procesamiento de lenguaje natural

El lenguaje es una herramienta que sirve para transmitir ideas, información, opiniones y sentimientos, entre otras funciones como persuadir, pedir información, o dar órdenes [10]. Se considera lenguaje natural al lenguaje que utilizamos para comunicarnos normalmente con otras personas (por ejemplo el español y el inglés). El lenguaje natural es muy importante actualmente debido a que gracias a este podemos compartir ideas con cualquier persona del mundo. Por esa razón, con el tiempo ha surgido la necesidad de que existan más herramientas que ayuden a la comunicación entre personas e incluso exista comunicación entre un humano y una computadora usando el lenguaje natural.

El procesamiento de lenguaje natural (PLN) o en inglés *Natural Language Processing*, es el área de Inteligencia Artificial que permite el estudio y análisis de tareas específicas en un lenguaje natural dado [31].

Modelos dentro del procesamiento de lenguaje natural

Dentro del procesamiento de lenguaje natural existen modelos con los que se trabaja actualmente: el modelo lógico y el probabilístico.

Modelos lógicos En los modelos lógicos (gramáticas) los lingüistas escriben reglas de patrones estructurales, empleando un formalismo gramatical concreto. Estas reglas, junto con la información almacenada en diccionarios computacionales, definen los patrones que hay que reconocer para resolver una tarea, ya sea buscar información, traducir, etc. Estos modelos lógicos surgen a partir de las teorías de N. Chomsky y pretenden reflejar la estructura lógica del lenguaje [1].

La estructura del lenguaje se analiza en diferentes niveles, el tipo de análisis a elegir depende del objetivo con el que se usarán. Los tipos de análisis son los siguientes:

- Morfológico o léxico: Este análisis consiste en el análisis interno de las palabras que forman oraciones para extraer lemas, rasgos flexivos, unidades léxica compuestas [1].
- Sintáctico: Este análisis consiste en el análisis de la estructura de las oraciones de acuerdo con el modelo gramatical usado [1].
- Pragmático: Este análisis incorpora el análisis del contexto de uso a la interpretación final [1].
- Semántico: Este análisis proporciona la interpretación de las oraciones [1].

Modelos probabilísticos Los modelos probabilísticos del lenguaje natural están basados en datos, esto quiere decir que los lingüistas recogen colecciones de ejemplos y datos (corpus) y a partir de ellos se calculan las frecuencias de diferentes unidades lingüísticas (letras, palabras, oraciones) y su probabilidad de aparecer en un contexto determinado. Calculando esta probabilidad, se puede predecir cuál será la siguiente unidad en un contexto dado, sin necesidad de recurrir a reglas gramaticales explícitas. Es decir, los algoritmos obtienen las posibles respuestas a partir de los datos observados en el corpus [1].

Dentro del procesamiento de lenguaje natural existen una gran cantidad de tareas a resolver, pero esta tesis esta dirigida a dos tareas principales la cuales son: Identificación de entidades y extracción de relaciones semanticas.

Etiquetado gramatical

Una herramienta muy ocupada en el procesamiento de lenguaje natural es el etiquetado gramatical *POS* (*part-of-speech tagging* en inglés) que consiste en asociar cada palabra de una oración a una categoría gramatical. Se entiende como categorías gramaticales a las unidades lingüísticas que consituyen la parte esencial de la gramatica en su faceta morfológica, [40].

El etiquetado gramatical cumplirá una función muy importante en las tareas de identificación de entidades y extracción de relaciones semanticas.

Existen varios procesos de etiquetado gramatical, entre los más conocidos estan *eagles* y *penn treebank* [54], los cuales son usados por herramientas etiquetadoras como *freeling* o *spacy*.

Spacy usa como base *penn treebank* para su etiquetado, es importante resaltar que usa el mismo etiquetado para textos en inglés y para textos en español, por otro lado *Freeling* usa *penn treebank* para etiquetado de texto en inglés y *eagles* para textos en español, en la Tabla 2.1 podemos ver algunas de las siglas y el significado para el etiquetado *penn treebank* que se usa en *spacy*.

2.1.1. Identificación de entidades

Dentro del área de procesamiento del lenguaje natural existe una tarea llamada “identificación de entidades", identificar entidades consiste en detectar entidades dentro de textos escritos en un lenguaje natural. En el área de informática se le considera como una entidad a una representación de un objeto del mundo real (como un concepto, persona, producto, etc.), del cual se almacena información [17]. Sin embargo, como se ha mencionado en el capítulo 1 para este trabajo de tesis abordaremos los siguientes tipos de entidades, las cuales han sido planteadas en *eHealth-KD challenge 2021* [50]:

- Concepto: Representación mental de un objeto, hecho, cualidad, situación, etc.

Tabla 2.1: Etiquetado *POS spacy*

Descripción	**Etiquetado POS**
Adjetivo	ADJ
Adposición	ADP
Auxiliar	AUX
Conjunción	CONJ
Conjunción de coordinación	CCONJ
Interjección	INTJ
Número	NUM
Partícula	PART
Pronombre	PRON
Nombre propio	PROPN
Verbo	VERB
Símbolo	SYM

- Acción: Hecho, acto u operación que implica actividad, movimiento o cambio y normalmente un agente que actúa voluntariamente, en oposición a quietud o acción no física.

- Predicado: Parte de la oración gramatical en la que se dice o se predica algo del sujeto; es un sintagma que está formado por un verbo, que es el núcleo, y unos complementos.

- Referencia: Explicación o relación de un acontecimiento que se hace de palabra o por escrito.

Para complementar la información proporcionada en el capítulo 1 acerca de esta tarea, es necesario resaltar que la tarea de identificación de entidades, es una tarea de extracción de información que tiene como objetivo encontrar y clasificar los tipos de entidades anteriormente mencionadas.

Para resolver la tarea de identificación de entidades existen varias herramientas, entre estas herramientas estan las arquitecturas en aprendizaje profundo. Las principales arquitecturas en aprendizaje profundo que han sido utilizadas en la tarea están dadas por *Convolutional Neural Network (CNN)* (Redes Neuronales Convolutivas), *Recurrent Neural Network (RNN)* (Redes Neuronales Recurrentes) y recientemente con redes neuronales tipo *transformer*.

2.1.2. Extracción de relaciones semánticas

Dentro del área de procesamiento del lenguaje natural existe una tarea llamada "extracción de relaciones semánticas", como su nombre lo menciona, se orienta principalmente a la semántica. La semántica estudia el significado de las unidades lingüísticas (palabras), y las diversas relaciones de sentido que se establecen entre ellas [22]. Una relación semántica es la relación que existe entre dos entidades considerando su significado, la tarea de extracción de relaciones semánticas consiste en obtener y clasificar las relaciones semánticas que hay entre las entidades previamente encontradas.

Es importante conocer que dentro de las relaciones semánticas existen una serie de fenómenos semánticos [51]:

Sinonimia: Dos o más palabras son sinónimas cuando tienen distinto significante y el mismo o parecido significado. Los sinónimos pertenecen a la misma categoría gramatical. Muy raramente existe la sinonimia total, es decir, sinónimos que sean totalmente idénticos (ejemplo: oculista / oftalmólogo). En cambio la sinonimia parcial es habitual; son palabras que, aunque tengan un mismo significado, presentan matices diferentes (ejemplo: alterado, conmocionado, inquieto, excitado, trastornado, enfadado, irritado, etc.) [51].

Hiperónimos: Palabras que por su mayor extensión significativa incluyen a otras (ejemplo: flor incluye amapola, clavel, tulipán, etc. que serían hipónimos)[51].

Hipónimos: Palabras que tienen una extensión más reducida y concreta (ejemplo: león, cabra, elefante pertenecen al hiperónimo animales)[51].

Meronimia – Holónimia: La relación meronimia también se conoce como relación parte-todo y partonimia, y se refiere a la relación entre un concepto o entidad y sus partes constituyentes. La distinción entre meronimia e hiponimia son claras para conceptos concretos pero difusas para conceptos abstractos. Se puede decir que las relaciones de hiponimia existen dentro de los conceptos, mientras que las relaciones de meronimia son entre conceptos [51].

Sin embargo, dentro de la tarea de extracción de relaciones semánticas se definen de forma adicional otros tipos de relaciones semánticas [50], por ejemplo; causa-efecto, tiene-propiedad, en-contexto, entre otras que se han mostrado en el planteamiento del problema.

2.1.3. Estilo de la anotación (Identificación de entidades)

El estilo de la anotación es un aspecto relevante a tomar en cuenta en la tarea de identificación de entidades, nos da una pauta de como se manejará el número de clases, ayuda a la red a identificar los elementos de una entidad (inicio, centro, fin), considerando que cada entidad se compone de una o más palabras y marcar las pautas para posteriormente obtener los resultados de la red en un formato deseado. Algunos de los estilos de anotación más comúnmente empleados en el reconocimiento de entidades son [31]:

- IO (Tjong Kim Sang y De Meulder, 2003): Diferenciando entre la anotación dentro de una entidad (*Inside*) y fuera de ella (*Outside*)
- BIO (Tjong Kim Sang, 2002): Anotando el inicio de la entidad (*Beginning*), dentro (*Inside*) y fuera de ella (*Outside*) .
- BILOU (Ratinov y Roth, 2009): Anotando el inicio de la entidad (*Beginning*), dentro (*Inside*), fin (*Last*) y fuera de ella (*Outside*). También cuando la entidad se compone de un solo elemento (*Unique*).

Estos estilos de anotación son los mas usados, sin embargo, no son los únicos existentes y se pueden crear variaciones dependiendo de las necesidades existentes.

2.1.4. Dependencias universales (Extracción de relaciones)

El etiquetador *spacy* utiliza las dependencias universales (*Universal Dependencies*(UD)) para generar sus etiquetas de dependencia, UD es un proyecto internacional abierto para crear bancos de árboles de varios idiomas [29]. Su objetivo principal es lograr coherencia de las anotaciones entre diferentes idiomas, tiene sus orígenes en tres proyectos relacionados: *Dependencias de Stanford* [27], etiquetas universales de parte del discurso de Google [47], e *interset interlingua*. Este esquema de anotación produce análisis sintácticos de oraciones en términos de las dependencias de la gramática de dependencia [29]. Algunas etiquetas usadas para este esquema de anotación son las siguientes:

- NSUBJ - Sujeto nominal.

- IOBJ - Objeto indirecto.
- OBJ - Objeto directo.
- CASE - Marcador de caso.
- ROOT - Palabra raíz.

2.1.5. Marcado descriptivo (Extracción de relaciones)

El lenguaje de marcado, específicamente el marcado descriptivo, este surge en los años 70 en IBM, el concepto de marcado descriptivo alcanza en la década de los 80 la categoría de estándar con la definición del lenguaje SGML (ISO/IEC IS 8879 *Standard Generalized Markup Language*). SGML permite definir lenguajes de marcas apropiados para diferentes aplicaciones específicas de procesamiento de texto. Además, para aumentar su capacidad expresiva, SGML también utiliza atributos asociados a los símbolos terminales. El marcado descriptivo describe, mediante las marcas o etiquetas definidas, la estructura lógica de la información [2], esta herramienta se usará para la extracción de relaciones, describiendo con marcas el etiquetado de dependencias.

2.2. Aprendizaje automático

En la actualidad, uno de los términos más escuchados cuando se habla de computación es la inteligencia artificial, la inteligencia artificial se refiere a sistemas que imitan la inteligencia humana para realizar tareas [44]. El aprendizaje automático o *Machine Learning* (*ML*) es una subsección de la inteligencia artificial (IA), su principal objetivo es desarrollar sistemas que aprendan en función de los datos que consumen (usan algoritmos para identificar patrones en los datos y a partir de esos patrones crean modelos que pueden hacer predicciones). La adaptabilidad del aprendizaje automático lo hace una buena opción en escenarios con múltiples variables en sus datos. Algunas de las ventajas que ofrece el aprendizaje automático son [43]: Descubrir información, mejorar la integridad de los datos, mejorar la experiencia del

usuario, reducir riesgos, reducir costes, etc. Existen 3 técnicas principales usadas en el aprendizaje automático:

- Aprendizaje supervisado: Aborda conjuntos de datos estructurados para aprender y posteriormente realizar una predicción.
- Aprendizaje no supervisado: Busca patrones y relaciones en conjuntos de datos no estructurados.
- Aprendizaje reforzado: Aprende información de un bucle de comentarios dados.

2.3. Aprendizaje profundo

El Aprendizaje profundo o también conocido como *Deep learning*, es una subcategoria del aprendizaje automático y consisten en el uso de redes neuronales artificiales (*artificial neural networks*) para resolver tareas como lo es el procesamiento del lenguaje natural (PLN) entre otras [38], [8]. Las redes neuronales artificiales (*ANN*), son algoritmos que simulan el funcionamiento de las redes neuronales convencionales, consisten en capas de nodos, entre las cuales se encuentran la capa de entrada, capas ocultas y una capa salida. Cada nodo se conecta a otro y tiene un valor de peso y valor de umbral asociado, si el valor umbral se supera en un nodo, este nodo enviará datos al siguiente nodo (que se encuentra en la siguiente capa de la red). Las redes neuronales artificiales siempre dependen de un buen conjunto de datos para su entrenamiento [21].

En la actualidad existen varios tipos de arquitecturas destinadas para el aprendizaje profundo, que se usan tanto para las tareas de reconocimiento de entidades como para la extracción de relaciones semánticas, a continuación se describen las más comunes [23]:

2.3.1. Redes neuronales convolucionales

Las redes neuronales convolucionales (*CNN*) se utilizan con mayor frecuencia para tareas de clasificación y visión. Antes de las *CNN*, se utilizaban métodos manuales de extracción de características que consumían mucho tiempo para identificar objetos

en imágenes. Sin embargo, las redes neuronales convolucionales brindan un enfoque más escalable para las tareas de clasificación de imágenes y reconocimiento de objetos, aprovechando la multiplicación de matrices, para identificar patrones dentro de una imagen. Pueden ser computacionalmente exigentes, requiriendo unidades de procesamiento gráfico (*GPU*) para entrenar estos modelos. Las *CNN* se distinguen de otras redes neuronales por su rendimiento superior con entradas de señales de imagen, voz o audio. Tienen tres tipos principales de capas, que son:

- Capa convolucional: (La convolución consiste en filtrar una imagen usando una máscara [33]) Esta es la capa principal de las redes convolucionales, su uso es indispensable [12].

- Capa de agrupación: Esta capa toma los mapas de características producidos en la capa de convolución y los agrupa. Esta capa produce la reducción de dimensionalidad, ayudando a evitar el sobreajuste del modelo [12].

- Capa completamente conectada (*FC*): Es como cualquier capa de una red neuronal clásica, pero tiene su arquitectura completamente conectada, su función principal es llevar a cabo una agrupación de la información obtenida hasta ese momento [12].

La aplicación de *CNN* para el procesamiento del lenguaje natural no es tan común. Sin embargo, es importante darle una mención debido a trabajos como [53] donde se utilizan este tipo de redes.

Otros tipos de capas indispensables en redes neuronales convolucionales y en otros tipos de redes neuronales son los siguientes:

- Capa *ReLu*: La capa de unidades lineales rectificadas (*Rectified Linear Units Layer*) es un tipo de capa muy común, su función es la de aumentar la propiedad de no linealidad de la función de activación [12].

- Capa de pérdida (*Loss layer*): En esta capa se comparan las predicciones con los valores reales, sus funciones estan dadas por la función de pérdida *softmax* y la función de pérdida euclídea [12].

2.3.2. Redes neuronales recurrentes

Las redes neuronales recurrentes (*Recurrent Neural Networks, RNN* por sus siglas en inglés) se llaman recurrentes porque realizan la misma tarea para cada elemento de una secuencia, y la salida depende de los cálculos previos. También las *RNN* tienen una memoria que captura la información que entra, pero con ciertas limitaciones en la cantidad de datos que se retienen en la memoria. Actualmente se han desarrollado distintas mejoras en las RNN que permiten hacer frente a las decadencias del modelo original [23]. Existen tres versiones principales de *RNN*:

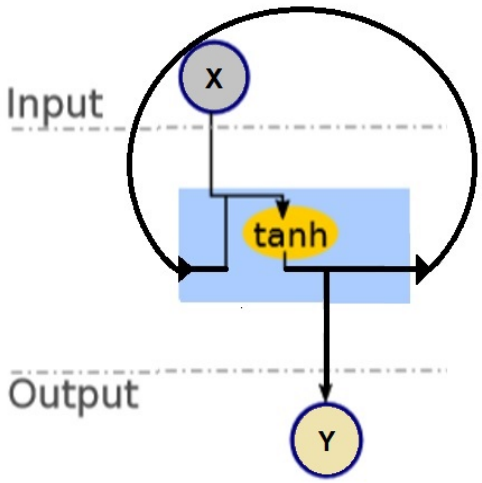

Figura 2.1: Ejemplo versión simple de *RNN* (*vanilla*) [23].

- La versión simple (Llamada en inglés *vanilla* [23]): Tiene una estructura muy simple con una sola capa *tanh* (que es la función de activación), con valores entre -1 y 1 [23], como se puede observar en la Figura 2.1.
- *LSTM (Long Short Term Memory)*: Es un tipo de red neuronal recurrente principalmente capaz de retener información por largos periodos de tiempo, olvidar información innecesaria y exponer cuidadosamente la información en cada paso del tiempo [23].
- *GRU (Gated Recurrent Units)*: Es un tipo de red neuronal recurrente similar

a las *LSTM*, pero utiliza dos puertas solamente (una de actualización y una de reinicio).

La principal diferencia entre versiones de redes neuronales recurrentes radica en la arquitectura de la unidad de memoria.

Las redes *LSTM* son un tipo de redes neuronales recurrentes que consisten de múltiples puertas y son capaces de aprender dependencias a largo plazo debido a que su estructura tiene la capacidad de recordar información por periodos largos de tiempo, olvidar información innecesaria y exponer cuidadosamente la información en cada paso del tiempo. En la Figura 2.2 se muestran las etapas de una neurona *LSTM*. Estas son: 1) Capa de olvido, 2) Capa de puerta de enlace y valores candidatos, 3) Nueva celda y 4) Nueva salida.

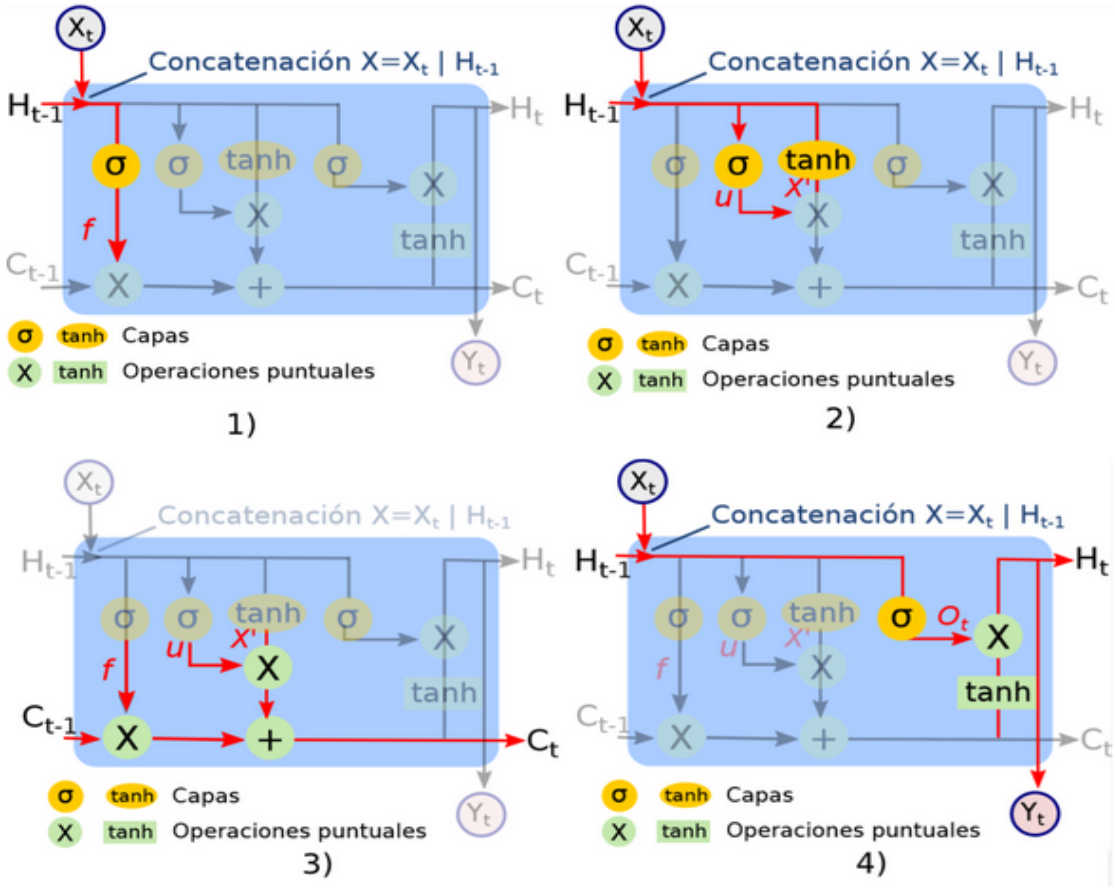

Figura 2.2: Ejemplo etapas en la arquitectura de una neurona *LSTM* [23].

Las redes *BiLSTM* son una variante bidireccional de las redes *LSTM*, se compone de dos capas de neuronas (una capa que aprende representaciones previas y otra que retrocede en el el tiempo) como se puede observar en la Figura 2.3, básicamente entrenan dos *LSTM* en lugar de uno en la secuencia de entrada, y aunque el uso de modelos basados en redes neuronales recurrentes llega a necesitar de una mayor cantidad de información para funcionar mejor y el entrenamiento de las mismas suele ser tardado, se ha podido comprobar que tiene buenos resultados.

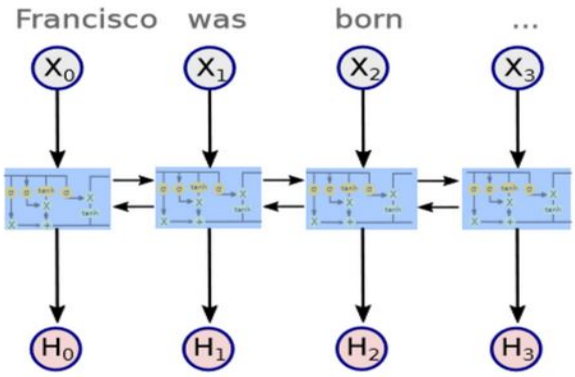

Figura 2.3: Ejemplo red *BILSTM* [23].

2.3.3. Redes *Transformers*

Por ultimo, las redes neuronales tipo transformer son redes neuronales recientes diseñadas para secuencias, basadas en la autoatención, han demostrado estar bien adaptadas a tareas relacionadas con el texto. Como se muestra en la Figura 2.4 este tipo de redes tienen como principal innovación la sustitución de las capas recurrentes, como las *LSTM*, por las denominadas capas de atención. Estas capas de atención codifican cada palabra de una frase en función del resto de la secuencia, permitiendo así introducir el contexto en la representación matemática del texto, motivo por el cual a los modelos basados en *Transformer* se les denomina también incrustaciones contextuales. [57] La arquitectura de *Transformer* incluye otras innovaciones, como las incrustaciones posicionales, que permiten al algoritmo conocer la posición relativa de cada palabra del texto.

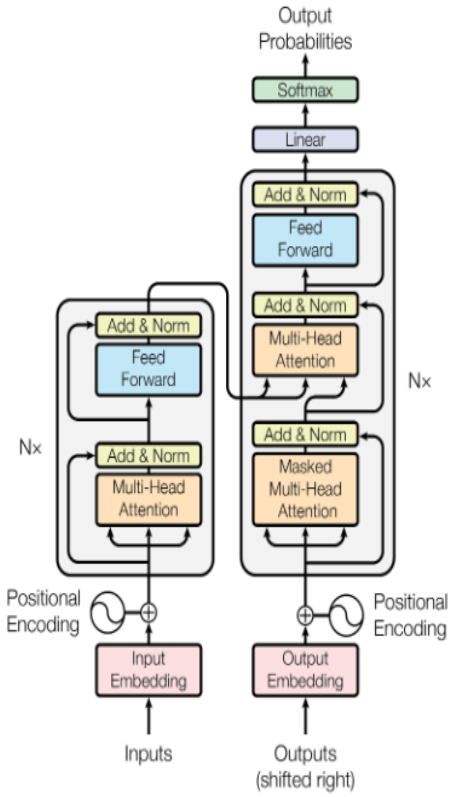

Figura 2.4: Ejemplo de red *Transformer* [57].

Un ejemplo del uso de este tipo de redes es *BERT (Bidirectional Encoder Representations from Transformers)* que ha tenido un gran impacto en estos últimos años en el dominio del Procesamiento del Lenguaje Natural, eso incluye las tareas de reconocimiento de entidades y las de extracción de relaciones. También se menciona en [55] que se descubrió que estos modelos no pueden funcionar tan bien como en el dominio general cuando no hay información específica de dominio integrada.

BERT es un modelo de representación de lenguaje, *BERT* está diseñado para pre-entrenar representaciones bidireccionales profundas de texto no etiquetado por condicionamiento conjunto en todas sus capas [20]. Existen múltiples modelos pre-entrenados basados en BERT para diferentes idiomas, estos modelos se encuentran en repositorios como lo es "huggingface.co".

2.3.4. Incrustaciones de palabras en el aprendizaje profundo

Las tareas de identificación de entidades y extracción de relaciones semánticas han sido abordadas con varios grupos de modelos de aprendizaje profundo (*Deep Learning*), empleando principalmente características a dos niveles de granularidad (aprendizaje sin supervisión):

1. Nivel palabra: Empleando incrustaciones de palabras.

2. Nivel carácter: Empleando incrustaciones de caracteres.

Las incrustaciones de palabras (*word embeddings*) es el nombre para un conjunto de técnicas de modelado de lenguaje y aprendizaje de características en el (Procesamiento del lenguaje natural) PLN donde las palabras o frases del vocabulario se asignan a vectores de números reales. La transformación de palabras a números es necesaria porque las redes neuronales de aprendizaje profundo requieren en su entrada vectores de valores continuos, en lugar de cadenas de texto sin formato [11]. La clave del enfoque es la idea de usar una representación distribuida densa para cada palabra. Cada palabra está representada por un vector de valor real, a menudo decenas o cientos de dimensiones. Esto contrasta con las miles o millones de dimensiones requeridas para representaciones de palabras escasas, tales como la codificación *one-hot*. Existen diferentes técnicas para aprender a incrustar una palabra a partir de datos de texto. Algunas de las más usadas son las siguientes [56]:

Capa de inserción: Es una incrustación de palabras que se aprende conjuntamente con un modelo de red neuronal, en tareas como el modelado del lenguaje o la clasificación de documentos [56].

Word2Vec: Es un método estadístico para aprender una palabra independiente incrustada de un corpus de texto. Existen dos modelos de aprendizaje diferentes que pueden ser utilizados como parte del enfoque de *Word2Vec* [56]:

- Bolsa de Palabras Continua, o modelo *CBOW*.

- Modelo Continuo *Skip-Gram*.

GloVe: Algoritmo de Vectores Globales para la Representación de Palabras ó *GloVe*, es extensión de Word2Vec. Fue desarrollado usando técnicas de factorización

matricial tales como el Análisis Semántico Latente (*LSA*, por sus siglas en inglés) son buenas en el uso de estadísticas globales de texto pero no tanto como los métodos aprendidos como *Word2Vec* [56].

FastText: Creado por *Facebook* ahora *Meta Platforms, Inc.*, es una librería para incrustación de palabras liviana, gratuita y de código abierto. Permite crear un algoritmo de aprendizaje no supervisado o de aprendizaje supervisado para obtener representaciones vectoriales de palabras. *Facebook* pone a disposición modelos previamente entrenados para 294 idiomas [56].

La herramientas como *FastText* y *Word2Vec*, nos permiten crear modelos propios de incrustaciones de palabras a partir de un corpus, generalmente estos modelos funcionarán mejor mientras más palabras tenga el corpus que se utilizará para crearlos.

2.3.5. Campos aleatorios condicionales en el aprendizaje profundo

Los campos aleatorios condicionales (*Conditional random fields*, *CRF* por sus siglas en inglés) es un modelo estocástico basado en el modelo oculto de Márkov (*Hidden markov model*, *HMM* por sus siglas en inglés), fue uno de los mejores modelos para el etiquetado, segmentación de datos secuenciales y para extraer información de documentos. Actualmente el modelo *CRF* por si solo ya no es tan eficiente y el aprendizaje profundo (*Deep Learning*) lo ha reemplazado. La principal debilidad de *CRF* es que son incapaces de modelar la similitud semántica entre dos palabras. Para superar este problema, muchos *CRF* se basan en *gazetters* (lista de personas, organizaciones, lugares, etc.). Sin embargo, esto es una solución sub-óptima ya que los *gazetters* están intrínsecamente limitados y pueden ser muy costosos para desarrollarse [31]. Sin embargo, existen modelos que combinan *CRF* con redes neuronales recurrentes como en [34] y que han obtenido buenos resultados.

Capítulo 3

Estado del arte

En este capítulo se describen los trabajos encontrados en *eHealth-KD challenge 2021* [48] y *eHealth-KD challenge 2020* [50]. Los sistemas se clasifican según la subtarea que realiza, contemplando primero la subtarea A (identificación de entidades), después la subtarea B (extracción de relaciones) y finalmente, el escenario principal donde se contemplan ambas las subtareas anteriormente mencionadas. También se han recopilado los artículos que no han participado en *eHealth-KD challenge*, contemplando también trabajos que van orientados a análisis de textos en inglés.

3.1. Sistemas para reconocimiento de entidades

En 2017 encontramos diversos artículos ajenos a *eHealth-KD challenge*, que están relacionados con el reconocimiento de entidades como [42] que explora características de corpus en español para el reconocimiento de entidades. En [15] podemos observar una implementación donde se usan campos aleatorios condicionales en la identificación de entidades, combinándolo con técnicas de aprendizaje automático, técnicas de pre-procesamiento semántico y una red semántica. Por último, en 2017 en [58] observamos diversos modelos para el reconocimiento de entidades, concluyendo que uno de los modelos que usaba redes neuronales concurrentes obtuvo los mejores resultados bajo la métrica F_1 (85.91 %). Dentro de esta investigación se usaron dos corpus (*i2b2 2010* y *MIMIC II*) y el mejor modelo implementado fue una red neuronal *BiLSTM*, con varias técnicas estándar de aprendizaje profundo, en la capa de

entrada, se combinó las incrustaciones de palabras y las incrustaciones de caracteres en un vector de entrada. Las incrustaciones de palabras se entrenaron previamente a partir del corpus *MIMIC II* y las incrustaciones de caracteres se iniciaron con valores aleatorios. La capa de clasificación final utilizó una función de pérdida de *CRF*.

Más tarde, en 2019 en [18] se presenta un sistema de reconocimiento de entidades en español para el dominio biomédico. El cual utiliza como corpus un conjunto de reportes de radiología. Este trabajo compara el algoritmo *SiMREDA (Simple Entity Detection Algorithm for Medium Resources language)* con el algoritmo de *CRF*, donde la entrada es la secuencia de tokens del texto. Las características de un token y el patrón de etiquetas asignadas a palabras anteriores se utilizan para determinar la etiqueta más probable para el token actual. Para la evaluación se utilizó el mismo dataset y se dividió de tal forma que el 80 % fuera para el desarrollo y el 20 % para la evaluación. Los resultados arrojaron que el algoritmo *CRF* obtuvo mejores resultados.

En 2020, orientándose al dominio biomédico y usando textos en inglés, [55] presenta una mejora para un modelo *BERT* (*Bidirectional Encoder Representations from Transformers*) llamada *BioBERT*. Modelos entrenados como *ELMo* (*Embeddings from Language Models*) y *BERT* lograron un rendimiento de vanguardia en muchas tareas de PNL en el dominio general. Por lo tanto, se aplicaron a *Biomedical named entity recognition (BioNER)*, sin embargo, se descubrió que estos modelos no pueden funcionar tan bien como en el dominio general cuando no hay información específica de dominio integrada. *BioBERT* está pre-entrenado en grandes cuerpos biomédicos en inglés sin procesar y logra un rendimiento de vanguardia en *BioNER*. En [55] se usó información sintáctica como etiquetas *POS*, relaciones de dependencia y *KVMN (Key-Value Memory Networks)* que asigna el mayor peso a las "anomalías" y su información sintáctica portadora de "*NP*" (*Noun Phrase*) entre todas las demás instancias sintácticas, ya que podrían ser señales fuertes para los nombres de enfermedades. *KVMN* puede ponderar dicha información sintáctica de acuerdo con la importancia de las características del contexto y así, hasta cierto punto, evitar los errores causados por la incorporación de información sintáctica auto-procesada. Para la evaluación se usaron 6 diferentes corpus de conjuntos de datos en inglés y los métodos que usaron *KVMN* obtuvieron mejores resultados en la mayor parte

de los casos. También en 2020, en [3] se presenta un modelo basado en *BERT* y también orientado al dominio biomédico en español, el modelo usa como corpus las estadísticas sobre *The Spanish Clinical Case Corpus* (*SPACCC*). También hace uso de una pérdida de campos aleatorios condicionales. Este modelo fue entrenado con 128,000 elementos, el modelo extrae 4 diferentes tipos de entidades: normalizables, no normalizables, proteínas y menciones de sustancia general. En la evaluación, solo se utilizan los primeros 3 tipos de entidades. Para clasificar entidades, se usó un esquema *BIO ([B] egining, [I] nside, [O] outside)*.

En [53] se presenta un modelo de dos partes, una red de auto-atención multidimensional (*MDSA*) y un modelo basado en redes neuronales convolucionales (*CNN*). El mecanismo de atención tradicional solo calcula una puntuación para cada vector de palabra, y la atención multidimensional cambiará este resultado. El modelo usa el vector de palabras procesado por el mecanismo de atención como la entrada de red de la *CNN* de cuatro capas. Después de la capa de red *CNN*, se obtiene el vector de salida correspondiente al vector de entrada uno a uno. Los resultados de la capa *CNN* serán procesados por la capa de salida. Los resultados calculados representan la distribución de etiquetas para cada ubicación. La capa de salida contiene la capa completamente conectada, la función *softmax* y el método de abandono. Para este modelo se usaron dos corpus y *GloVe 840B* para la incrustación de palabras inicial. Este modelo muestra los mejores resultados de todos los modelos analizados en esa investigación. *MDSA-CNN* obtuvo mejores resultados que *Bi-LSTM-CNN-CRF* un método principal en *NER*. En [32] se presenta un modelo de identificación de entidades para cualquier dominio en general. Este modelo combina una red *LSTM* bidireccional y campos aleatorios condicionales (*BiLSTM*-CRF) agregando *Multi-Task Learning (MTL)* y un *framework* llamado *Mixture of Entity Experts (MoEE)*. En este trabajo se usa información de *CoNLL-2003 English NER*, las entidades encontradas se clasifican en 4 tipos: personas, ubicaciones, organizaciones y diversos. Se hicieron experimentos con diferentes incrustaciones de palabras y se evaluaron los resultados con la métrica F_1 de los cuales el modelo que mejor obtuvo resultados fue *BiLSTM-CRF* con *MTL* and *MoEE* usando *Fast-Text (freeze settings)* con 69.53 % y el segundo mejor resultado fue con 68.59 % usando un modelo pre entrenado con *BERT*.

En 2021 tenemos a [59] que está orientado al dominio biomédico en inglés usando re-corrección de etiquetas y destilación de conocimientos para obtener un modelo de reconocimiento de alto desempeño. La investigación usa dos modelos base que serían *BioBERT-CRF* y *BiLSTM-CRF* a los cuales se les aplican las estrategias de corrección de etiquetas y la destilación de conocimientos para comprimir la información en los modelos. En ambos casos y con distintos conjuntos de información, la re-corrección de etiquetas da mejoras a ambos modelos.

Durante *eHealth-KD challenge* en 2019 en [49] podemos ver que [24] propone para el reconocimiento de entidades un sistema que consiste en una fase de pre-procesamiento, donde los datos de entrada se adaptan y preparan, un modelo de aprendizaje profundo supervisado y un paso de pos-procesamiento para resolver sistemas errores a través de reglas hechas a mano (esquema de anotaciones *BILOU*). Los diferentes atributos considerados como entrada de la pila de aprendizaje profundo son (palabras, parte del discurso, carcasa) consta de una capa *BiLSTM* seguida de dos capas densas. Las entradas del modelo están representadas por los vectores, que representan incrustación de etiquetas *POS* e incrustación de palabras, respectivamente. La última capa densa corresponde a la capa de salida. En [24] se han aplicado dos tipos de reglas al resultado del aprendizaje profundo: El primer conjunto de reglas está orientado a corregir errores frecuentes, ampliando o reduciendo el alcance de una frase clave detectada, o modificando su tipo, según la carcasa y la etiqueta *POS* información. Por otro lado, el segundo conjunto de reglas apunta a asegurar que la salida final del sistema sigue correctamente el formato de salida *BILOU*.

En [5] se propone un sistema con una estrategia que utiliza un modelo híbrido que combina una memoria bidireccional larga corta capa (*BiLSTM*) con una capa *CRF* para la identificación de entidades. Además, se utilizan incrustaciones de palabras específicas del dominio. Para cada token en el que se dividió la oración, la entrada para ese token consta de una lista de tres vectores de características: Codificaciones de caracteres, vectores de etiquetado *POS*, índices de palabras. La arquitectura está conformada por cuatro componentes principales: Matriz de incrustación de palabras, *BiLSTM* de incrustación de caracteres, *BiLSTM* a nivel token y un clasificador *CRF*. La capa *CRF* produce una secuencia de etiquetas en el sistema de etiquetado

BMEWO-V. Esta clasificación de etiquetado la "B" corresponde para el inicio de una frase clave, "M" para el contenido de en medio, "E" para el final, "W" para los tokens que son una frase clave en sí mismos y "O" para los tokens que no representan nada. En [9] para la detección de entidades se propuso una arquitectura basada en un *BiLSTM* con un clasificador *CRF* como la última capa de la red. Usando dos capas *BiLSTM* con 100 unidades recurrentes con el optimizador de Adam y una probabilidad de abandono ingenua de 0.25. Las clases correspondientes están codificadas en el esquema de etiquetado de secuencia estándar de *BIO*. Los tokens se introducen en la red como incrustaciones de 1024 dimensiones obtenidas promediando las tres capas de salida de un modelo *ELMo (Embeddings from Language Models)* (pre-entrenado en español). Como los modelos oficiales de *Allen NLP ELMo*7 solo están disponibles para inglés, se usaron modelos *ELMo* pre-entrenados en español.

Durante *eHealth-KD challenge* en 2020 [46] se plantea la subtarea como un problema de predicción de etiquetas que toma el texto sin formato de la entrada sentencia y genera dos secuencias de etiquetas independientes: una en el sistema de etiquetas *BMEWO-V* para predicción de entidad, y otra con etiquetas correspondientes a los tipos de entidad (concepto, acción, referencia y predicado) para fines de clasificación. Utiliza capas *BiLSTM* como codificadores contextuales. La etiqueta "Ninguno" se incluye en este último para casos en los que no hay entidad presente. En [7] se propone el reconocimiento de entidades nombradas con un *BiLSTM* basado en personajes, una etiquetadora de secuencia capacitada en el conjunto de capacitación proporcionado por los organizadores. El sistema empleado para *NER* se compone de tres principales componentes; un modelo de lenguaje basado en caracteres (LM) que genera representaciones de palabras contextuales basadas en caracteres. En la parte superior de esta capa *LM*, una capa *BiLSTM* captura las dependencias secuenciales entre las palabras de la secuencia de entrada. Finalmente, una capa *CRF* para manejar la inferencia de etiquetado. En [16] el sistema usa el etiquetado $BILUOV$, utiliza una capa de codificación de caracteres para transformar la representación de los caracteres de cada token en un único vector que captura las dependencias morfológicas del token. La capa de codificación de caracteres consta de una capa de incrustación, seguida de una capa *BiLSTM*. Se entrenan cuatro instancias independientes con esta arquitectura, cada una correspondiente a un tipo de entidad.

Finalmente, en *eHealth-KD challenge* en 2021 [41] obtiene el cuarto lugar con una solución basada en *BiLSTM* (solamente detrás de modelos basados en *BERT*) dirigida a la incrustación de palabras y caracteres, también hace uso de información *POS* de *spacy* y etiquetado *BILUOV*.

En la Tabla 3.1, se muestra una comparativa entre sistemas para el reconocimiento de entidades entre participantes del *eHealth-KD challenge*.

Como se muestra *eHealth-KD challenge* provee información para el desarrollo, un corpus de entrenamiento y un corpus de prueba. Esta información es extraída de corpus de *MedlinePlus*, *Wikinews*, y documentos relacionados con tópicos de salud, pero mostrando una gran variedad en términos de formato y estructura.

3.2. Sistemas para extracción de relaciones

Respecto a la extracción de relaciones en 2019, el sistema propuesto por [24] consta de una capa *BiLSTM*, pero se enriquece con la adición de una capa de atención, es necesario pre-procesar la entrada al igual que el modelo propuesto por [24] para la identificación de entidades. Las características de entrada son: Entidades y grafos de dependencia. El modelo hace uso de una capa *BiLSTM* seguida de una capa de atención que considera la salida de la capa anterior como un todo y fusiona la información de la palabra en un vector en un nivel superior, lo que intenta representar las relaciones de atención a nivel oración. La salida de la capa de atención se dirige luego a una capa densa de salida. La entrada se modela utilizando incrustaciones, a nivel palabra, etiquetas *POS* e información de entidad respectivamente y vectores *One-Hot*, que contienen información sobre la dependencia entre términos, tipo de dependencia e información de carcasa, respectivamente. Otra manera de usar *BiLSTM* es en [5], donde en su sistema para la extracción de relaciones se propone un clasificador multiclase, tomando como entrada una secuencia de vectores de los tokens en la ruta de dependencia más corta entre pares de frases clave. Este clasificador necesita pre-procesamiento para construir una entrada apropiada a los modelos. En este método se hace uso de la técnica de dependencia más corta. Este modelo espera una secuencia de tokens. La entrada para cada token consiste en una lista de cuatro características: Índices de palabras, vector de relación de dependencia

Tabla 3.1: Tabla comparativa entre sistemas para el reconocimiento de entidades entre participantes del *eHealth-KD challenge*.

Autor(es)	Corpus usado	Pre-procesamiento	Modelos usados	F_1 score
[24]	Corpus *eHealth-KD challenge*.	Esquema de anotaciones *BILOU*, tokenización basada en la división de espacios en blanco, y los atributos considerados: palabras, *POS*, *casing*.	Una capa *BiLSTM* seguida de dos capas densas.	0.7543
[5]	Corpus en español de Wikipedia de páginas de contenido médico (27 millones de palabras).	Incrustaciones de palabras (*Word2vec - CBOW*), cada token consta de: codificadores de carácter, vector *POS-tag*, índices de palabras.	Matriz de incrustación de palabras, *BiLSTM* de incrustación de caracteres, *BiLSTM* a nivel de token, clasificador *CRF*.	0.8156
[9]	Corpus *eHealth-KD challenge*.	Una red como incrustaciones de 1024 dimensiones, que se obtiene promediando las tres capas de salida de un *ELMo* (pre-entrenado en español).	Un *BiLSTM* con un clasificador *CRF* como la última capa de la red.	0.8167
[16]	Corpus *eHealth-KD challenge*.	incrustaciones estáticas de *FastText* previamente entrenadas e incrustaciones de estilo contextual (*es-forward + es* - al revés) entrenado con Wikipedia.	Un modelo de lenguaje basado en caracteres, una capa *BiLSTM* y una capa *CRF*.	0.794967
[7]	Corpus *eHealth-KD challenge*.	Los tokens se introducen en un modelo *BERT* para obtener incrustaciones.	Una red neuronal profunda que recibe los tokens de entrada y emite conjuntamente predicciones para varias variables de salida diferentes.	0.6918
[46]	Corpus *eHealth-KD challenge*.	Descomposición de palabras a nivel carácter, extracción de características sintácticas y análisis de dependencia. Características: Incrustación contextual (*BERT-base*), Incrustación de caracteres, etiquetado *POS*, incrustación de dependencias, *BMEWO-V* y etiquetas de tipo de entidad.	Capas *BiLSTM* como codificadores contextuales	0.814312
[41]	Corpus *eHealth-KD challenge*.	Información *POS* de *spacy* y etiquetado *BILUOV*.	*BiLSTM* dirigida a la incrustación de palabras y caracteres.	0.814312

sintáctica, codificación de etiquetas *BMEWO-V* y un tipo de codificación de frase clave. La arquitectura está conformada por tres componentes principales: La matriz de incrustación de palabras, *BiLSTM* apilados y dos clasificadores multiclase densos. El vector de incrustación se concatena con los otros tres vectores de entrada, y todos juntos sirven como entrada para cada paso de tiempo de las capas *BiLSTM* apiladas. En [9], para la extracción de relaciones, se propone un enfoque de aprendizaje multitarea (decidir si las entidades están relacionadas, identificar el tipo de relación y decidir la correcta dirección) y se organizó la información de cada relación candidata en dos escenarios: contextos globales y locales. También en [9] el sistema consta de un modelo *BiLSTM* con una capa superior de atención para cada incrustación concatenada. Este modelo captura lo más importante de la información sintáctica y semántica de cada secuencia. El contexto global se basa en el supuesto de que es más probable que una asociación entre dos entidades sea expresado entre de una de las tres secuencias: Antes-Entre: De las palabras antes y entre los dos candidatos. Entre: De las palabras entre las dos entidades candidatas. Entre-Después: De las palabras entre y después de los dos candidatos. El contexto local puede proporcionar información útil para la detección del tipo y dirección de la relación, así como la presencia de la relación misma. Este contexto también se basa en una secuencia de palabras, que contiene la información de las palabras ubicadas a la izquierda y a la derecha de las entidades candidatas. En ambos contextos, cada secuencia se representa mediante la concatenación de siguientes incrustaciones: tokens, etiquetas *POS*, tipos de entidad y dependencias.

En 2020 [46] se interpreta la extracción de relaciones como una serie de pares consultas entre las entidades presentes en la oración de destino. Por tanto, predice la existencia de una cierta relación sobre características derivadas tanto de la oración como del par de entidades. En [7] para la extracción de relaciones, se usa una estrategia de aprendizaje por transferencia, ajustando modelos existentes de idiomas pre-entrenados multilingües en los datos anotados de la tarea. Se han construido tres sistemas: *XLMem, XLMem** y *XLMem*+MTB*. Todos los modelos con arquitectura de marcadores de entidad (*XLMem*), pero difieren en las estrategias de entrenamiento y los datos. Se puede destacar que este sistema obtuvo la mejor evaluación en extracción de relaciones en *eHealth-KD challenge 2020* [48]. En [16]

el sistema se divide en tres etapas: codificación de las entidades fuente y principal, codificación de la ruta entre las entidades, y combinación de representación y predicción. Similar al etiquetador *BILUOV*, el clasificador por pares usa una capa de codificador de caracteres.

Por último, en *eHealth-KD challenge* en 2021 [41] tiene el sistema con la evaluación más alta usando una red basada en *BiLSTM*, sin embargo, su F_1 es de 0.05384. Por otro lado, [4] obtiene el tercer lugar en la subtarea y usa el etiquetado *BILUOV* e incrustaciones posicionales para codificar la distancia relativa entre cada palabra y las dos entidades objetivo en la oración. El sistema usa capas de inscrustación, junto con capas convolucionales, con una capa densa de salida. Mientras que [6] obtiene el primer lugar en la subtarea usando un sistema basado en *XLM-RoBERTa* junto con técnicas de modelado matemático.

En la Tabla 3.2, se muestra una comparativa entre sistemas para la extracción de relaciones semánticas entre participantes del *eHealth-KD challenge*.

3.3. Sistemas para ambas subtareas

A diferencia de los sistemas donde solo se resuelve una tarea; ya sea la identificación de entidades, o la extracción de relaciones, hay un conjunto de artículos donde se propone usar el mismo sistema para ambas subtareas. Por ejemplo, en 2019, [36] propuso el sistema (*Joint-BERT-RCNN*) con mejor puntuación en todas las evaluaciones en [49]. El trabajo en [36] inspira su enfoque en modelos de traducción basados en la atención, en el que la salida es generada sucesivamente por ejecutar el modelo para un token de entrada en particular a la vez. Este sistema toma un documento y un índice de token y calcula los límites y clases de la frase clave más corta a la que pertenece, y las relaciones de cada token de otras entidades. Por lo tanto, el modelo debe ejecutarse para cada token del documento de entrada. Las capas recurrentes y de convolución permiten mirar tantos contextos locales y globales de cada token de entrada. El contexto local es capturado por la salida de la capa *RNN* y la salida de la capa de convolución no agrupada, y son concatenadas para cada paso de tiempo. El contexto global es capturado por la salida de la capa de convolución agrupada. Los resultados finales son luego generados por una capa *CRF*. Se aborda la extracción

Tabla 3.2: Tabla comparativa entre sistemas para la extracción de relaciones entre participantes del *eHealth-KD challenge*.

Autor(es)	Entrada	Modelo	F_1 score	Año
[24]	Incrustaciones de palabras, etiquetado *POS* y vectores *One-Hot*. Características: Entidades, gráfico de dependencia.	Una capa *BiLSTM*, una capa de atención y una capa densa de salida.	0.7543	2019
[5]	Vectores de características: índice de palabras, *syntactyc dependency relation vector*, *BMEWO-V tag enconding*, tipo de codificación de frase clave	Matriz de incrustación de palabras, *BiLSTM* apilados, dos clasificadores multiclase densos.	0.8156	2019
[9]	Para cada par de entidades se define un contexto local y un contexto global	Un modelo *BiLSTM* con una capa de atención para cada incrustación concatenada.	0.8167	2019
[16]	Representación de caracteres, Incrustación de palabras, *POS Tag Representation*, Representación del árbol de dependencia, Representación del tipo de entidad.	Una capa de codificación de caracteres con tres capas *BiLSTM* para cada incrustación concatenada.	0.545035	2020
[46]	La entrada es la salida del sistema de reconocimiento de entidades.	Una capa *BiLSTM*, una capa *LSTM* unidireccional y un *ChildSum Tree-LSTM*.	0.59879	2020
[7]	La entrada es la salida del sistema de reconocimiento de entidades.	*XLM* con arquitectura de marcadores de entidad (*XL-Mem*).	0.633235	2020
[41]	Etiquetado *BILUOV* e incrustaciones posicionales.	Capas convolucionales.	0.31771	2021
[6]	Combinaciones de cada posible par, generando un ejemplo repetido por par, y marcadores alrededor de las entidades para evitar el sobreajuste.	*XLM-RoBERTa*.	0.4304	2021

de relaciones a nivel token. Las oraciones son primero tokenizadas, las etiquetadas son analizadas por dependencia usando *FreeLing*. Cada token se representa por un vector, que resulta de la concatenación de las características enumeradas a continuación: Codificación *One-Hot* de los campos de categoría y tipo de la etiqueta de parte del discurso del token del conjunto de etiquetas de *FreeLing*. Vector normalizado que codifica las dependencias encontradas en la ruta entre el token y el token de destino (el que se está decodificando). Codificación *One-Hot* de la distancia entre el token y el token de destino. Incrustación de palabras contextualizada del token, calculada extrayendo los pesos de la última capa de un pre-entrenado multi-idioma, de propósito general (Se usó con modelo *BERT-Base, Multilingual Cased*).

Con una técnica diferente, en [13] propone un sistema (*Voting LSTMs*) con una arquitectura *BiLSTM* apilada de 2 capas con diferentes entradas. Las estrategias de minería de datos son una construcción de un conjunto de modelos para generar predicciones y el uso de una función sustituta de la métrica F_1 como función de pérdida. Para la identificación de entidades, la entrada usa la secuencia de palabras de cada oración y su parte del discurso (*POS*) y consta de dos capas apiladas de alimentación *BiLSTM* con la incrustación de palabras y la codificación *One-Hot* del *POS* para cada palabra de la oración. La misma arquitectura se utiliza para todos los modelos de detección de entidades sin ajuste fino de parámetros. Para la detección de cada relación semántica, se propone una red neuronal que consiste en la integración de las capas explicadas más 2 capas *BiLSTM* apiladas. Se utiliza la misma arquitectura para todas los tipos de relaciones sin ajustar los parámetros.

Por último, en [26], para identificar entidades médicas y con 3 diferentes enfoques para detectar relaciones, se usaron los modelos: *Join AB-LSTM*, *BiLSTM* con *CRF* y analizador de relaciones de pareja, de los cuales *BiLSTM* con *CRF* y *Join AB-LSTM* han tenido resultados similares, donde *Join AB-LSTM* obtuvo mejores resultados en el escenario principal y un resultado bajo en la detección de relaciones. Este hace uso de datos externos con la intención de completar la información que ya se ha proporcionado. También se usó incrustaciones de palabras de los registros médicos electrónicos (50 millones de palabras). La única diferencia relevante en el uso de ambos modelos para la identificación de entidades y la extracción de relaciones es que para la detección de relaciones se pasa como parámetro las entidades detectadas

previamente.

En 2020, [45] obtuvo el mejor resultado para ambas tareas. Propuso un sistema que consiste en utilizar una red neuronal profunda que recibe los tokens de entrada y emite conjuntamente predicciones para varias variables de salida diferentes. Los tokens se introducen en un modelo *BERT* para obtener su contexto (incrustaciones). Estas incrustaciones se pasan a una capa de clasificación que emite *logits* con la predicción de que cada token es o no una entidad de cierto tipo. Las incorporaciones se pasan a un pequeño modelo *DistilBERT* inicializado aleatoriamente con solo dos capas de dos cabezas de atención cada una. El objetivo de este modelo es capturar más interacciones entre los pares de fichas a través de la auto-atención. Las representaciones de pares de tokens resultantes se pasan a varias capas de clasificación para hacer predicciones sobre la relación entre los tokens de cada par. Los pares se clasifican en clasificadores binarios que deciden respectivamente si los tokens que forman el par están conectados por una relación.

El sistema de [37] analiza la entrada utilizando *FreeLing* y cada uno de sus tokens se codifica utilizando *BERT, Word2Vec* o un modelo de incrustación de palabras previamente entrenado con *FastText*. El modelo aplica filtros de convolución a los tokens codificados de los documentos de entrada, combina las salidas del filtro a nivel palabra de cada token de entrada y el token de origen especificado con incrustaciones a nivel oración, y genera los límites de cada frase clave que contiene el token de origen, así como las probabilidades de que todos los demás tokens sean el objetivo de una relación que tenga el token de origen especificando frases clave como fuente. Para generar todas las relaciones posibles, el modelo debe ejecutarse para cada token de entrada y tener todas las probabilidades brutas combinadas en cada uno de ellos.

En [34], el sistema usa *BiLSTM + CRF*, en el que se combinaron diferentes tipos de incrustación de palabras como: Incrustaciones médicas de *FastText*, Incrustaciones contextuales de palabras e incrustaciones de palabras previamente entrenadas basadas en redes *transformers*. Los resultados indican que la combinación de todos los tipos de incrustaciones de palabras mejora los resultados de la identificación de entidades. Este sistema obtuvo la mejor evaluación a la hora de identificar entidades, sin embargo, no fue lo mismo para la extracción de relaciones.

En [52], el sistema consiste en un sistema de tuberías que hace uso de una red neu-

Tabla 3.3: Tabla comparativa entre sistemas para ambas subtareas entre participantes del *eHealth-KD challenge*.

Autor(es)	Corpus usado	Pre-procesamiento	Modelos usados	F_1 score
[36]	Corpus proporcionado *eHealth-KD challenge*.	*BIOUE-tag* Características: Codificación *One-Hot* de los campos de categoría y tipo y *One-Hot* de la distancia entre el token y el token de destino, vector normalizado que codifica las dependencias encontradas, incrustación de palabras contextualizada del token (*BERT-base*).	Compuesto por un conjunto de capas compartidas y dos capas de salida independientes capas de salida comparten la misma estructura, una capa completamente conectada y *CRF*.	0.6394
[13]	Corpus proporcionado *eHealth-KD challenge*, Billion Word Corpus.	Las palabras se representan mediante incrustaciones de *FastText* entrenadas en español.	Arquitectura *BiLSTM* apilada de 2 capas.	0.6218
[26]	Corpus proporcionado *eHealth-KD challenge*.	Incrustaciones de palabras.	*BiLSTM* y *CRF*.	0.4869
[45]	Corpus proporcionado *eHealth-KD challenge*.	Los tokens se introducen en un modelo *BERT* para obtener incrustaciones.	Una red neuronal profunda que recibe los tokens de entrada y emite conjuntamente predicciones para varias variables de salida diferentes.	0.665564
[37]	*SciELO database* y un subconjunto de corpus de salud de wikipedia.	Analiza la entrada utilizando *FreeLing* y cada uno de sus tokens se codifica utilizando un *BERT*, *Word2Vec* o un modelo de incrustación de palabras previamente entrenado con *FastText*.	La red está compuesta de un conjunto de capas intermedias compartidas y dos capas de salida independientes.	0.626679
[34]	Corpus de *MeSpEN*, información extra de diferentes clínicas como: *Mayo Clinic, World Health Organization and WebMD*, 2.5 TB de datos *CommonCrawl* filtrados.	Incrustaciones de palabras biomédicas, BETO (*BERT* español), *XLM-RoBERTa*.	*BiLSTM* con *CRF*.	0.42069
[52]	Corpus proporcionado *eHealth-KD challenge*.	Convierten la entrada del formato de entrenamiento *eHealth-KD challenge* 2021, al formato utilizado en el *eHealth-KD challenge* 2019, posteriormente al formato utilizado por Joint *AB-LSTM*.	*AB-LSTM*, *BiLSTM*.	0.395153

ronal conjunta *AB-LSTM* junto con una fase previa al proceso y una fase posterior al proceso. La red empleó incrustaciones de palabras como característica principal, incrustaciones capacitadas del dominio clínico. Esta red combina dos redes neuronales ampliamente utilizadas: una *BiLSTM* con agrupación máxima y una *BiLSTM* de atención. *Joint AB-LSTM* se alimenta de las sentencias pre-procesadas, sus entidades y las relaciones entre ellos y las incrustaciones de distancia creadas previamente.

En la Tabla 3.3, se muestra una comparativa entre sistemas para el reconocimiento de entidades entre participantes del *eHealth-KD challenge*.

Capítulo 4

Metodología de solución

Usando los conceptos teóricos y el estudio del estado del arte, se proponen soluciones para las tareas de identificación de entidades y extracción de relaciones.

Este capítulo se divide en cuatro secciones:

1. Pre-procesamiento de la información: Esta primera sección va orientada al preprocesamiento del corpus para usarse en los sistemas.

2. Arquitectura y extracción de información: Esta segunda sección explica las arquitecturas utilizadas con el fin de resolver cada subtarea. Para la identificación de entidades se consideró dos tipos de arquitecturas principales, *BiLSTM*, *BiLSTM-CNN* Y *BiLSTM-CRF*. Por último, para la extracción de relaciones se consideró una arquitectura *BiLSTM-CRF*, y una arquitectura *BERT* (red *transformer*).

3. Programación y entrenamiento de la red neuronal de aprendizaje profundo: Esta sección aborda el proceso que se utilizó para el entrenamiento y las herramientas que se utilizarón para la programación de cada red.

4. Evaluación de los resultados: En esta sección se muestra la metodologia para la evaluación de los resultados.

4.1. Pre-procesamiento de la información

En esta sección se muestra la descripción de cada corpus, la estructura para el pre-procesamiento, la descripción de la matriz de incrustación de palabras, los estilos de anotación para la identificación de entidades, la descripción de los etiquetadores a usar y la propuesta para el pre-procesamiento de cada sistema.

4.1.1. Descripción del corpus

Para la parte de pre-procesamiento se usarán principalmente los corpus proporcionados por *eHealth Challenge 2021*, pero también considerando los datos del 2020. Es importante resaltar que las oraciones en estos corpus en su mayoría son en español, pero también contienen oraciones completas en inglés.

Para la incrustación de palabras se considera extraer información relacionada (de Wikipedia) a los corpus de entrenamiento y desarrollo en español.

Los corpus proporcionados pueden ser organizados según la subtarea a la que se orienta: Identificación de entidades y corpus para extracción de relaciones.

Existen 3 corpus diferentes:

- *Development* (Para el desarrollo del sistema): Este corpus esta orientado para dar una idea al desarrollador de que contendrá el corpus de prueba (*Test*) y de entrenamiento (*Training*).

- *Training* (Para entrenamiento del sistema): Este corpus va orientado al entrenamiento de las redes neuronales, cada corpus contiene 100 oraciones en español.

- *Testing* (Para prueba del sistema): Este corpus va orientado a la evaluación de los modelos entrenados, cada corpus contiene 100 oraciones (50 en español y 50 en inglés).

De los corpus mencionados, se utilizarán los corpus de entrenamiento y prueba, se omite el corpus de desarrollo debido a que su función principal es dar una idea de como se conforma cada corpus y puede llegar a contener algunos errores.

Tabla 4.1: Ejemplo formato corpus proporcionado en [50].

El sistema vascular es la red de vasos sanguíneos del cuerpo.
Una conmoción cerebral es un tipo de lesión cerebral.
El moho puede causarle problemas de salud.
Estos síntomas pueden conducir a pérdida de peso y de líquidos corporales.

Tabla 4.2: Ejemplo formato *BRAT standoff*.

T1 Concept 3 10;11 19 sistema vascular
T2 Predicate 26 29 red
T3 Concept 33 38;39 49 vasos sanguíneos
T4 Concept 54 60 cuerpo
R0 arg Arg1:T2 Arg2:T3
R1 is-a Arg1:T1 Arg2:T2
R2 part-of Arg1:T1 Arg2:T4

Cada corpus está compuesto por dos archivos: Un archivo ".txt" que contiene las oraciones (Tabla 4.1), y otro ".ann" que contiene las entidades, el tipo de entidad, la posición de cada entidad y las relaciones (Tabla 4.2). Como se muestra en la Tabla 4.2 las entidades y relaciones vienen correctamente clasificadas (en formato *BRAT standoff*).

Para extraer la información del corpus proporcionado, se utilizó el archivo "ann-tools.py", igualmente proporcionado por *eHealth-KD challenge 2021* para la lectura de los archivos ".txt" y ".ann". Sin embargo, se escribió un programa convirtiendo palabras a minúsculas, separa los símbolos de las palabras (para mejorar tokenización) y que tokeniza el texto, incluyendo las palabras excluidas por el archivo ".ann".

4.1.2. Estructura del pre-procesamiento

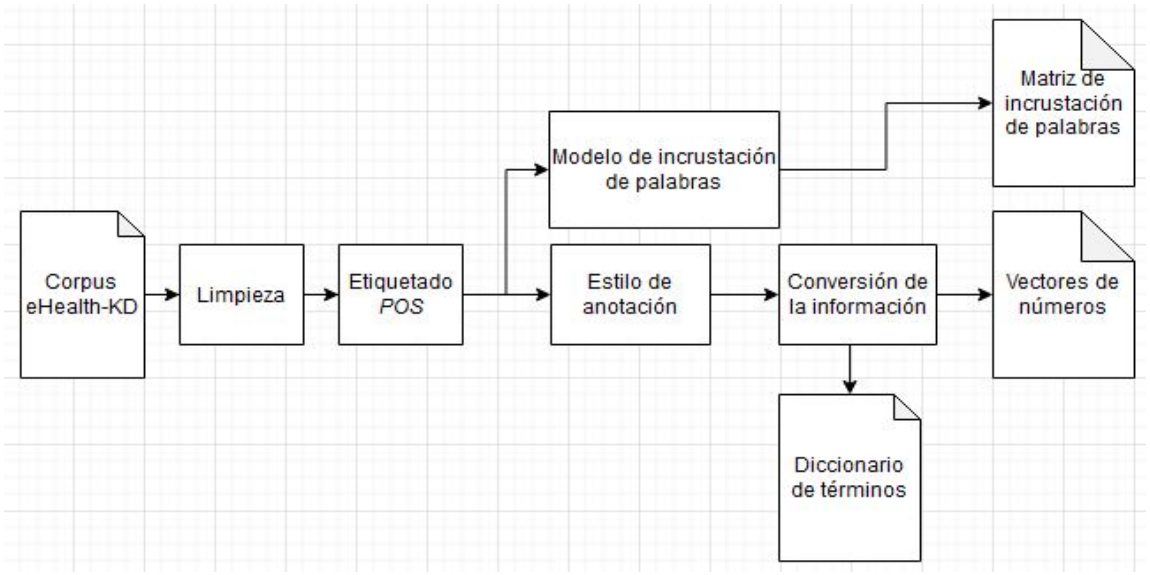

Figura 4.1: Pasos realizados para el pre-procesamiento de la identificación de entidades.

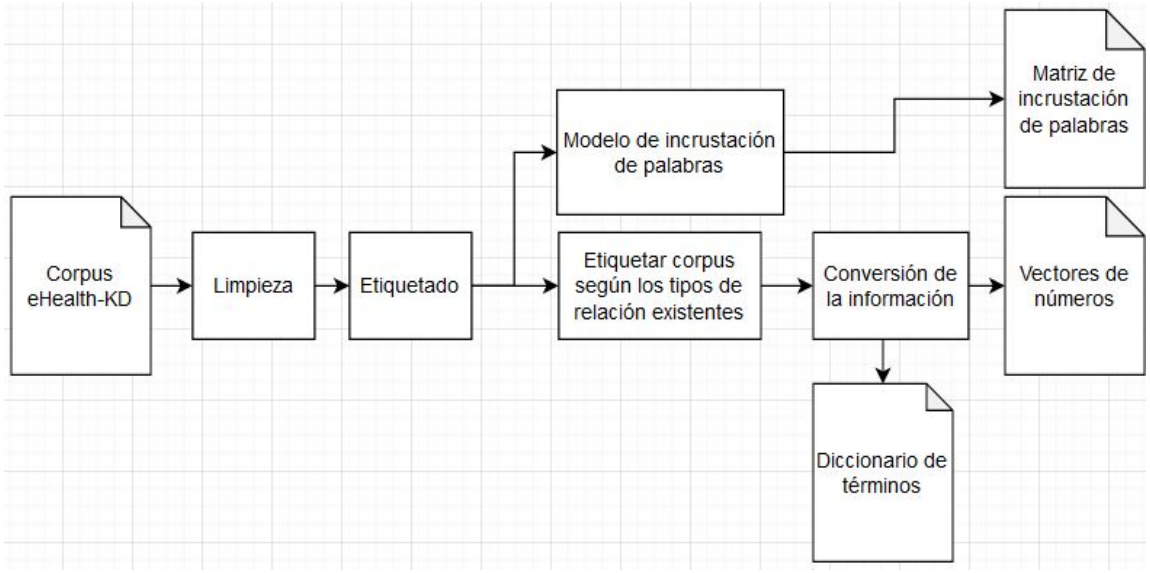

Figura 4.2: Pasos realizados para el pre-procesamiento de la extracción de relaciones.

Para el pre-procesamiento de la información en la identificación de entidades, en la Figura 4.1 se han considerado varios pasos:

1. Limpieza del texto: En este paso se dividen los corpus proporcionados en varias oraciones, se introducen espacios entre símbolos y palabras, y se eliminan las palabras que generan problemas a la hora de ordenar entidades.

2. Etiquetado: Las oraciones del paso anterior se tokenizan y se les agrega su etiquetado *POS* a cada palabra, para este paso se usan los etiquetadores *Spacy* y *Freeling*.

3. Estilo de anotación: En este paso se crea un vector para cada diferente estilo de anotación, también vectores de la información adicional que se pueda anexar.

4. Conversión de la información: Los vectores de palabras y caracteres se convierten en vectores de números, se genera un diccionario de términos, donde cada término tiene su equivalente en número (este diccionario sirve para posteriormente descifrar la salida del sistema). La salida de esta etapa ayuda a traducir los resultados del sistema de identificación de entidades.

5. Modelo de incrustación de palabras: El modelo de incrustación de palabras recibe el vocabulario junto con las etiquetas y expresiones, para convertir la información en una matriz de vectores de números, esta salida se usa para crear el sistema identificación de entidades.

Para el pre-procesamiento de la información en la extracción de relaciones, en la Figura 4.2 se han considerado los siguientes pasos:

1. Limpieza del texto: En este paso se dividen los corpus proporcionados en varias oraciones, se introducen espacios entre símbolos y palabras, y se eliminan las palabras que generan problemas a la hora de ordenar entidades.

2. Etiquetado: Las oraciones del paso anterior se tokenizan y se les agrega su etiquetado *POS* y etiquetado de dependencias a cada palabra, para este paso se usa *Spacy*.

3. Etiquetado: Se crea un diccionario con las relaciones dentro de cada oración y se etiquetan según el tipo de relación que sean.

4. Conversión de la información: Los vectores de palabras y caracteres se convierten en vectores de números, se genera un diccionario de términos, donde cada término tiene su equivalente en número (este diccionario sirve para posteriormente descifrar la salida del sistema). La salida de esta etapa ayuda a traducir los resultados del sistema de extracción de relaciones semánticas.

5. Modelo de incrustación de palabras: El modelo de incrustación de palabras recibe el vocabulario junto con las etiquetas y expresiones, para convertir la información en una matriz de vectores de números, esta salida se usa para crear el sistema de extracción de relaciones semánticas.

4.1.3. Etiquetado

Durante el etiquetado para ambas subtareas se hace uso como primera opción al etiquetador *Spacy*, que implementa el etiquetado *penn treebank* para textos en español, como para textos en inglés y el etiquetador *Freeling 4.2*, que implementa el etiquetado *penn treebank* para textos en inglés y *eagles* para textos en español.

El etiquetado gramatical (*part-of-speech tagging* en inglés) complementa la información del texto, para ello es importante contemplar que el etiquetado gramatical usualmente es diferente para español e inglés, y depende de la herramienta que se use para etiquetar. Es por ello que se usarán dos librerías de *Spacy* en su versión más grande, una para inglés y otra para español, pero que comparten el mismo formato de etiquetado.

- es_core_news_lg (español de 542 mb)
- en_core_web_lg (inglés de 714 mb)

En la Tabla 4.4 podemos observar como se implementa el etiquetado *POS* para cada palabra.

Para *Freeling* el etiquetado *part-of-speech tagging* es diferente para inglés y español. Con el objetivo de obtener un etiquetado igual en *Freeling 4.2*, para textos

Tabla 4.3: Conversión de etiquetado de *Freeling 4.2*

Etiqueta	***Freeling 4.2***	***eagles* (reducida)**
Número cardinal	CD	Z
Determinante	DT	D
Existencial *there*	EX	VM
Palabra extranjera	FW	X
Preposición o conjunción subordinada	IN	CS
Adjetivo	JJ	A
Adjetivo comparativo	JJR	A
Adjetivo superlativo	JJS	A
Modal	MD	VA
Sustantivo, singular o masa	NN	NC0N
Sustantivo plural	NNS	NC0P
Sustantivo propio, singular	NNP	NP0S
Sustantivo propio, plural	NNPS	NP0P
Predeterminante	PDT	DI
Pronombre personal	PRP	PP
Pronombre posesivo	PRP$	PX
Adverbio	RB	RG
Adverbio comparativo	RBR	RG
Adverbio superlativo	RBS	RG
Partícula	RP	RG
to	TO	SPS
Interjección	UH	I
Verbo, forma base	VB	VMN
Verbo, tiempo pasado	VBD	VM0S
Verbo, gerundio o participio presente	VBG	VMP
Verbo, participio pasado	VBN	VMPS
Verbo, presente, no tercera persona	VBP	VAMP
Verbo, presente, tercera persona	VBZ	VAMP3
Wh-Determinante	WDT	DT
Wh-Sustantivo	WP	PT

Tabla 4.4: Ejemplo de etiquetado *POS*

Token	**Etiquetado POS**
Algunos	DET
asilos	NOUN
de	ADP
ancianos	NOUN
cuentan	VERB
con	ADP
unidades	NOUN
de	ADP
cuidados	NOUN

Tabla 4.5: Ejemplo de etiquetado de dependencias en *Spacy*

Palabra	**Dependencia**	**Palabra padre**
Autonomous	amod	cars
cars	nsubj	shift
shift	root	shift
insurance	compound	liability

en español e inglés, se realizó una conversión del etiquetado *penn treebank* a una versión reducida de *eagles*, como se observa en la Tabla 4.3. El etiquetado reducido de *eagles* consiste en recortar o modificar algunas etiquetas para hacer coincidir las etiquetas con las etiquetas *Freeling 4.2* que fueron convertidas a *eagles*. De esta manera se pierde mucha información, sin embargo, se logra que a partir del corpus de entrenamiento se puedan obtener mejores resultados a la hora de evaluar textos en inglés, debido a que el corpus de entrenamiento no tiene ejemplos en inglés.

El etiquetado de dependencias a utilizar es el que ofrece *Spacy* (Dependencias universales), cada dependencia se caracteriza en términos de una función sintáctica. En la Tabla 4.5 se muestra un ejemplo de como se realiza el etiquetado de dependencias en *Spacy*.

4.1.4. Estilo de la anotación (identificación de entidades)

Se han considerado los siguientes estilos de anotación, los cuales se mencionan dentro del capítulo del Marco Teórico: *IO (Inside, Outside)*, *BIO (Beginning, Inside,*

Outside) y *BILOU (Beginning, Inside, Last, Outside, Unique)*.

Para implementar el estilo de anotación se debe considerar la clasificación de las entidades, en este caso: conceptos, acciones, predicados y referencias, además de agregar la etiqueta "Ninguna" para las palabras en el texto que no pertenecerán a ningún grupo de entidades anteriormente mencionado. Por ejemplo, en el estilo de anotación (*BILOU*) se agregan los siguientes prefijos (exepto "O_") para cada tipo de entidad.

- B_ : Para palabras inicio de una entidad
- I_ : Para palabras dentro de una entidad
- L_: Para palabra fin de una entidad
- O_ : Para palabras que no son entidades
- U_ : Entidades de una sola palabra

Para el estilo de anotación *BIO*, solo se agregan los siguientes prefijos:

- B_ : Para palabras inicio de una entidad
- I_ : Para palabras dentro y que finalizan una entidad
- O_ : Para palabras que no son entidades

De tal forma que se obtienen 17 etiquetas en total en el caso del estilo de anotación *BILOU*, la Tabla 4.6 nos muestra un ejemplo del estilo de anotación *BILOU*.

En la Tabla 4.7 se muestra un ejemplo del estilo de anotación, donde las etiquetas que comienzan con "P_" hacen referencia a las entidades que se conforman por más de una palabra, este etiquetado está basado principalmente en el estilo de anotación *IO* (*Inside, Outside*) y *BIO* (*Beginning, Inside, Outside*)[31].

Finalmente, la Tabla 4.8 nos muestra un ejemplo del estilo de anotación *BIO*.

Tabla 4.6: Ejemplo de estilo de anotación *BILOU*.

Token	**Etiqueta**
Algunos	U_Predicado
asilos	B_Concepto
de	I_Concepto
ancianos	L_Concepto
cuentan	O_
con	O_
unidades	U_Concepto
de	O_
cuidados	U_Acción

Tabla 4.7: Ejemplo de estilo de anotación usando "P_"

Token	**Etiqueta**
Algunos	Predicado
asilos	P_Concepto
de	P_Concepto
ancianos	P_Concepto
cuentan	O_
con	O_
unidades	Concepto
de	O_
cuidados	Acción

Tabla 4.8: Ejemplo de estilo de anotación *BIO*.

Token	**Etiqueta**
Algunos	B_Predicado
asilos	B_Concepto
de	I_Concepto
ancianos	I_Concepto
cuentan	O_
con	O_
unidades	B_Concepto
de	O_
cuidados	B_Acción

4.1.5. Modelo de incrustación de palabras

Con la finalidad de generar la matriz de inscrustación de palabras, que se utilizará para implementar la arquitectura *BiLSTM-CRF* propuesta en esta tesis, es necesario crear un modelo de incrustación de palabras. Para obtener el modelo que más se ajuste a las necesidades de las tareas planteadas en este tesis, se recopilarán modelos ya existentes y se crearán modelos a partir de un corpus recopilado de información de *Wikipedia*. Los modelos serán creados a partir de las siguientes herramientas: *Fasttext*, *Word2vec* y *GloVe*.

Para crear el corpus se plantea utilizar una librería de Wikipedia para Python, la cual permite realizar búsquedas y guardar la información en textos. Para la creación de cada modelo de incrustación de palabras se plantean los siguientes pasos:

1. A partir del corpus de entrenamiento proporcionado por *eHealth-KD challenge 2021*, se crea un vocabulario de palabras.

2. Con la librería de *Wikipedia* de *Python*, se buscará y guardará cualquier artículo relaciondado con las palabras incluidas en el vocabulario.

3. Con el corpus creado se entrenarán diferentes modelos de inscrutación de palabras, haciendo uso de las diferentes herramientas propuestas y con diferentes parámetros de configuración.

Matriz de incrustación de palabras La matriz de incrustación de palabras, se creará a partir de uno de los diferentes modelos de incrustación recopilados (esto incluye a los modelos creados) y se probará en conjunto a la arquitectura propuesta para ver que matriz obtiene mejores resultados.

Para crear cada matriz, se utilizan todas las palabras diferentes (vocabulario) del corpus de entrenamiento y de prueba, de las cuales con la ayuda del modelo de incrustación, se obtiene el vector de números que representa cada palabra.

Tabla 4.9: Frecuencia de aparición de entidades en base al etiquetado *POS*.

Etiqueta POS	**Frecuencia**
NOUN	342
VERB	116
ADJ	91
PROPN	24
PRON	15

Tabla 4.10: Frecuencia de aparición de entidades (conformadas por más de una palabra) en base al etiquetado *POS*.

Patrón de etiquetado POS	**Frecuencia**
NOUN ADJ	33
PROPN PROPN	9
NOUN ADP NOUN	8
NOUN ADP PROPN	6
PROPN PROPN PROPN	3

4.1.6. Propuesta para el pre-procesamiento (identificación de entidades)

Con la finalidad de mejorar el desempeño de los sistemas de identificación de entidades, se propone anexar información adicional en la etapa de "estilo de anotación y etiquetado" usando el etiquetado *POS*.

A partir de los corpus de entrenamiento y desarrollo proporcionados, se extraen los patrones y su frecuencia dentro del texto. La Tabla 4.9 nos muestra un ejemplo de la frecuencia con la que cada tipo de palabra aparece como una entidad, en este caso los sustantivos (etiqueta *NOUN*) tienen mayor frecuencia de aparición como entidades dentro del texto.

De igual forma que las entidades de una sola palabra se capturaron los patrones para cada entidad que se conforma por más de una palabra, como por ejemplo "asilos de ancianos" donde el etiquetado correspondiente seria *"NOUN ADP NOUN"*. En la Tabla 4.10 se muestra un ejemplo de como se captura la frecuencia de estos patrones.

La información de la frecuencia del etiquetado *POS* se clasificó de forma manual en 6 rangos:

Tabla 4.11: Ejemplo de secuencia de etiquetas creadas con estilo de anotación *BIO* y *BILOU*

Palabra	**POS**	***BIO***	***BILOU***
algunos	DET	B_BAJA_FRECUENCIA	U_BAJA_FRECUENCIA
asilos	NOUN	B_BAJA_FRECUENCIA	B_BAJA_FRECUENCIA
de	ADP	I_BAJA_FRECUENCIA	I_BAJA_FRECUENCIA
ancianos	NOUN	I_BAJA_FRECUENCIA	L_BAJA_FRECUENCIA
cuentan	VERB	B_MEDIA_FRECUENCIA	U_MEDIA_FRECUENCIA
con	ADP	B_MUYBAJA_FRECUENCIA	U_MUYBAJA_FRECUENCIA

- Frecuencia muy alta (0.75 - 1).
- Frecuencia alta(0.5 - 0.75).
- Frecuencia media(0.35 - 0.5).
- Frecuencia baja.(0.1 - 0.35).
- Frecuencia muy baja (0 - 0.1).
- Ninguna frecuencia. (0)

La clasificación se obtiene con base en la probabilidad de aparición de los patrones encontrados, se calcula dentro de un rango de 0 a 1, tomando la frecuencia de aparición de una etiqueta como el total y dividiendo entre la frecuencia de aparición de esa etiqueta como entidad.

Por ejemplo, si un sustantivo (*NOUN*) aparece en total 678 veces y aparece 534 veces como entidad, las palabras que sean etiquetadas como sustantivo tienen una probabilidad de aparecer de (*534/678*) *0.78* y una probabilidad de 0 si no aparecen ninguna vez como entidad.

Para entidades que se componen por más de una palabra se buscaron secuencias de palabras que cumplan con los patrones establecidos anteriormente y posteriormente se clasifican como se ha mostrado previamente, utilizando el estilo de anotación *BIO* y *BILOU*. En la Tabla 4.11 podemos ver como quedaría la secuencia de palabras, con el estilo de anotación añadido.

4.1.7. Propuesta para el pre-procesamiento (extracción de relaciones semánticas)

Con la finalidad de mejorar el desempeño de los sistemas de extracción de relaciones semánticas, además del etiquetado *POS* y de dependencias, se propone anexar información de patrones e información de dependencias en forma de etiquetas de lenguaje de marcado (marcado descriptivo).

La propuesta de patrones intenta imitar lo que se realizó con la identificación de entidades, para esta propuesta se han considerado los patrones de aparición contemplando en las etiquetas que señalan el tipo de entidad y etiquetado *POS* (Ejemplo: Concept-Noun_Action-Noun, Action-Noun_Action-Noun, Action-Verb_Action-Noun, etc.), de manera muy similar a como se lleva a cabo en la identificación de entidades, primero las entidades se organizan por frecuencia de aparición en una relación (Muy alta frecuencia, alta frecuencia, baja frecuencia y muy baja frecuencia), la frecuencia de aparición depende del tipo de elemento que se considere "a" o "b", una entidad puede tener baja frecuencia de aparición como un elemento "a" (padre) en una relación, sin embargo, puede tener una alta frecuencia de aparición como elemento "b" (hijo) en una relación, de tal manera que pare este ejemplo el etiquetado sería este "B-FRECUENCIA_A_A-FRECUENCIA_B".

En la segunda propuesta, debido a que la función "token.dep_" de *Spacy* por si sola no indica la palabra padre en una relación de dependencia, se pretende agregar esta información utilizando lenguaje de marcas, para agregar la información de dependencias en forma de marcado descriptivo es importante contemplar las funciones "token.dep_" y "token.head" de *Spacy* para obtener los elementos padre e hijo en cada dependencia. El etiquetado agregado consiste es marcar el inicio y fin de cada oración con las etiquetas "<oración>" y "</oración>", y marcar el inicio y fin de cada dependencia dependiendo de la dependencia que se use, por ejemplo, en la oración "si su hijo tiene" existe una dependencia de tipo "NSUBJ" (sujeto nominal) entre las palabras "hijo" y "tiene", por lo que el etiquetado final sería el siguiente "<oración>si su <NSUBJ>hijo tiene</NSUBJ></oración>".

4.2. Arquitectura y extracción de información

En esta sección se muestran las arquitecturas utilizadas con la finalidad de resolver cada subtarea.

4.2.1. Identificación de entidades

Para la identificación de entidades se plantea hacer uso de tres sistemas, uno basado en la red *BiLSTM*, *BiLSTM* con *CNN*y otro con *BiLSTM* con *CRF*.

En la Figura 4.3 podemos observar como los archivos de la salida del pre-procesamiento son la entrada para el sistema de identificación de entidades, cada sistema arrojará un vector de números como resultado, que se debe convertir usando los diccionarios de términos para ayudar a obtener el archivo final en formato *BRAT*.

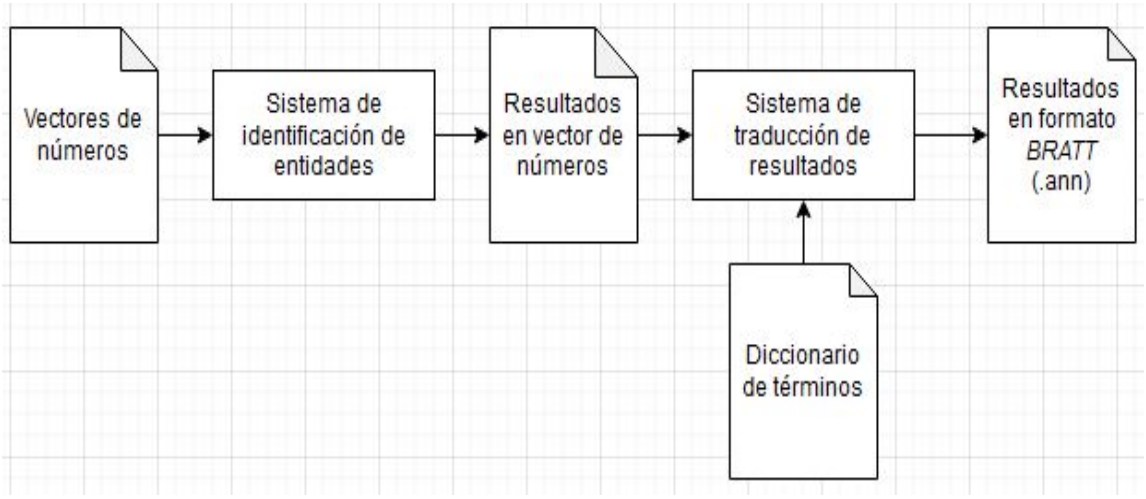

Figura 4.3: Pasos para la identificación de entidades.

Sistema *BiLSTM* Para este sistema a cada etiqueta se le asigna de forma manual un valor decimal del 0 al 1 para introducirlo dentro de la red *BiLSTM*, la Tabla 4.12 nos muestra un ejemplo más claro de esta asignación, de esta misma manera se asignarán valores para el etiquetado *BIO* y *BILOU* [31].

Para clasificar la salida, se hace uso de la fórmula de distribución normal con la finalidad de obtener una clasificación más exacta en los valores de salida.

Tabla 4.12: Ejemplo de asignación de valores para sistema *BILSTM*

Token	**Etiqueta**	**Valor**
Algunos	Predicado	0.75
asilos	P_Concepto	0.18
de	P_Concepto	0.18
ancianos	P_Concepto	0.18
cuentan	O_	0.00
con	O_	0.00
unidades	Concepto	0.25
de	O_	0.00
cuidados	Acción	0.50

Tabla 4.13: Ejemplo de estilo de anotación *BIO* y matriz binaria para sistema *BiLSTM-CRF*

Token	**Clasificación**	**Valor**
Algunos	B_Predicate	[0,0,0,0,0,0,0,0,1]
asilos	B_Concept	[0,1,0,0,0,0,0,0,0]
de	I_Concept	[0,0,1,0,0,0,0,0,0]
ancianos	I_Concept	[0,0,1,0,0,0,0,0,0]
cuentan	O_	[1,0,0,0,0,0,0,0,0]
con	O_	[1,0,0,0,0,0,0,0,0]
unidades	B_Concept	[0,0,0,1,0,0,0,0,0]
de	O_	[1,0,0,0,0,0,0,0,0]
cuidados	B_Action	[0,0,0,0,1,0,0,0,0]

Sistema *BiLSTM-CRF* Para la entrada de este sistema, en lugar de darle un valor decimal a cada etiqueta, se convierten los valores de las etiquetas a una matriz binaria para incorporarlas a la red. La salida del sistema es una matriz binaria con una dimensión del $n * c$, donde n es el número de palabras y c es el número de clasificaciones resultantes del estilo de anotación. En la Tabla 4.13 y la Tabla 4.14 se muestra un ejemplo más detallado para los estilos de anotación *BIO* y *BILOU* correspondientemente.

Sistema *BiLSTM-CNN* Para la entrada de este sistema se introducen las palabras a nivel carácter, para ello se define una lista de caracteres (Por ejemplo: “abc-

Tabla 4.14: Ejemplo de estilo de anotación *BILOU* y matriz binaria para sistema *BiLSTM-CRF*

Token	**Clasificación**	**Valor**
Algunos	U_Predicate	[0,0,0,0,0,0,0,0,1...]
asilos	B_Concept	[0,1,0,0,0,0,0,0,0...]
de	I_Concept	[0,0,1,0,0,0,0,0,0...]
ancianos	L_Concept	[0,0,0,1,0,0,0,0,0...]
cuentan	O_	[1,0,0,0,0,0,0,0,0...]
con	O_	[1,0,0,0,0,0,0,0,0...]
unidades	U_Concept	[0,0,0,0,1,0,0,0,0...]
de	O_	[1,0,0,0,0,0,0,0,0...]
cuidados	U_Action	[0,0,0,0,0,1,0,0,0...]

Tabla 4.15: Entrada para incrustaciones a nivel carácter

Token	**Valor**
Algunos	[32,26,37,53,45,52,51,0,0...]
asilos	[32,51,87,26,52,51,0,0,0...]
de	[49,56,0,0,0,0,0,0,0...]
ancianos	[32,45,17,87,32,45,52,51,0...]
cuentan	[17,53,56,45,38,32,45,0,0...]
con	[17,52,45,0,0,0,0,0,0...]

defghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZ 0123456789 ._+¡¿!?:,") y se crea un diccionario donde se asigna un número a cada letra, debido a que la red necesita números enteros positivos como entrada (no es necesario un orden en específico). Para representar cada palabra se crea un vector con longitud l, donde l es la longitud de la palabra más larga del corpus y los espacios en blanco son llenados en 0, como se muestra en la Tabla 4.15.

4.2.2. Entrada para entrenamiento y predicción de los sistemas de identificación de entidades

Para el entrenamiento y predicción de la red tenemos 3 vectores diferentes, estos son considerados como eje "X": Vector de palabras, vector *POS*, vector frecuencias *POS*.

Tabla 4.16: Ejemplo entrada sistema (eje “X"), para la identificación de entidades.

Palabra	**POS**	**Frecuencias**
algunos	DET	B_BAJA_FRECUENCIA
asilos	NOUN	B_BAJA_FRECUENCIA
de	ADP	I_BAJA_FRECUENCIA
ancianos	NOUN	I_BAJA_FRECUENCIA
cuentan	VERB	B_MEDIA_FRECUENCIA
con	ADP	B_MUYBAJA_FRECUENCIA

Mientras que el vector con los estilos de anotación que contiene las respuestas se considera como el eje “Y”.

En la Tabla 4.16 podemos ver un ejemplo de como quedaría la secuencia, la unión de vectores como eje “X", para la identificación de entidades.

Mientras que en la Tabla 4.17 podemos ver como quedaría la secuencia la unión de vectores como eje “Y".

Estos ejes son obtenidos del corpus de entrenamiento y de evaluación. Para el entrenamiento, la entrada del sistema son los vectores “X”, “Y” (del corpus de entrenamiento), como se da el ejemplo en la Figura 4.4.

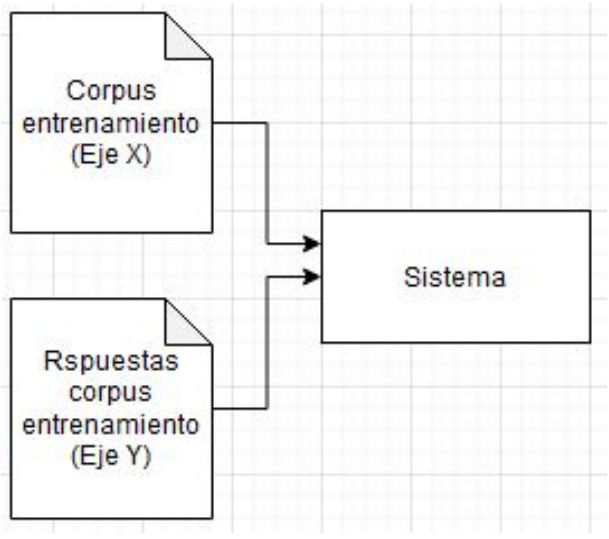

Figura 4.4: Ejemplo de entrada de ejes para entrenamiento.

Sin embargo, para la predicción de los resultados, la entrada del sistema es el

Tabla 4.17: Ejemplo entrada sistema (eje “Y"), para identificación de entidades.

Estilo de anotación
U_Predicado
B_Concepto
I_Concepto
L_Concepto
O_
O_

vector “X” (del corpus de evaluación), como se ejemplifica en la Figura 4.5, lo cual nos da una salida con el formato del vector “Y”.

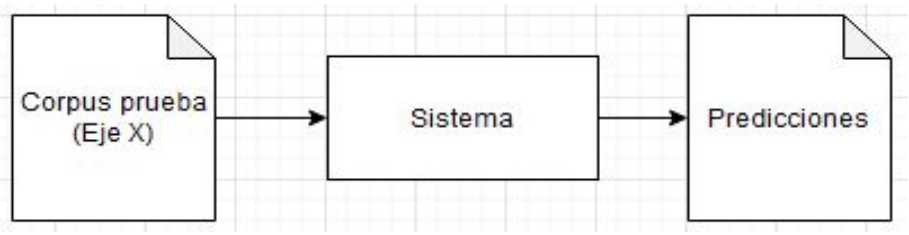

Figura 4.5: Ejemplo de entrada de ejes para predicción.

4.2.3. Extracción de relaciones

Para la extracción de relaciones semánticas se plantea hacer uso del mismo sistema (*BiLSTM* con *CRF*) que se usó para la identificación de entidades.

En la Figura 4.6 podemos observar como el archivo con los vectores de números (la salida de la etapa de pre-procesamiento) es la entrada para el sistema de extracción de relaciones semánticas, este archivo contiene información de las entidades detectadas e información del texto. En este caso se entrenará una red para cada tipo de relación o una red para todos los tipos de relaciones existentes (dependiendo de la propuesta), cada sistema arrojará un vector de números como resultado, que se debe convertir usando los diccionarios de términos y las entidades identificadas para crear las relaciones, estas relaciones se deben reagrupar para finalmente obtener el archivo final en formato *BRAT*.

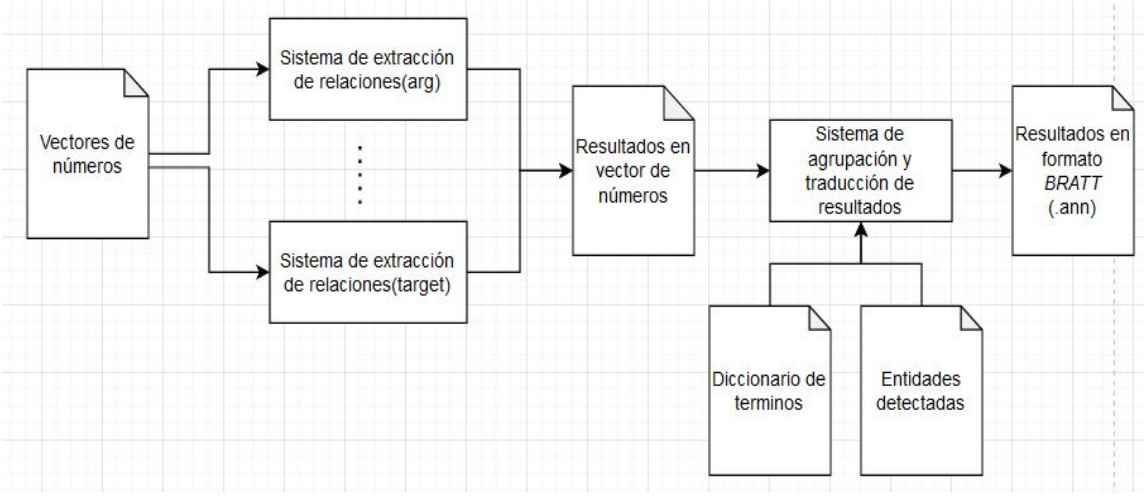

Figura 4.6: Pasos para la extracción de relaciones.

Sistema *BiLSTM-CRF* Al igual que en la entrada del sistema *BiLSTM* con *CRF* para identificación de entidades se convierten los valores de las etiquetas a una matriz binaria para incorporarlas a la red. La salida del sistema es una matriz binaria con una dimensión $n * c$, donde n es el número de palabras y c es el número de clasificaciones. Sin embargo, para esta red no se utilizará ningún estilo de anotación como el *BIO* o *BILOU*, las entradas para este sistema se explicarán a detalle en la sección 4.2.4.

4.2.4. Entrada para entrenamiento y predicción de los sistemas de extracción de relaciones semánticas

Para la entrada de este sistema, se plantean 4 opciones de entrada para las arquitecturas propuestas, y son las siguientes:

Opción 1 Considera una etiqueta por palabra, también plantea crear un modelo para los 13 tipos de relación, en la Tabla 4.18 se muestra un ejemplo de como se implementaría esta opción para la red propuesta. Para cada oración se tendría que obtener el elemento "a" y "b" de las relaciones existentes. Sin embargo, esta opción se tiene que descartar debido a la cantidad de etiquetas de clasificación que se generarían

Tabla 4.18: Ejemplo de etiquetado para extracción de relaciones (con múltiples relaciones) en red *BiLSTM-CRF* (Opción 1).

Token	**Clasificación**	**Valor**
Una	O_	[1,0,0,0,0,0,0,0,0,...]
quemadura	causes_is-a	[0,0,1,0,0,0,0,0,0,...]
es	O_	[1,0,0,0,0,0,0,0,0,...]
una	O_	[1,0,0,0,0,0,0,0,0,...]
lesión	in-place_is-a	[0,1,0,0,0,0,0,0,0,...]
en	O_	[1,0,0,0,0,0,0,0,0,...]
los	O_	[1,0,0,0,0,0,0,0,0,...]
tejidos	in-place	[0,0,0,1,0,0,0,0,0,...]

por combinar los 13 tipos de relaciones diferentes que existen (13!/(13-2)!2! = 78), es decir, 78 clasificaciones diferentes, esto sin considerar si un elemento es "a" o "b" en una relación.

Opción 2 Plantea una etiqueta por palabra, pero a diferencia de la primera opción se genera un modelo por cada tipo de relación, es decir, se estarían entrenando 13 modelos (uno por cada tipo de relación). En esta opción se plantean 3 diferentes tipos de entradas: Entrada simple (etiquetado *POS*, dependencias), entrada basada en patrones (patrones de etiquetas *POS*) y entrada de lenguaje de marcado (etiquetas dependencias). En la Tabla 4.19 se muestra como se implementaría la segunda opción. Como se observa, al igual que la primera opción se tiende a recuperar solamente un elemento de las relaciones encontradas, pero con la ventaja de que el número de clasificaciones es mucho menor (3 clasificaciones por relación: no es relación, elemento "a" y elemento "b").

Opción 3 También contempla una etiqueta por palabra, y al igual que la primera opción propone entrenar un modelo para los 13 tipos de relación, pero para evitar la gran cantidad de etiquetas que se crean, utiliza las marcas (E1) y (E2) para marcar una relación dentro de cada oración. En la Tabla 4.20 se muestra como se implementaría la opción 3 usando las marcas "[E1],[/E1]" y "[E2],[/E2]", para indicar donde inicia y termina cada elemento en una posible relación, para ello se hacen combinaciones (pares) de todas entidades en una oración, y se muestran repitiendo

Tabla 4.19: Ejemplo de etiquetado para extracción de relaciones (con una relación) en red *BiLSTM-CRF* (Opción 2).

Token	**Clasificación**	**Valor**
Una	O_	[1,0,0]
quemadura	a_is-a	[0,1,0]
es	O_	[1,0,0]
una	O_	[1,0,0]
lesión	b_is-a	[0,0,1]
en	O_	[1,0,0]
los	O_	[1,0,0]
tejidos	O_	[1,0,0]

Tabla 4.20: Ejemplo de etiquetado para extracción de relaciones (con una relación) en red *BiLSTM-CRF* (Opción 3).

Token	**Clasificación**	**Valor**
Una	O_	[1,0]
[E1]	O_	[1,0]
quemadura	is-a_a	[0,1]
[/E1]	O_	[1,0]
es	O_	[1,0]
una	O_	[1,0]
[E2]	O_	[1,0]
lesión	is-a_b	[0,1]
[/E2]	O_	[1,0]
en	O_	[1,0]
los	O_	[1,0]
tejidos	O_	[1,0]

Tabla 4.21: Ejemplo de etiquetado base final para extracción de relaciones para red *BiLSTM-CRF* (Opción 4).

Oración	**Clasificación**
Una [E1]quemadura[/E1] es una [E2]lesión[/E2] en los tejidos	is-a
Una quemadura es una [E1]lesión[/E1] en los [E2]tejidos[/E2]	in-place
Una [E1]quemadura[/E1] es una lesión en los [E2]tejidos[/E2]	None

Tabla 4.22: Ejemplo entrada con dependencias (*Spacy*) para red *BiLSTM-CRF* para extracción de relaciones

Token	**Tipo de entidad/Etiquetado(POS)**	**Etiquetado (DEP)**
Las	O_-DET	DET
manchas	Concept-NOUN	NSUBJ
blancas	Concept-ADJ	AMOD
son	O_-AUX	COP
más	O_-ADV	ADVMOD
comunes	Concept-NOUN	ROOT
donde	O_-PRON	OBL

la oración, pero marcando las diferentes combinaciones, este etiquetado es el usado en trabajos como el de [6]. Sin embargo, para redes recurrentes (*BiLSTM*) este tipo de etiquetado genera muchos registros que no contienen relaciones, y que no ayudan al aprendizaje de este tipo de redes neuronales, es por esa razón que esta opción se descarta.

Opción 4 Esta opción a diferencia de las demás, contempla una etiqueta por oración y plantea entrenar un modelo para los 13 tipos de relación, al igual que la opción 3 utiliza las marcas (E1) Y (E2) para marcar una relación dentro de cada oración, pero al contemplar una etiqueta por oración el número de elementos sin relación es menor que la opción 3. En la Tabla 4.21 se puede observar como se implementaría la opción 4.

Propuestas de entrada para la opción 2 Como se ha mencionado anteriormente la opción 2 tiene 3 propuestas diferentes de entrada y son las siguientes:

Entrada simple En la Tabla 4.22 se muestra la entrada simple para la red agregando el etiquetado de dependencias y combinando los tipos de entidad con el etiquetado *POS*(*Spacy* o *Freeling*).

Entrada basada en patrones En la Tabla 4.23 se cambia la segunda columna por un etiquetado de patrones previamente explicado en la sección 4.1.7.

Tabla 4.23: Ejemplo entrada con patrones y dependencias (*Spacy*) para red *BiLSTM-CRF* para extracción de relaciones

Token	**Patrones**	**Etiquetado (DEP)**
Una	O_	DET
quemadura	B_FRECUENCIA_A_B_FRECUENCIA_B	NSUBJ
es	O_	COP
una	O_	DET
lesión	MB_FRECUENCIA_A_B_FRECUENCIA_B	ROOT
en	O_	CASE
los	O_	DET

Tabla 4.24: Ejemplo de entrada usando etiquetas de marcado descriptivo para dependencias (*Spacy*) para red *BiLSTM-CRF* para extracción de relaciones

Token	**Tipo de entidad/Etiquetado(POS)**	**Etiquetado (DEP)**
<ORACION>	<ETIQUETA>	<ETIQUETA>
si	O_-SCONJ	MARK
su	O_-DET	DET
<NSUBJ>	<ETIQUETA>	<ETIQUETA>
hijo	O_-NOUN	NSUBJ
<OBJ>	O_-VERB	<ETIQUETA>
tiene	Concept-NOUN	ADVCL
</NSUBJ>	</ETIQUETA>	</ETIQUETA>

Entrada de lenguaje de marcado descriptivo Por último, en la Tabla 4.24 se usa como base la Tabla 4.22 y se le agregan etiquetas con un marcado descriptivo para delimitar las dependencias. Estas etiquetas de marcado descriptivo delimitan de forma más detallada el inicio y el fin de cada relación de dependencias, esto se explicó más a detalle en la sección 4.1.7.

4.3. Programación y entrenamiento de la red neuronal de aprendizaje profundo

En esta sección se muestra el método que se utilizará para entrenar los sistemas propuestos usando validación cruzada.

4.3.1. Método para validación cruzada durante el entrenamiento

Con la finalidad de comparar los resultados con los de *eHealth-KD challenge 2021*, se optará por no unir el corpus de entrenamiento con el corpus de pruebas y para el entrenamiento de los modelos se ocupará, solamente el corpus de entrenamientos proporcionado (corpus de entrenamiento proporcionados por *eHealth-KD challenge 2020 y 2021*). Los pasos a realizar durante el entrenamiento de cada modelo, y que se observan en la Figura 4.7 son los siguientes:

- Se divide el corpus de entrenamiento (solamente el corpus de entrenamiento) en 2 partes (20 % para evaluación del entrenamiento y 80 % para entrenamiento) de forma aleatoria.

- Se entrena el modelo propuesto con el 80 % del corpus entrenamiento.

- Se evalua la red neuronal con el 20 % del corpus de entrenamiento destinado a la evaluación del modelo entrenado, esta evaluación principalmente usa la métrica de exactitud (*accuracy*) que mide el porcentaje de casos que el modelo ha acertado, esta métrica no se usará para evaluar el sistema, se usará como herramienta para ajustar el desempeño del sistema.

- Se realizan cambios en los parámetros de entrenamiento con base en los resultados (lote, épocas, taza de aprendizaje, etc.), estos cambios se realizan en base a los resultados de la evaluación del paso anterior.

- Una vez obtenido el mejor resultado basándose en varios experimentos se procede a evaluar el modelo entrenado con el corpus de evaluación de *eHealth-KD challenge 2021*.

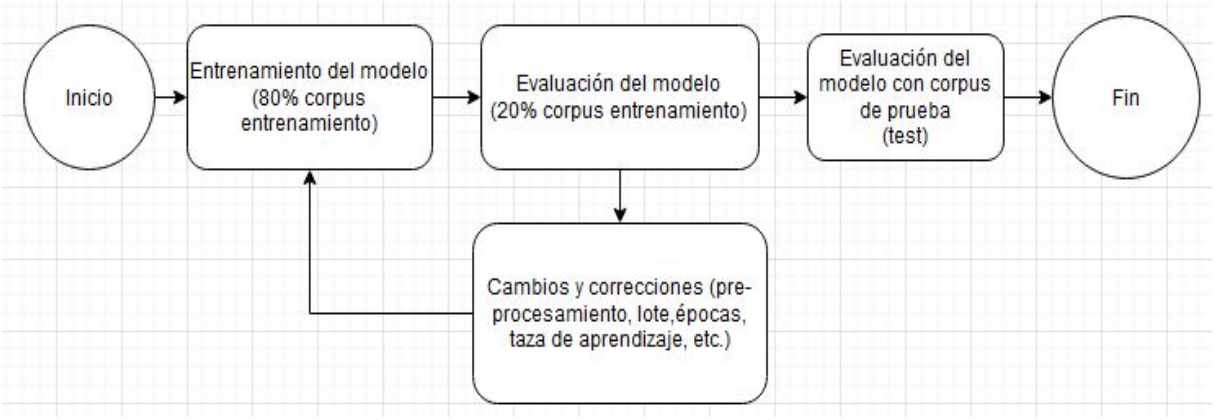

Figura 4.7: Evaluación del entrenamiento de cada modelo.

La configuración del modelo final, se ejecuta durante 10 instancias (corpus de prueba) para verificar los resultados con las métricas de evaluación.

4.4. Evaluación de los resultados

En esta sección se presentan las métricas usadas y como se implementan para la evaluación del escenario principal, finalmente, el método de evaluación usado.

4.4.1. Métricas de evaluación

A continuación se definirán las métricas que se usarán para evaluar los resultados en este trabajo de tesis.

Para calcular las métricas, primero se deben definir las siguientes puntuaciones: Coincidencias correctas (C), parciales(P), faltantes(M), incorrectas (I), y falsas (S).

Las coincidencias correctas (C) se informan cuando un texto en el archivo de resultados del sistema (*res*) coincide exactamente con un intervalo de texto correspondiente en el archivo de respuestas (*gold*) en los valores de inicio (*START*), fin (*END*), y también el tipo de entidad. Solo existe una coincidencia correcta por entrada en el archivo *gold*. Por lo tanto, las entradas duplicadas contarán como falsas. Las coincidencias incorrectas (I) cuando los valores *START* y *END* coinciden, pero no el tipo. Las coincidencias parciales (P) se reportan cuando dos intervalos (*START*,

END) tienen una intersección no vacía, como en el caso de "vías respiratorias" y "respiratorias". Una frase parcial solamente se comparará con una única frase correcta. Por ejemplo, "tipo de cáncer" podría ser una coincidencia parcial tanto para "tipo" como para "cáncer", pero nada más se cuenta una vez como una coincidencia parcial con la palabra "tipo". Las coincidencias faltantes (M) son las que aparecen en el archivo gold pero no en el archivo *res*. Las coincidencias falsas (S) son aquellas que aparecen en el archivo *res* pero no en el archivo *gold*.

Precisión: Se refiere a la *Precisión* como una tasa de aceptación, la *Precisión* es la proporción de términos recuperados realmente relevantes, del total de los términos recuperados. Con la precisión podemos medir la calidad del modelo en tareas de clasificación [25]. En la Ecuación 4.1, se puede observar como se calculará la precisión para cada subtarea , utilizando las puntuaciones correctas (C) y falsas (S).

$$Precisión = \frac{C}{C+S} \tag{4.1}$$

Exhaustividad: La exhaustividad también llamada (*Recall* en inglés) nos va a informar sobre la cantidad de elementos que el modelo es capaz de identificar. Es la proporción de términos relevantes recuperados, del total de los términos que son relevantes en la base de datos, independientemente de que éstos, se recuperen o no [25]. En la Ecuación 4.2, se puede observar como se calculará la exhaustividad para cada subtarea, utilizando las puntuaciones correctas (C) y faltantes (M).

$$Exhaustividad = \frac{C}{C+M} \tag{4.2}$$

F_1: Por último, F_1 combina las medidas de *Precisión y Exhaustividad* en un solo valor. Para calcular F_1 se utilizará la ecuación 4.3.

$$F_1 = 2 * \frac{Precision * Exhaustividad}{Precision + Exhaustividad} \tag{4.3}$$

La evaluación de los resultados que se obtengan se verificarán para el escenario principal, que es contemplando ambas subtareas [50]. Para esto se utilizarán las métricas de precisión (ver ecuación 4.4.) y exhaustividad (ver ecuación 4.5). Donde los subíndices corresponden a cada tarea, subtarea A para identificación de entidades y

subtarea B para extracción de relaciones. Los archivos de salida esperados y reales no necesitan coincidir en el identificador de cada entidad, ni en su orden. Las coincidencias del evaluador se basan en el inicio y el final de los intervalos de texto y el tipo correspondiente. Finalmente, para combinar las medidas de precisión y exhaustividad en un solo valor se usa F_1 (ver ecuación 4.3), que se calcula haciendo la media armónica entre la precisión y la exhaustividad.

Precisión (Escenario principal): Para medir la *Precisión* del escenario principal se utilizará la ecuación 4.4, donde C_A son las respuestas correctas de la subtarea A, C_B son las respuestas correctas de la subtarea B, P_A son las respuestas parciales de la subtarea A, S_A son las respuestas falsas de la subtarea A y S_B son las respuestas falsas de la subtarea B.

$$Precisión = \frac{C_A + C_B + 1/2P_A}{C_A + I_A + C_B + P_A + S_A + S_B} \tag{4.4}$$

Exhaustividad (Escenario principal): Para medir la *Exhaustividad* del escenario principal se utilizará la ecuación 4.5, donde I_A son las respuestas incorrectas de la subtarea A, M_A son las respuestas faltantes de la subtarea A y M_B son las respuestas faltantes de la subtarea B.

$$Exhaustividad = \frac{C_A + C_B + 1/2P_A}{C_A + I_A + C_B + P_A + M_A + M_B} \tag{4.5}$$

F_1 **(Escenario principal):** Por último, para calcular el F_1 en el escenario principal se utilizará la ecuación 4.3 explicada anteriormente.

4.4.2. Método de evaluación

Los corpus están clasificados para desarrollo, entrenamiento y pruebas. El último se utiliza para evaluar el desempeño del sistema. El corpus de testing está compuesto por dos archivos, un ".txt" que contiene el texto de prueba y un ".ann" (*gold* que contiene todas las respuestas correctas).

Para una correcta evaluación se ha usado el archivo "score.py" que proporciona la tarea de *eHealth-KD challenge 2021*, esta aplicación nos permite comparar el archivo ".ann" creado por nosotros con el archivo *gold* y posteriormente nos proporciona

la siguiente información para la identificación de entidades: Respuestas correctas, respuestas incorrectas, respuestas parciales, respuestas falsas, respuestas perdidas, métrica exhaustividad, métrica precisión y métrica F_1. Para la extracción de relaciones nos proporciona la siguiente información: Respuestas correctas, respuestas incorrectas, respuestas falsas, respuestas perdidas, métrica exhaustividad, métrica precisión y métrica F_1.

Los resultados del programa "score.py" nos proporciona la misma información para la extracción de relaciones a excepción de las respuestas parciales que no aplican en la tarea de extracción de relaciones.

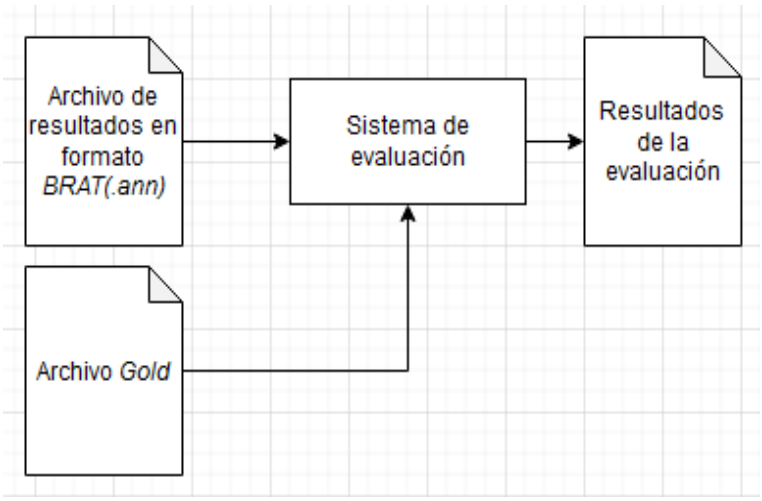

Figura 4.8: Fase de evaluación de los resultados.

Para la creación del archivo ".ann" se ha creado un programa diferente para cada tipo de red. Por ejemplo, para la red *BiLSTM*, los valores de salida se clasifican y el valor más cercano a la respuesta se traduce para obtener la etiqueta, posteriormente se va agregando palabra por palabra y descartándose aquellas palabras con el etiquetado *"None"*.

En la Figura 4.8 podemos observar como la entrada para la evaluación es el archivo de resultados del sistema y el archivo con las respuestas (*gold*), el sistema de evaluación utilizará las ecuaciones de precisión y exhaustividad para obtener el resultado de F_1.

Capítulo 5

Resultados

En este capítulo se muestran los resultados de la etapa de pre-procesamiento, las evaluaciones y resultados de los sistemas propuestos para la identificación de entidades y la extracción de relaciones (usando las métricas de exhaustividad, precisión y F_1), por último, la evaluación del escenario principal (usando las métricas de exahustividad, precisión y F_1).

5.1. Resultados de la etapa de pre-procesamiento

En esta sección se muestran los resultados obtenidos en la etapa de pre-procesamiento (información del corpus, modelos de incrustación de palabras obtenidos y los resultados del pre-procesamiento para cada solución propuesta):

5.1.1. Resultados: Información del corpus

Para esta tesis es necesario conocer como está conformado el corpus a usar. El corpus proporcionado por *ehealth-KD challenge* [50] tiene las siguientes características:

- *Development* (Para el desarrollo del sistema): Conformado por 100 oraciones, 50 en inglés y 50 en español, con aproximadamente 1037 palabras en español y aproximadamente 1452 palabras en inglés.

- *Training* (Para entrenamiento del sistema): Conformado por 100 oraciones en español y aproximadamente 1753 palabras en español.
- *Testing* (Para prueba del sistema): Conformado por 100 oraciones, 50 en inglés y 50 en español, con aproximadamente 1102 palabras en español y aproximadamente 1478 palabras en inglés.

El corpus de desarrollo sirve principalmente para dar una idea de que elementos contiene el corpus de evaluación, a diferencia del corpus de entrenamiento y de evaluación, que son importantes para la evaluación del sistema.
El corpus de entrenamiento (*training*) proporcionado tiene un total de 728 entidades en total, todas las entidades son en español. El corpus de evaluación (*testing*), está compuesto por un total de 934 entidades, de las cuales 451 son en español y 483 son en inglés.
En cuanto a cada tipo de entidad, el corpus de entrenamiento contiene:

- 464 entidades tipo concepto.
- 73 entidades tipo predicado.
- 18 entidades tipo referencia.
- 173 entidades tipo acción.

Mientras que el corpus de evaluación contiene:

- 323 entidades tipo concepto en español y 364 en inglés.
- 37 entidades tipo predicado en español y 63 en inglés.
- 6 entidades tipo referencia en español y 2 en inglés.
- 85 entidades tipo acción en español y 54 en inglés.

Ambos corpus contienen 13 tipos de relaciones diferentes, 678 relaciones en corpus de entrenamiento (en español) y 818 relaciones en corpus de testing (en español e inglés). La oración más larga en el corpus de entrenamineto contiene 47 palabras, en

Tabla 5.1: Cantidad de relaciones por corpus

Tipo de relación	**Entrenamiento**	**Prueba**	**Entrenamiento 2020**
Target	172	130	1281
Subject	107	103	674
In-context	92	198	502
In-place	64	78	304
Is-a	55	50	458
Domain	40	43	269
Causes	36	34	292
Arg	29	29	254
In-time	21	41	126
Part-of	19	3	67
Has-property	15	71	134
Entails	14	21	117
Same-as	14	16	93

el corpus de prueba es de 58 palabras aproximadamente. Finalmente, la oración con más relaciones tiene 19 relaciones.

En la Tabla 5.1 se puede observar la cantidad de relaciones que hay en el corpus de entrenamiento (*training*), el corpus de prueba (*testing*) y debido a que se puede observar que la cantidad de ejemplos es limitada para cada tipo de relación en el corpus de prueba y de entrenamiento, se agregó el corpus de entrenamiento de *eHealth-KD challenge 2020* (no se considera el corpus de evaluación de *eHealth-KD challenge 2020* debido a errores de lectura con el archivo).

5.1.2. Resultados: Incrustación de palabras

Con el fin de mejorar el rendimiento del sistema, se han probado diversos modelos de incrustación de palabras. Para ello se han recopilado varios modelos de incrustación existentes y con el fin de crear varios modelos de incrustación de palabras se han recopilado varios textos (en español) para crear un corpus, este corpus cuenta con 40 millones de palabras y aproximadamente 1 millón de palabras diferentes (vocabulario), con un peso aproximado de 363 megabytes. Para la creación de este corpus

Tabla 5.2: Modelos de incrustación de palabras

Herramienta	**Algoritmo**	**Palabras**	**Dimensión**	**Idioma**	**Origen**
Wor2vec.	*Skip-gram*	20M	490	Español	Creado
Wor2vec.	*Cbow*	20M	490	Español	Creado
Fasttext.	***Skip-gram***	**20M**	**490**	**Español**	**Creado**
Fasttext.	*Skip-gram*	40M	360	Español/Inglés	Creado
Fasttext.	*Skip-gram*	40M	700	Español/Inglés	Creado
Fasttext.	*Cbow*	600B	300	Inglés	*Fasttext* [39]
Fasttext.	*Skip-gram*	600B	300	Inglés	*Fasttext* [39]
Fasttext.	*Skip-gram*	16B	300	Inglés	*Fasttext* [39]
GloVe.	–	42B	300	Inglés	*stanford.edu* [30]

se utilizó una librería de Wikipedia para Python, la cual permite realizar búsquedas y guardar la información en textos. El programa que generó el corpus se encargó de tokenizar la información del corpus de entrenamiento de *eHealth-KD challenge 2021*, posteriormente, se realizó la búsqueda de cada token en Wikipedia y descargó todos los documentos relacionados con la búsqueda. Finalmente, a partir del corpus obtenido, se generaron varios modelos de incrustación de palabras usando los algoritmos proporcionados por *Word2vec* y *Fasttext*.

En la Tabla 5.2 se muestran las diferentes características como lo son la cantidad de palabras que se usó para su entrenamiento, dimensiones, el idioma y la herramienta con la que se creó cada modelo recopilado.

Podemos observar en la Tabla 5.2 que la mitad de los modelos descritos fueron creados a partir del corpus recopilado que se compone de 40 millones de palabras con textos en idioma inglés como español, con un aproximado de 50 % por cada idioma, aproximadamente 900,000 palabras diferentes y un tamaño de total de 360 MB. La otra mitad de los modelos de incrustación obtenidos fueron obtenidos de sus fuentes oficiales respectivas [39] y [30].

5.2. Herramientas utilizadas

En esta subsección se enlista la version de *python*, las librerías de *python* y la versión de cada librería que se utilizaron para implementar las arquitecturas elegidas.

BiLSTM Para la creación de la red *BiLSTM* se utilizaron las librerías y las respectivas versiones de la siguiente lista:

- Python 3.7.11
- Tensorflow 2.6.0
- Keras 2.6.0
- Numpy 1.19.2
- Spacy 3.1.2
- Gensim 4.0.1

BiLSTM* con *CRF* y *BiLSTM* con *CNN Para la creación de la red *BiLSTM* con *CRF* y con *CNN* fueron necesarias las librerías y las respectivas versiones de la lista:

- Python 3.7.11
- Tensorflow 1.15.0
- Keras 2.2.4
- Numpy 1.19.2
- Spacy 3.1.3
- keras_contrib 0.0.2

Las versiones enlistadas no deben cambiarse, debido a que la librería de *keras_contrib* es indispensable para la implementación de *CRF* en la red, además cada librería tiene compatibilidad con versiones muy específicas de *Tensorflow*.

Finalmente, para el etiquetado, los modelos usados en *Spacy* son:

- Español: en_core_web_lg
- Inglés: es_core_news_lg

5.3. Resultados de sistemas para la identificación de entidades

En esta sección podemos observar los resultados de la implementación de los sistemas propuestos para la identificación de entidades y su evaluación.

5.3.1. Sistema *BiLSTM*

La arquitectura de este sistema está compuesta principalmente por capas *BiLSTM*, la arquitectura de este sistema tiene específicamente:

- 1 capa incrustación (*Embedding*).
- 2 capas *BiLSTM* con 70 unidades.
- 1 capa *BiLSTM* con 10 unidades.
- 1 capa densa con 7 unidades.
- 1 capa dropout con perdida de 0.3.
- 1 capa densa con 1 unidad.

En la Figura 5.1 podemos observar el orden de las capas usadas en ese sistema, la red se entrenó con 21 épocas y un tamaño de lote (*batch size*) de 40. Finalmente, para la evaluación se hizo uso de los archivos proporcionados por *eHealth-KD challenge 2021*.

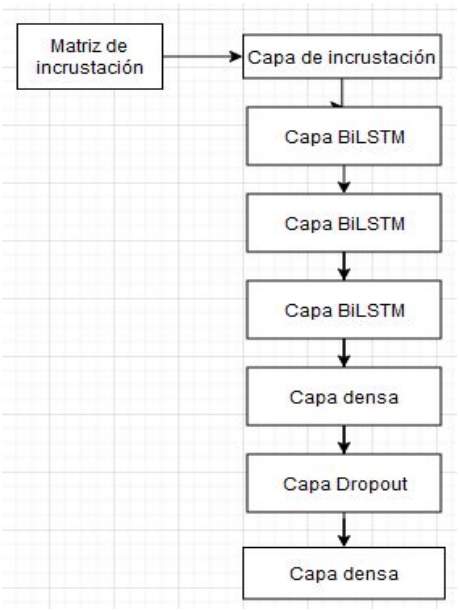

Figura 5.1: Arquitectura red *BiLSTM*.

5.3.2. Sistema *BiLSTM-CRF*

La arquitectura de este sistema va orientada a la incrustación de palabras, la arquitectura está compuesta principalmente por una capa *BiLSTM* y una capa *CRF* que beneficia la clasificación de las entidades, la arquitectura de este sistema está compuesta específicamente por:

- 1 capa incrustación (*Embedding*).
- 1 capa *BiLSTM* con 490 unidades.
- 1 capa *LSTM* con 980 unidades.
- 1 capa densa con 9 unidades.
- 1 capa *CRF* con 9 unidades.

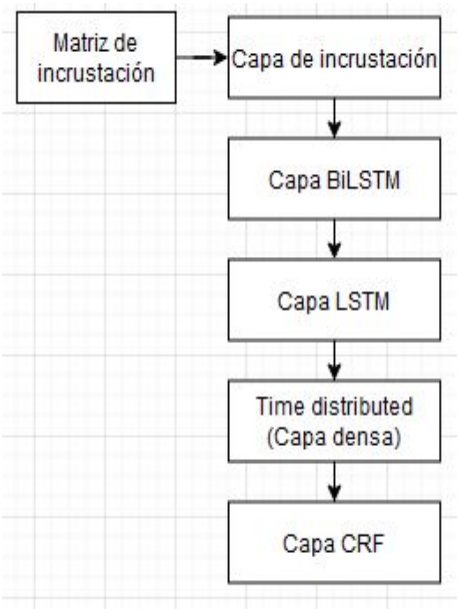

Figura 5.2: Arquitectura red *BiLSTM-CRF*.

En la Figura 5.2 podemos observar el orden de las capas usadas en ese sistema, la red se entrenó con 21 épocas y un tamaño de lote (*batch size*) de 110. Para la red *BiLSTM* con *CRF* las pruebas se hicieron con los mismos estilos de anotación mostrados en la sección de pre-procesamiento.

5.3.3. Sistema *BiLSTM-CNN*

La arquitectura de este sistema está compuesta principalmente por capas *BiLSTM* y *CNN* orientadas a la incrustación de caracteres. El bloque de caracteres se encarga de analizar las palabras a nivel carácter para posteriormente realizar la clasificación por medio de una red *BiLSTM*. La arquitectura de este sistema está compuesta específicamente por:

- 1 bloque de caracteres (capas *CNN*).
- 1 capa *BiLSTM* con 128 unidades.
- 1 capa *dropout* con 128 unidades con perdida de 0.5.

- 1 capa densa con 128 unidades.
- 1 capa *dropout* con 9 unidades con perdida de 0.5.

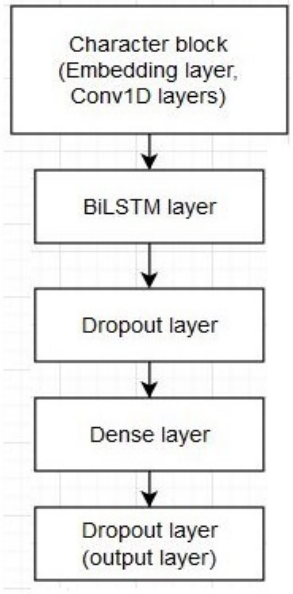

Figura 5.3: Arquitectura red *BiLSTM-CNN*.

La arquitectura de este sistema, a diferencia del sistema *BiLSTM-CRF*, se orienta a la incrustación de caracteres [35]. En la Figura 5.3 podemos observar el orden de las capas usadas en ese sistema [14], la red se entrenó con 21 épocas y un tamaño de lote (*batch size*) de 110. Al igual que en sistemas anteriores, las pruebas se hicieron con los mismos estilos de anotación mostrados en la sección de pre-procesamiento.

5.3.4. Evaluación de los sistemas propuestos para identificación de entidades

En la Tabla 5.3 y la Tabla 5.4 se muestra una comparación de los estilos de anotación en los sistemas de identificación de entidades propuestos (para los etiquetadores *Spacy* y *Freeling*), para obtener los resultados se ejecutó el modelo 10 veces y se obtuvo el promedio de los resultados. Se puede observar en la Tabla 5.4 que el

Tabla 5.3: Resultados de los sistemas para identificación de entidades, comparando estilos de anotación (*Spacy*).

Sistema	Exhaustividad	Precisión	F_1
BiLSTM-CNN - Anotación P	**0.6685**	**0.4853**	**0.5616**
BiLSTM-CNN - Anotación BIO	0.6843	0.4715	0.5581
BiLSTM-CNN - Anotación BILOU	0.6766	0.4696	0.5540
BiLSTM-CRF - Anotación P	**0.6342**	**0.5460**	**0.5868**
BiLSTM-CRF - Anotación BIO	0.6800	0.4950	0.5726
BiLSTM-CRF - Anotación BILOU	0.6935	0.4752	0.5639

Tabla 5.4: Resultados de los sistemas para identificación de entidades, comparando estilos de anotación (*Freeling*).

Sistema	Exhaustividad	Precisión	F_1
BiLSTM-CNN - Anotación P	0.5834	0.3964	0.4720
BiLSTM-CNN - Anotación BIO	0.5021	0.4957	0.4989
BiLSTM-CNN - Anotación BILOU	**0.5503**	**0.5125**	**0.5307**
BiLSTM-CRF - Anotación P	0.4850	0.5107	0.4975
BiLSTM-CRF - Anotación BIO	0.5187	0.4681	0.4921
BiLSTM-CRF - Anotación BILOU	**0.4920**	**0.5349**	**0.5125**

estilo de anotación BILOU se favorece del etiquetado de *Freeling*, sin embargo, los resultados no se comparan con los del etiquetador *Spacy*.

Los mejores resultados se obtuvieron con los modelos de incrustación de *Fasttext*, no hubo mucha diferencia entre los resultados de los modelos más grandes de *Fasttext*, así que el modelo que se utilizó finalmente fue el de creación propia con 40 millones de palabras y 360 dimensiones que se observó en la Tabla 5.2 de la sección 5.1.2.

Finalmente, debido a que en la anotación P obtuvo los mejores resultados con *Spacy*, en la Tabla 5.5 se muestra una comparación de los sistemas añadiendo la propuesta de etiquetado *POS* con los diferentes estilos de anotación (usando como base el etiquetado *Spacy* con el estilo de anotación P).

Tabla 5.5: Resultados de los sistemas para identificación de entidades, añadiendo la propuesta con diferentes estilos de anotación *Spacy*).

Sistema	Exhaustividad	Precisión	F_1
BiLSTM-CNN - Anotación P (base)	**0.6685**	**0.4853**	**0.5616**
BiLSTM-CNN (P) - POS Anotación P	0.6483	0.4806	0.5516
BiLSTM-CNN (P) - POS Anotación BIO	0.6344	0.4897	0.5520
BiLSTM-CNN (P) - POS Anotación BILOU	0.6522	0.4826	0.5540
BiLSTM-CRF - Anotación P (base)	**0.6342**	**0.5460**	**0.5868**
BiLSTM-CRF (P) - POS Anotación P	0.6750	0.5125	0.5821
BiLSTM-CRF (P) - POS Anotación BIO	0.6310	0.5102	0.5620
BiLSTM-CRF (P) - POS Anotación BILOU	0.6363	0.5195	0.5710

5.4. Resultados de sistemas para la extracción de relaciones

En esta sección podemos observar los resultados de la implementación de los sistemas propuestos para la extracción de relaciones y su evaluación correspondiente.

5.4.1. Sistema *BiLSTM-CRF*

La arquitectura de este sistema es la misma usada en la identificación de entidades, la arquitectura de este sistema está compuesta específicamente por:

- 1 capa *Embedding* (Incrustación).
- 1 capa *BiLSTM* con 490 unidades.
- 1 capa *LSTM* con 490 unidades.
- 1 capa densa con 3 unidades.
- 1 capa *CRF* con 3 unidades.

Para la capa densa y la capa *CRF* la cantidad de unidades utilizadas depende de la cantidad de clases que se evaluen en cada propuesta. Para las pruebas se usarón lotes (*batch size*) de 35,70,28 y se entrenó de 10 a 35 épocas dependiendo de las configuraciones que se muestran a continuación.

5.4.2. Sistema *BERT* (*BERT multilingual base model (cased)*)

Debido a los resultados obtenidos por la arquitectura *BiLSTM-CRF* en la extracción de relaciones se implementó la arquitectura *BERT*.

La arquitectura de este sistema está compuesta específicamente por:

- Modelo preentrenado *BERT multilingual base model (cased)*.
- 1 capa *Dropout* (.5).
- 1 capa de transformación lineal (*linear layer*).

Para el entrenamiento de esta red se realizó con una longitud máxima de 56 palabras por oración, 35 épocas y una tasa de aprendizaje de 5e-5 (0.00005), con una espera de 7 épocas antes de empezar a reducir la tasa de aprendizaje por época.

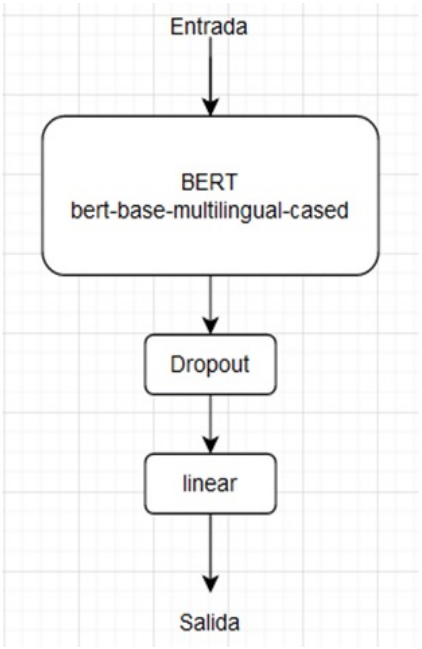

Figura 5.4: Arquitectura red *BERT*.

Como se ha mencionado se implementó un modelo de *BERT* pre-entrenado (*BERT multilingual base model*) [19], para introducir la información dentro de la

Tabla 5.6: Ejemplo de entrada de etiquetado para red *BERT*

Tipo de relación	**Valor entrada**
None	0
Target	1
Subject	2
Is-a	3
Same-as	4
In-context	5
...	...
Arg	13

red se hace uso de la librería *BertTokenizer* para obtener los valores de cada palabra dentro del modelo ya entrenado.

Para la red *BERT*, se utilizó solamente la opción 4 explicada en la sección 4.2.4, cabe aclarar que para introducir las etiquetas en la arquitectura *BERT* y a diferencia de la arquitectura *BiLSTM-CRF* (que utiliza como entradas un vector de ceros y un uno), solo es necesario convertir las etiquetas a número entero positivo (cada etiqueta es representada por un número diferente), en este caso las etiquetas corresponden a un tipo de relación como se puede ver en el ejemplo de la Tabla 5.6.

5.4.3. Evaluación de los sistemas propuestos

Se realizaron varias pruebas para cada tipo de relación respecto la opción 2 con diferentes entradas (la entrada simple, entrada con patrones y la entrada con marcado descriptivo) que son explicadas en la sección de pre-procesamiento.

Los resultados mostrados a continuación están evaluados conforme al total de relaciones del corpus de prueba (*test*), es decir, aunque sea solo un tipo de relación, se evalúa como si estuviesen todos los tipos de relación incluidos, en la 5.7 se puede observar los resultados para cada tipo de relación que se obtuvieron con la opción 2 y la entrada simple.

En la Tabla 5.8 se puede observar los resultados para cada tipo de relación que se obtuvieron con la opción 2 y la entrada con patrones, como se puede notar este tipo de entrada ayuda en el aumento de la precisión del sistema, sin embargo, diminuye considerablemente el valor de la exhaustividad, lo que significa una disminución

Tabla 5.7: Resultados para cada tipo de relación con opción 2 - Entrada simple (BiLSTM-*CRF*)

Tipo de relación	**Exhaustividad**	**Precisión**	**F_1**	**F_1** máximo
Target	0.0340	0.1783	0.0571	0.2102
Subject	0.0461	0.1357	0.0689	0.1312
In-context	0.0145	0.0727	0.0242	0.2937
In-time	0.0085	0.1061	0.0157	0.0792
In-place	0.0048	0.0634	0.0090	0.0967
Causes	0.0012	0.0112	0.0021	0.0680
Is-a	0.0036	0.0681	0.0069	0.0566
Has-property	0.0097	0.0689	0.0170	0.1162
Same-as	0.0000	0.0000	0.0000	0.0120
Entails	0.0024	0.0139	0.0041	0.0381
Domain	0.0012	0.2000	0.0024	0.0791
Arg	0.0048	0.1818	0.0094	0.0635
Part-of	0.0000	0.0000	0.0000	0.0048

considerable de respuestas que da el sistema, es por ello que en muchas relaciones su F_1 es de 0.

En los resultados de la Tabla 5.9 se puede observar los resultados para cada tipo de relación que se obtuvieron con la opción 2 y la entrada de marcado descriptivo, se puede notar que en algunas relaciones el valor de F_1 aumenta, esto es debido a que esta entrada sacrifica precisión a cambio de obtener más variedad de resultados (aumenta el valor de exhaustividad), lo que se puede observar en la Tabla 5.12.

Para la representación de la opción 4 en la red *BiLSTM-CRF*, fue necesario cambiar algunos ajustes, el modelo se entrenó durante 70 épocas, con un lote de 20, 42 palabras como máximo por oración y activando la opción que nos permite guardar el mejor resultado en la red al entrenar. Además, para el entrenamiento se ajustó el corpus de entrenamiento para introducir 2 oraciones sin relación por cada oración que contenga una relación (con la finalidad de evitar un sobre ajuste en el descartado de relaciones), los resultados pueden ser observados en la Tabla 5.10.

Finalmente, para la representación de la opción 4 en la red *BERT* es necesario realizar el entrenamiento en 2 pasos para obtener mejores resultados, entrenando en primera instancia solo con oraciones que contengan una relación (como si se

Tabla 5.8: Resultados para cada tipo de relación con opción 2 - Entrada con patrones (BiLSTM-*CRF*)

Tipo de relación	**Exhaustividad**	**Precisión**	**F_1**	**F_1** máximo
Target	0.0109	0.1429	0.0203	0.2102
Subject	0.0072	0.3333	0.0142	0.1312
In-context	0.0097	0.1455	0.0182	0.2937
In-time	0.0000	0.0000	0.0000	0.0792
In-place	0.0048	0.0851	0.0091	0.0967
Causes	0.0000	0.0000	0.0000	0.0680
Is-a	0.0036	0.3333	0.0036	0.0566
Has-property	0.0000	0.0000	0.0000	0.1162
Same-as	0.0000	0.0000	0.0000	0.0120
Entails	0.0000	0.0000	0.0000	0.0381
Domain	0.0000	0.0000	0.0000	0.0791
Arg	0.0012	0.0526	0.0023	0.0635
Part-of	0.0000	0.0000	0.0000	0.0048

Tabla 5.9: Resultados para cada tipo de relación con opción 2 - Entrada de marcado descriptivo (*BiLSTM-CRF*)

Tipo de relación	**Exhaustividad**	**Precisión**	**F_1**	**F_1** máximo
Target	0.0449	0.1588	0.0700	0.2102
Subject	0.0510	0.1405	0.0748	0.1312
In-context	0.0340	0.1366	0.0544	0.2937
In-time	0.0036	0.0491	0.0067	0.0792
In-place	0.0024	0.0312	0.0045	0.0967
Causes	0.0012	0.0113	0.0021	0.0680
Is-a	0.0012	0.0833	0.0023	0.0566
Has-property	0.0133	0.0774	0.0228	0.1162
Same-as	0.0000	0.0000	0.0000	0.0120
Entails	0.0024	0.0344	0.0045	0.0381
Domain	0.0024	0.1429	0.0047	0.0791
Arg	0.0060	0.2174	0.0118	0.0635
Part-of	0.0024	0.0040	0.0030	0.0048

Tabla 5.10: Resultados para cada tipo de relación con opción 4 - (*BiLSTM-CRF*)

$\mathbf{F}_1$	**Precisión**	**Exhaustividad**	**Correctas**	**Falsas**	**Faltantes**
0.1020	0.1201	0.0887	73	535	750

Tabla 5.11: Resultados para cada tipo de relación con opción 4 - (*BERT*)

$\mathbf{F}_1$	**Precisión**	**Exhaustividad**	**Correctas**	**Falsas**	**Faltantes**
0.3042	0.4174	0.2394	197	275	626

entrenase la red para clasificación de relaciones) y posteriormente un entrenamiento incluyendo las oraciones que contienen una relación y las que no. A diferencia de la red *BiLSTM-CRF*, la red *BERT* aprende mejor con un mayor número de muestras (oraciones), a pesar de que muchas de estas muestras sean oraciones que no contengan una relación, esto quiere decir que el modelo aprovecha mejor estas muestras y no se produce el sobre ajuste que se produce en la red *BiLSTM-CRF*, los resultados se pueden observar en la Tabla 5.11.

En la Tabla 5.12 se puede observar una comparación de resultados totales (uniendo los resultados obtenidos de cada tipo de relación), implementando los diferentes tipos de entrada propuestos. Como se puede notar la entrada simple como la entrada de marcado descriptivo, poseen sus ventajas y sus desventajas, pese a que por separado tengan resultados diferentes, los resultados en conjunto terminan siendo muy similares.

Por último, en la Tabla 5.13 se muestra una comparación de los mejores resultados de los sistemas de extracción de relaciones semánticas, según cada opción de etiquetado. Al igual que en la identificación de entidades se usó el mismo modelo de incrustación *Fasttext*, de 40 millones de palabras y 360 dimensiones.

Tabla 5.12: Resultados totales de opción 2 (BiLSTM-*CRF*) con las diferentes entradas propuestas.

Sistema	**Exhaustividad**	**Precisión**	$\mathbf{F}_1$
Opción 2 (Entrada simple)	0.1312	0.0889	0.1060
Opción 2 (Entrada patrones)	0.0388	0.0927	0.0547
Opción 2 (Entrada de marcado descriptivo)	0.1652	0.0770	0.1051

Tabla 5.13: Resultados de los mejores resultados en las opciones de etiquetado 2 y 4, para la extracción de relaciones.

Sistema	**Exhaustividad**	**Precisión**	**F_1**
BiLSTM Opción 2	0.1312	0.0889	0.1060
BERT Opción 4	0.2394	0.4174	0.3042

5.5. Resultados de los sistemas propuestos en *eHealth-KD Challenge 2021*

En esta sección se comparan los mejores resultados obtenidos en cada subtarea y en el escenario principal, esta comparación se realiza con los resultados de los demás participantes del evento *eHealth-KD Challenge 2021* [50].

5.5.1. *Comparación de resultados de los sistemas propuestos para la subtarea A y subtarea B*

Tabla 5.14: Resultados de los mejores sistemas para cada escenario.

Arquitectura	Subtarea	*Precisión*	*Exhaustividad*	$Medida - F_1$
BiLSTM	A	0.5460	0.6342	0.5868
BERT	B	0.4174	0.2394	0.3042

La mejor puntuación obtenida dentro de los sistemas propuestos para las subtareas A y B se puede ver en la Tabla 5.14, estos resultados son los que se utilizarán para compararse con los demas participantes de *eHealth-KD Challenge 2021* [50].

En la Tabla 5.15 se muestra una comparación de los resultados de la subtarea A con los participantes del *eHealth-KD challenge 2021*. También se muestra cuál es la arquitectura de red neuronal en la que se basa el sistema de cada participante.

En la Tabla 5.16 se muestra una comparación de los resultados de la subtarea B con los participantes del *eHealth-KD challenge 2021*. También se muestra cuál es la arquitectura de red neuronal en la que se basa el sistema de cada participante.

Tabla 5.15: Comparación de resultados de la subtarea A

Equipo	*Precisión*	*Exhaustividad*	$Medida - F_1$	Arquitectura
PUCRJ-PUCPR-UFMG	0.71491	0.69733	0.70601	*BERT*
Vicomtech	0.69987	0.74706	0.68413	*BERT*
IXA	0.61372	0.69840	0.65333	*BERT*
UH-MMM	0.54604	0.68503	0.60769	*BiLSTM*
Sistema propuesto	**0.5460**	**0.6342**	**0.5868**	***BiLSTM***
uhKD4	0.51751	0.53743	0.52728	*BiLSTM*
Yunnan-Deep	0.52036	0.27166	0.33406	*BiLSTM*
baseline	0.35034	0.27166	0.30602	–
JAD	0.31579	0.22460	0.26250	*BERT*

Tabla 5.16: Comparación de resultados de la subtarea B

Equipo	*Precisión*	*Exhaustividad*	$Medida - F_1$	Arquitectura
IXA	0.4535	0.4094	0.4304	*BERT*
Vicomtech	0.5418	0.2831	0.3719	*BERT*
uhKD4	0.5562	0.2223	0.3177	*CNN*
Sistema propuesto	**0.4147**	**0.2394**	**0.3042**	***BERT***
PUCRJ-PUCPR-UFMG	0.3665	0.2053	0.2632	*BERT*
Sistema propuesto	**0.1312**	**0.0889**	**0.1060**	***BiLSTM***
UH-MMM	0.0772	0.0413	0.0538	*BiLSTM*
baseline	0.4375	0.0170	0.0327	–
JAD	0.3750	0.0036	0.0072	*BERT*

Tabla 5.17: Comparación de resultados del escenario principal

Equipo	*Precisión*	*Exhaustividad*	$Medida - F_1$
Vicomtech	0.5407	0.5346	0.5310
PUCRJ-PUCPR-UFMG	0.5684	0.5027	0.5283
IXA	0.4645	0.5386	0.4988
Sistema propuesto	**0.5070**	**0.4493**	**0.4764**
uhKD4	0.4852	0.3743	0.3743
UH-MMM	0.2916	0.4037	0.3386
baseline	0.3370	0.1768	0.2320
JAD	0.2344	0.0714	0.1094

5.5.2. *Comparación de los resultados en el escenario principal*

Con la finalidad de obtener la evaluación del escenario principal se usaron las puntuaciones de los mejores sistemas propuestos de cada subtarea, es decir, 522 correctas, 190 incorrectas, 228 parciales, 493 falsas, 195 perdidas (para la identificación de entidades) y 197 correctas, 275 falsas, 626 perdidas (para la extracción de relaciones).

Estas puntuaciones se utilizaron para obtener las métricas de exhaustividad , precisión y F_1 con las ecuaciones vistas en la sección 4.4.1 y como se puede observar de la Ecuación 5.1 a la Ecuación 5.5.

$$Precisión = \frac{500 + 197 + 1/2(186)}{500 + 104 + 197 + 186 + 296 + 275} \tag{5.1}$$

$$Precisión = \frac{790}{1,558} = 0{,}5070 \tag{5.2}$$

$$Exhaustividad = \frac{500 + 197 + 1/2(186)}{500 + 104 + 197 + 186 + 145 + 626} \tag{5.3}$$

$$Exhaustividad = \frac{790}{1,758} = 0{,}4493 \tag{5.4}$$

$$F_1 = 2 * \frac{0{,}5070 * 0{,}4493}{0{,}5070 + 0{,}4493} = 0{,}4764 \tag{5.5}$$

Como resultados se obtuvo 0.5070 de precisión, 0.4493 de exhaustividad y 0.4764 de F_1 en el escenario principal. En la Tabla 5.17 se muestra una comparación de

los participantes del *eHealth-KD challenge 2021*, con los resultados obtenidos en el escenario principal .

Conclusiones

Para resolver la identificación de entidades y extracción de relaciones semánticas se implementaron algoritmos usando arquitecturas de redes neuronales recurrentes *BiLSTM* (vistas en el estado del arte), para la identificación de entidades se exploró principalmente la propuesta de solución dirigida a encontrar patrones en el etiquetado *POS* (usando *Spacy* y *Freeling*), mientras que para la extracción de relaciones se exploró principalmente la propuesta del etiquetado de dependencias de *Spacy*. También se han comparado los resultados de la arquitectura *BiLSTM* con una arquitectura *BERT* para la extracción de relaciones.

Se puede observar en la subtarea A, que los resultados varian según el etiquetador (*POS*) y el estilo de anotación que se implementen, es decir, el impacto del estilo de anotación no es el mismo con diferentes etiquetadores (*POS*).

Durante el proceso de experimentación de la subtarea B, se observó que los modelos basados en redes *BiLSTM* tienen muchas limitaciones a la hora de aprender características del contexto de una oración. Por esa razón se implementó una arquitectura usando un modelo pre-entrenado *BERT* que aprende mejor las características del contexto de una oración, pero necesita de un conjunto de datos más amplio para generar un aprendizaje con mejores resultados, por lo tanto, no se utilizó la misma estrategia que con las redes *BiLSTM* en la subtarea B (un sistema para detectar un tipo de relación en específico).

En la subtarea de identificación de entidades los resultados del sistema logran superar el quinto lugar y tiene el segundo mejor resultado implementando una red *BiLSTM*. En la subtarea de extracción de relaciones se logró mejorar el resultado respecto a las arquitecturas *BiLSTM*, en cuanto a la solución implementada usando *BERT* se logró posicionar los resultados en un virtual cuarto lugar.

Utilizando los resultados obtenidos de los sistemas de identificación de entidades (subtarea A) y extracción de relaciones (subtarea B), se obtiene la evaluación para ambas subtareas (escenario principal). El resultado del escenario principal logró superar el F_1 del cuarto participante, como se puede observar en la Tabla 5.17. Con los resultados del escenario principal se puede concluir que se lograron los objetivos planteados en esta tesis.

Finalmente, como trabajo a futuro se plantea mejorar las estrategias implementadas con *Spacy* y *Freeling* (en ambas subtareas), se plantea implementar otras arquitecturas basadas en redes *BiLSTM* (incluyendo incrustaciones a nivel palabra y carácter dentro del mismo modelo) para mejorar los resultados. También se plantea desarrollar estrategias para implementar la arquitectura *BERT* en la identificación de entidades, así como implementar soluciones para ambas subtareas utilizando otros modelos pre-entrenados orientados al español como los modelos basados en BETO (la versión en español de *BERT*), o *MarIA* (que es una familia de modelos de lenguaje en español) [28].

Bibliografía

[1] Moreno A. Procesamiento del lenguaje natural ¿qué es? 2018. URL https://www.iic.uam.es/inteligencia/que-es-procesamiento-del-lenguaje-natural/.

[2] B. Fernández-Manjón y J. L. Sierra A. Fernández-Valmayor, A. Navarro. Los lenguajes de marcado y el paradigma documental. 2022. URL http://elies.rediris.es/elies24/fernandezvalmayor_cap3.htm.

[3] Liliya Akhtyamova. Named entity recognition in spanish biomedical literature: Short review and bert model. En *2020 26th Conference of Open Innovations Association (FRUCT)*, págs. 1–7. IEEE, 2020. ISSN 978-952-69244-2-7. doi: 10.23919/FRUCT48808.2020.9087359.

[4] Dayany Alfaro-González, Dalianys Pérez-Perera, Gilberto González-Rodríguez, Antonio Jesús Otaño-Barrera, y Rocío Cruz Linares. uhkd4 at ehealth-kd challenge 2021: Deep learning approaches for knowledge discovery from spanish biomedical documents. En *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2021) co-located with the Conference of the Spanish Society for Natural Language Processing (SEPLN 2021), XXXVII International Conference of the Spanish Society for Natural Language Processing., Málaga, Spain, September, 2021*, tomo 2943 de *CEUR Workshop Proceedings*, págs. 703–711. CEUR-WS.org, 2021. URL http://ceur-ws.org/Vol-2943/ehealth_paper5.pdf.

[5] Jorge Mederos Alvarado, Ernesto Quevedo Caballero, Alejandro Rodríguez Pérez, y Rocío Cruz Linares. UH-MAJA-KD at ehealth-kd challenge 2019: Deep learning models for knowledge discovery in spanish ehealth documents. En *Pro-*

ceedings of the Iberian Languages Evaluation Forum co-located with 35th Conference of the Spanish Society for Natural Language Processing, IberLEF@SEPLN 2019, Bilbao, Spain, September 24th, 2019, tomo 2421 de *CEUR Workshop Proceedings*, págs. 85–94. CEUR-WS.org, 2019. URL http://ceur-ws.org/Vol-2421/eHealth-KD_paper_9.pdf.

[6] Edgar Andrés. IXA at ehealth-kd challenge 2021: Generic sequence labelling as relation extraction approach. En *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2021) co-located with the Conference of the Spanish Society for Natural Language Processing (SEPLN 2021), XXXVII International Conference of the Spanish Society for Natural Language Processing., Málaga, Spain, September, 2021*, tomo 2943 de *CEUR Workshop Proceedings*, págs. 668–674. CEUR-WS.org, 2021. URL http://ceur-ws.org/Vol-2943/ehealth_paper1.pdf.

[7] Edgar Andrés, Oscar Sainz, Aitziber Atutxa, y Oier Lopez de Lacalle. IXA-NER-RE at ehealth-kd challenge 2020. En *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020) co-located with 36th Conference of the Spanish Society for Natural Language Processing (SEPLN 2020), Málaga, Spain, September 23th, 2020*, tomo 2664 de *CEUR Workshop Proceedings*, págs. 151–162. CEUR-WS.org, 2020. URL http://ceur-ws.org/Vol-2664/eHealth-KD_paper8.pdf.

[8] BBVA. ¿qué es el aprendizaje automático? 2016. URL https://www.bbvaopenmind.com/tecnologia/mundo-digital/que-es-el-aprendijaze-profundo/.

[9] Àlex Bravo, Pablo Accuosto, y Horacio Saggion. Lastus-taln at iberlef 2019 ehealth-kd challenge: Deep learning approaches to information extraction in biomedical texts. En *Proceedings of the Iberian Languages Evaluation Forum co-located with 35th Conference of the Spanish Society for Natural Language Processing, IberLEF@SEPLN 2019, Bilbao, Spain, September 24th, 2019*, tomo 2421 de *CEUR Workshop Proceedings*, págs. 51–59. CEUR-WS.org, 2019. URL http://ceur-ws.org/Vol-2421/eHealth-KD_paper_5.pdf.

[10] Raúl E. López Briega. Introducción al procesamiento del lenguaje natural. 2022. URL https://iaarbook.github.io/procesamiento-del-lenguaje-natural/. IAAR: El sitio de la comunidad argentina de inteligencia artificial IAAR.

[11] Andrea Belén Calibar, Jorge Holleger, y Raúl O. Klenzi. Análisis de similitud en documentos de texto mediante técnicas de ciencia de datos basadas en aprendizaje profundo (deep learning). En *XX Workshop de Investigadores en Ciencias de la Computaci éon*, págs. 246–250. Universidad Nacional del Nordeste (UNNE), 2018. ISBN 78-987-3619-27-4. URL https://core.ac.uk/download/pdf/158326657.pdf.

[12] Carmelo Bonilla Carrión. *Redes convolucionales*. Trabajo fin de grado, Universidad de Sevilla, 2020.

[13] Neus Català y Mario Martín. Coin_flipper at ehealth-kd challenge 2019: Voting lstms for key phrases and semantic relation identification applied to spanish ehealth texts. En *Proceedings of the Iberian Languages Evaluation Forum co-located with 35th Conference of the Spanish Society for Natural Language Processing, IberLEF@SEPLN 2019, Bilbao, Spain, September 24th, 2019*, tomo 2421 de *CEUR Workshop Proceedings*, págs. 17–25. CEUR-WS.org, 2019. URL http://ceur-ws.org/Vol-2421/eHealth-KD_paper_1.pdf.

[14] Jason P. C. Chiu y Eric Nichols. Named entity recognition with bidirectional lstm-cnns. *CoRR*, abs/1511.08308, 2015. URL http://arxiv.org/abs/1511.08308.

[15] Antonio Circiello, Andrea Cappelli, Sonia Bergamaschi, y Marco Varone. Effects of semantic analysis on named-entity recognition with conditional random fields. En *Proceedings of the 25th Italian Symposium on Advanced Database Systems, Squillace Lido (Catanzaro), Italy, June 25-29, 2017*, tomo 2037 de *CEUR Workshop Proceedings*, pág. 234. CEUR-WS.org, 2017. URL http://ceur-ws.org/Vol-2037/paper_33.pdf.

[16] Juan Pablo Consuegra-Ayala y Manuel Palomar. Uh-matcom at ehealth-kd challenge 2020. En *Proceedings of the Iberian Languages Evaluation Forum (Iber-*

LEF 2020) co-located with 36th Conference of the Spanish Society for Natural Language Processing (SEPLN 2020), Málaga, Spain, September 23th, 2020, tomo 2664 de *CEUR Workshop Proceedings*, págs. 112–124. CEUR-WS.org, 2020. URL http://ceur-ws.org/Vol-2664/eHealth-KD_paper4.pdf.

[17] IBM Corporation. Conceptos clave: entidad, atributo y tipo de entidad. 2021. URL https://www.ibm.com/docs/es/imdm/11.6?topic=concepts-key-entity-attribute-entity-type.

[18] Viviana Cotik, Horacio Rodríguez, y Jorge Vivaldi. Spanish named entity recognition in the biomedical domain. En *Information Management and Big Data*, Communications in computer and information science, págs. 233–248. Springer International Publishing, Cham, 2019. doi:https://doi.org/10.1007/978-3-030-11680-4_23.

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, y Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, y Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. En *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, págs. 4171–4186. Association for Computational Linguistics, 2019. doi:10.18653/v1/n19-1423. URL https://doi.org/10.18653/v1/n19-1423.

[21] IBM Cloud Education. Redes neuronales. 2020. URL https://www.ibm.com/mx-es/cloud/learn/neural-networks.

[22] Real Academia Española. semántico. 2022. URL https://dle.rae.es/sem%C3%A1ntico.

[23] Pech May F. Procesamiento de lenguaje natural con deeplearning. 2018. URL http://rios.tecnm.mx/cdistribuido/recursos/DLScr/PLN.html.

[24] Hermenegildo Fabregat, Andres Duque Fernandez, Juan Martínez-Romo, y Lourdes Araujo. Nlp_uned at ehealth-kd challenge 2019: Deep learning for named entity recognition and attentive relation extraction. En *Proceedings of the Iberian Languages Evaluation Forum co-located with 35th Conference of the Spanish Society for Natural Language Processing, IberLEF@SEPLN 2019, Bilbao, Spain, September 24th, 2019*, tomo 2421 de *CEUR Workshop Proceedings*, págs. 67–77. CEUR-WS.org, 2019. URL http://ceur-ws.org/Vol-2421/eHealth-KD_paper_7.pdf.

[25] Juan Diego Gómez Fierros. *Poblado automático de Ontologías Espaciales a Partir de Texto no Estructurado*. Tesis de maestría, Centro Nacional de Investigación y Desarrollo Tecnológico, Departamento de Ciencias Computacionales, 2012.

[26] Iakes Goenaga, Sergio Santana, Sara Santiso Gonzáles, Koldo Gojenola, Alicia Pérez, y Arantza Casillas. Ixamed at ehealth-kd challenge 2019: Using different paradigms to solve clinical relation extraction. En *Proceedings of the Iberian Languages Evaluation Forum co-located with 35th Conference of the Spanish Society for Natural Language Processing, IberLEF@SEPLN 2019, Bilbao, Spain, September 24th, 2019*, tomo 2421 de *CEUR Workshop Proceedings*, págs. 43–50. CEUR-WS.org, 2019. URL http://ceur-ws.org/Vol-2421/eHealth-KD_paper_4.pdf.

[27] The Stanford Natural Language Processing Group. Stanford dependencies. 2022. URL https://nlp.stanford.edu/software/stanford-dependencies.html.

[28] Asier Gutiérrez-Fandiño, Jordi Armengol-Estapé, Marc Pàmies, Joan Llop-Palao, Joaquín Silveira-Ocampo, Casimiro Pio Carrino, Aitor Gonzalez-Agirre, Carme Armentano-Oller, Carlos Rodríguez Penagos, y Marta Villegas. Spanish language models. *CoRR*, abs/2107.07253, 2021. URL https://arxiv.org/abs/2107.07253.

[29] hmong.es. Dependencias universales. 2022. URL https://hmong.es/wiki/Universal_Dependencies.

[30] Christopher D. Manning Jeffrey Pennington, Richard Socher. Glove: Global vectors for word representation. 2015. URL https://nlp.stanford.edu/projects/glove/.

[31] Copara Zea Jenny Linet. *Reconocimiento de entidades nombradas para el idioma español utilizando Conditional Random Fields con características no supervisadas*. Tesis de maestría, Universidad Católica San Pablo - Perú, 2017.

[32] Zihan Liu, Genta Indra Winata, y Pascale Fung. Zero-resource cross-domain named entity recognition. *CoRR*, abs/2002.05923, 2020. URL https://arxiv.org/abs/2002.05923.

[33] Patricio Loncomilla. Deep learning: Redes convolucionales. 2022. URL https://ccc.inaoep.mx/~pgomez/deep/presentations/2016Loncomilla.pdf.

[34] Pilar López-Úbeda, José Manuel Perea Ortega, Manuel Carlos Díaz-Galiano, María Teresa Martín-Valdivia, y Luis Alfonso Ureña López. SINAI at ehealth-kd challenge 2020. En *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020) co-located with 36th Conference of the Spanish Society for Natural Language Processing (SEPLN 2020), Málaga, Spain, September 23th, 2020*, tomo 2664 de *CEUR Workshop Proceedings*, págs. 141–150. CEUR-WS.org, 2020. URL http://ceur-ws.org/Vol-2664/eHealth-KD_paper7.pdf.

[35] Edward Ma. Character embedding. 2018. URL https://github.com/makcedward/nlp/blob/master/sample/nlp-character_embedding.ipynb.

[36] Salvador Medina y Jordi Turmo. TALP-UPC at ehealth-kd challenge 2019: A joint model with contextual embeddings for clinical information extraction. En *Proceedings of the Iberian Languages Evaluation Forum co-located with 35th Conference of the Spanish Society for Natural Language Processing, IberLEF@SEPLN 2019, Bilbao, Spain, September 24th, 2019*, tomo 2421 de *CEUR Workshop Proceedings*, págs. 78–84. CEUR-WS.org, 2019. URL http://ceur-ws.org/Vol-2421/eHealth-KD_paper_8.pdf.

[37] Salvador Medina y Jordi Turmo. TALP at ehealth-kd challenge 2020. En *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020) co-*

located with 36th Conference of the Spanish Society for Natural Language Processing (SEPLN 2020), Málaga, Spain, September 23th, 2020, tomo 2664 de *CEUR Workshop Proceedings*, págs. 85–93. CEUR-WS.org, 2020. URL http://ceur-ws.org/Vol-2664/eHealth-KD_paper1.pdf.

[38] Azure Microsoft. ¿qué es el aprendizaje automático? 2022. URL https://azure.microsoft.com/es-mx/overview/what-is-machine-learning-platform/#uses.

[39] Tomás Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, y Armand Joulin. Advances in pre-training distributed word representations. *CoRR*, abs/1712.09405, 2017. URL http://arxiv.org/abs/1712.09405.

[40] Fernández Pérez Milagros. *Las categorías gramaticales (morfológicas) del español*. Universidade de Santiago de Compostela. Servizo de Publicacións e Intercambio Científico, 1993. ISBN 84-7191-969-9. URL http://hdl.handle.net/10347/12094.

[41] Loraine Monteagudo-García, Amanda Marrero-Santos, Manuel Santiago Fernández-Arias, y Hian Cañizares-Diaz. UH-MMM at ehealth-kd challenge 2021. En *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2021) co-located with the Conference of the Spanish Society for Natural Language Processing (SEPLN 2021), XXXVII International Conference of the Spanish Society for Natural Language Processing., Málaga, Spain, September, 2021*, tomo 2943 de *CEUR Workshop Proceedings*, págs. 692–702. CEUR-WS.org, 2021. URL http://ceur-ws.org/Vol-2943/ehealth_paper4.pdf.

[42] Isabel Moreno, Ester Boldrini, Paloma Moreda, y María Teresa Romá-Ferri. Drugsemantics: A corpus for named entity recognition in spanish summaries of product characteristics. *Journal of biomedical informatics*, 72:8–22, 2017.

[43] Oracle. ¿qué es el aprendizaje automático? 2022. URL https://www.oracle.com/mx/data-science/machine-learning/what-is-machine-learning/.

[44] Oracle. ¿qué es la inteligencia artificial—ia? 2022. URL https://www.oracle.com/mx/artificial-intelligence/what-is-ai/.

[45] Aitor García Pablos, Naiara Pérez, Montse Cuadros, y Elena Zotova. Vicomtech at ehealth-kd challenge 2020. En *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020) co-located with 36th Conference of the Spanish Society for Natural Language Processing (SEPLN 2020), Málaga, Spain, September 23th, 2020*, tomo 2664 de *CEUR Workshop Proceedings*, págs. 102–111. CEUR-WS.org, 2020. URL http://ceur-ws.org/Vol-2664/eHealth-KD_paper3.pdf.

[46] Alejandro Rodríguez Pérez, Ernesto Quevedo Caballero, Jorge Mederos Alvarado, Rocío Cruz Linares, y Juan Pablo Consuegra-Ayala. UH-MAJA-KD at ehealth-kd challenge 2020. En *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020) co-located with 36th Conference of the Spanish Society for Natural Language Processing (SEPLN 2020), Málaga, Spain, September 23th, 2020*, tomo 2664 de *CEUR Workshop Proceedings*, págs. 125–135. CEUR-WS.org, 2020. URL http://ceur-ws.org/Vol-2664/eHealth-KD_paper5.pdf.

[47] Slav Petrov, Dipanjan Das, y Ryan T. McDonald. A universal part-of-speech tagset. *CoRR*, abs/1104.2086, 2011. doi:10.48550/ARXIV.1104.2086. URL http://arxiv.org/abs/1104.2086.

[48] Alejandro Piad-Morffis, Yoan Gutiérrez, Hian Cañizares-Diaz, Suilan Estevez-Velarde, Rafael Muñoz, Andrés Montoyo, y Yudivián Almeida-Cruz. Overview ehealth-kd 2020. En *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020) co-located with 36th Conference of the Spanish Society for Natural Language Processing (SEPLN 2020), Málaga, Spain, September 23th, 2020*, tomo 2664 de *CEUR Workshop Proceedings*, págs. 71–84. CEUR-WS.org, 2020. URL http://ceur-ws.org/Vol-2664/eHealth-KD_overview.pdf.

[49] Alejandro Piad-Morffis, Yoan Gutiérrez, Juan Pablo Consuegra-Ayala, Suilan Estevez-Velarde, Yudivián Almeida-Cruz, Rafael Muñoz, y Andrés Montoyo. Overview of the ehealth knowledge discovery challenge at iberlef 2019. En *Proceedings of the Iberian Languages Evaluation Forum co-located with 35th Conference of the Spanish Society for Natural Language Processing, IberLEF@SEPLN 2019, Bilbao, Spain, September 24th, 2019*, tomo 2421 de *CEUR Workshop*

Proceedings, págs. 1–16. CEUR-WS.org, 2019. URL http://ceur-ws.org/Vol-2421/eHealth-KD_overview.pdf.

[50] Alejandro Piad-Morfis, Suilan Estevez-Velarde, Yoan Gutiérrez, Yudivián Almeida-Cruz, Andrés Montoyo, y Rafael Muñoz. Overview of the ehealth knowledge discovery challenge at iberlef 2021. *Proces. del Leng. Natural*, 67:233–242, 2021. URL http://journal.sepln.org/sepln/ojs/ojs/index.php/pln/article/view/6392.

[51] Lasserre Chavez Hugo Raziel. *Extracción automática de relaciones taxonómicas de tipo hipónimo / hiperónimo en corpus de dominio*. Tesis de maestría, Benemérita Universidad Autónoma de Puebla, 2018.

[52] Sergio Santana, Alicia Pérez, y Arantza Casillas. Haplap at ehealth-kd challenge 2020. En *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020) co-located with 36th Conference of the Spanish Society for Natural Language Processing (SEPLN 2020), Málaga, Spain, September 23th, 2020*, tomo 2664 de *CEUR Workshop Proceedings*, págs. 136–140. CEUR-WS.org, 2020. URL http://ceur-ws.org/Vol-2664/eHealth-KD_paper6.pdf.

[53] Xinyu Song, Ao Feng, Weikuan Wang, y Zhengjie Gao. Multidimensional self-attention for aspect term extraction and biomedical named entity recognition. *Mathematical Problems in Engineering*, 2020:8604513, 2020. ISSN 1024-123X. doi:10.1155/2020/8604513.

[54] Ann Taylor, Mitchell Marcus, y Beatrice Santorini. *The Penn Treebank: An Overview*, págs. 5–22. Springer Netherlands, Dordrecht, 2003. ISBN 978-94-010-0201-1. doi:10.1007/978-94-010-0201-1_1. URL https://doi.org/10.1007/978-94-010-0201-1_1.

[55] Yuanhe Tian, Wang Shen, Yan Song, Fei Xia, Min He, y Kenli Li. Improving biomedical named entity recognition with syntactic information. *BMC Bioinformatics*, 21(1):539, 2020. ISSN 1471-2105. doi:10.1186/s12859-020-03834-6.

[56] Unipython. El modelo embeddings (incrustaciones)

de palabras. 2018. URL https://unipython.com/el-modelo-embeddings-incrustaciones-de-palabras/.

[57] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, y Illia Polosukhin. Attention is all you need. En *Advances in Neural Information Processing Systems*, tomo 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[58] Yonghui Wu, Min Jiang, Jun Xu, Degui Zhi, y Hua Xu. Clinical named entity recognition using deep learning models. *AMIA Annual Symposium proceedings. AMIA Symposium*, 2017:1812–1819, 2017.

[59] Huiwei Zhou, Zhe Liu, Chengkun Lang, Yibin Xu, Yingyu Lin, y Junjie Hou. Improving the recall of biomedical named entity recognition with label re-correction and knowledge distillation. *BMC Bioinformatics*, 22(1):295, 2021. ISSN 1471-2105. doi:10.1186/s12859-021-04200-w.