AFI Escuela de Finanzas, Madrid
Máster en Data Science y Big Data

Modelos de Aprendizaje Supervisado aplicados a Sistemas Q/A en Idioma Español

Claudia Quintana Wong

Tutor: **Dr. Francisco J. Rodríguez Aragón**

Memoria escrita
presentada en opción al título de
Máster en Data Science y Big Data

Septiembre de 2021

Resumen

El desarrollo de sistemas Pregunta/Respuesta es un paso crucial en el entendimiento del lenguaje natural. Este problema ha ganado gran atención desde el inicio de la lingüística computacional, como consecuencia, varios métodos han sido propuestos para resolver esta tarea en el idioma inglés, los más recientes, aplicando el enfoque de aprendizaje profundo. Sin embargo, los avances en el idioma español siguen siendo escasos. Head-QA es un conjunto de datos de preguntas de selección múltiple orientado al dominio biomédico en idioma español, y constituye la base de este proyecto. En este trabajo se proponen modelos basados en las redes profundas para la resolución del problema de selección de respuestas sobre la base del corpus Head-QA. Estos modelos no utilizan características léxicas y sintácticas como árboles de dependencia, partes de la oración y reconocedores de entidades nombradas, se basan solamente en el conjunto de datos. Se presentan, comparan y discuten los modelos matemáticos asociados y las arquitecturas respectivas. Finalmente, se presenta un conjunto de experimentos orientados a la evaluación de los modelos de aprendizaje profundo que demuestran la efectividad de la solución.

Abstract

Question/Answering systems are a crucial step in understanding natural language. This problem has gained great attention since the beginning of computational linguistics, as a consequence, several methods have been proposed to solve this task in the English language, the most recent ones applying the deep learning approach. However, advances in the Spanish language remain scarce. Head-QA is a dataset of multiple-choice questions oriented to the biomedical domain in Spanish, and forms the basis of this project. In this work, models based on deep networks are proposed for the resolution of the response selection problem based on the Head-QA corpus. These models do not use lexical and syntactic features such as dependency trees, parts of speech, and named entity recognizers, they only take into account the position of words in a sentence. The associated mathematical models and the respective architectures are presented, compared and discussed. Finally, a set of experiments aimed at evaluating deep learning models that demonstrate the effectiveness of the solution is presented.

Índice general

Índice de figuras

Índice de tablas

Introducción

Desde tiempos inmemoriales, el ser humano procesa la información con el fin de descubrir conocimiento. Con el surgimiento de la Web y el avance de las tecnologías digitales, la generación de información ha experimentado un crecimiento desmesurado. El gran volumen existente actualmente hace imposible procesar toda la información manualmente al mismo ritmo que se genera. Como resultado, existe una gran acumulación de información que está siendo desaprovechada en términos del descubrimiento de conocimiento. Por esta razón, una alternativa es recurrir al procesamiento automático de información con el fin de extraer la información relevante y organizarla de manera que pueda ser procesada por una computadora. Actualmente uno de los retos consiste en que la mayor parte del contenido presente en la Web es de naturaleza no estructurada pues se genera principalmente en forma de texto.

Dada esta sobrecarga de información y la creciente necesidad de consultar el contenido de la información generada, los sistemas pregunta/respuesta (conocidos como Q/A, del inglés *Question/Answering*) se han vuelto cada vez más importantes. Estos sistemas tienen como objetivo satisfacer a los usuarios que buscan la respuesta a una pregunta específica en lenguaje natural de manera automática.

El lenguaje humano es increíblemente complejo y diverso. Nos expresamos de disímiles maneras, verbalmente y por escrito. No solo existen cientos de idiomas y dialectos, sino que en cada idioma existe un conjunto único de reglas gramaticales y de sintaxis, términos y palabras coloquiales. Entender, interpretar y manipular el lenguaje humano no es tarea fácil para una computadora. La importancia de dar solución a este problema es un hecho que la comunidad científica no ha tardado en percibir, por lo cual muchos esfuerzos han sido dedicados a resolver esta tarea. Aunque todavía no se considera un problema resuelto, lo cierto es que se han presentado numerosas propuestas que se han superado en el tiempo, fundamentalmente en el idioma inglés.

La meta general es tomar texto del lenguaje y aplicar la lingüística y algoritmos para transformar o enriquecer el texto de tal forma que provea un mayor valor. Simulando la lógica humana, la clave consiste en localizar los aspectos importantes de la información y estructurar la información de manera que pueda ser manipulada por una computadora. Desde el punto de vista computacional, un sistema Q/A es un sistema de recuperación de información que dada una consulta en lenguaje natural debe dar una respuesta concreta que responda a dicha pregunta.

Los avances más recientes en este tema se han enfocado en el uso de modelos neuronales dada su facilidad para trabajar con texto en bruto. Los modelos neuronales han probado ser muy eficaces en el contexto del aprendizaje supervisado, por lo que se ha visto una tendencia entre los investigadores a desarrollar conjuntos de datos y métodos que se adaptan a la gran cantidad de datos y fortalezas de los métodos neuronales actuales. Sin embargo, estos sistemas son capaces de alcanzar un rendimiento muy cercano al rendimiento humano con un conocimiento muy superficial. Con el objetivo de contrarrestar este hecho, se han propuesto y desarrollado conjuntos de datos y algoritmos orientados a dar resppuesta a preguntas de selección múltiple, de manera que, se requiera un razonamiento mínimo para responder correctamente. A esta tarea, aú aún dentro del ámbito de los sistemas Q/A, se le conoce como *Answer Selection*.

El problema de *Answer Selection* consiste en dada una pregunta y un conjunto de respuestas, identificar cuál es la respuesta correcta entre las candidatas. Recientemente, varios métodos basados en aprendizaje profundo has sido propuestos para dar solución a esta tarea. Sin embargo, la escasez de conjuntos de datos principalmente en idiomas diferentes al español ha limitado el avance en esta lengua.

Vilares and Gómez-Rodríguez (2019) presentan un complejo conjunto de datos de selección múltiple en español, que requiere conocimiento y razonamiento en dominios complejos y que, incluso para los humanos con años de entrenamiento, es una tarea difícil. Aborda textos de dominios específicos como Medicina, Biología, Enfermería, entre otros. Este dataset constituye un paso de avance hacia modelos de aprendizaje robustos y confiables para construir sistemas Q/A en idioma español.

Este corpus a diferencia de otros de Q/A que se centran en el entendimiento del lenguaje, requiere un razonamiento más profundo y conocimiento sobre materias específicas. Por esta razón, las propuestas presentadas hasta el momento obtienen resultados muy inferiores a los alcanzados por humanos. Los autores presentan un conjunto de técnicas a ser utilizadas como benchmarks. Sin embargo, todas las propuestas que se presentan son no supervisadas o super-

visadas a distancia, los autores recomiendan como trabajos futuros incursionar en la aplicación de metodologías supervisadas, lo cual motiva este trabajo.

Teniendo en cuenta que el español es el segundo idioma más hablado en el mundo, después del chino mandarín, surge el **problema científico** que motiva la realización de esta investigación. A pesar de los avances en el idioma inglés, en el idioma español continúa la escasez de soluciones que aborden el problema de *Answer Selection* por lo que se mantiene la imposibilidad de transformar los datos existentes en la Web y otras fuentes de información no estructuradas en conocimiento.

La **pregunta científica** que se plantea es: ¿Será posible crear un modelo matemático computacional supervisado de selección múltiple dedicado a responder preguntas en idioma español sobre la base de las concepciones y tecnologías de la inteligencia artificial y el aprendizaje automático?

El **objetivo general** consiste en desarrollar un modelo pregunta/respuesta para el idioma español fundamentado científica y tecnológicamente. Para lograr el cumplimiento del objetivo general, se plantea a continuación un conjunto de objetivos específicos:

1. Profundizar desde los puntos de vista teórico-conceptual y práctico en el área de los sistemas pregunta/respuesta, especialmente en los de selección múltiple.

2. Concebir y diseñar un algoritmo que utilice el enfoque supervisado. Diseñar e implementar modelos de aprendizaje profundo con diferentes arquitecturas de red para dar respuesta a las preguntas.

3. Evaluar cualitativa y experimentalmente los modelos propuestos y comparar los resultados de las soluciones concebidas.

El aporte principal del presente trabajo es desarrollar modelos supervisados sobre el dataset HeadQA que sirvan como benchmarks en la aplicación de modelos que sigan esta metodología.

La memoria escrita se divide en cuatro capítulos:

En el capítulo 1 “Estado del Arte”, se reseña el estado actual de la ciencia y la tecnología en los temas tratados, que han servido como base para la investigación y la obtención de los resultados.

En el capítulo 2 “Análisis y preprocesamiento”, se presenta la descripción y los detalles del análisis exploratorio del conjunto de datos. Asimismo, se detalla el preprocesamiento de los datos con el fin de prepararlos como entrada a los modelos.

En el capítulo 3 “Modelos Propuestos” se proponen diferentes modelos de aprendizaje principalmente supervisados y se describen las arquitecturas diseñadas. Asimismo, se detallan los aspectos técnicos de la implementación del prototipo, aprovechando convenientemente las herramientas existentes.

En el capítulo 4 “Resultados” se examina cualitativa y experimentalmente la validez de la solución implementada. Se comparan los resultados obtenidos por los modelos de aprendizaje implementados sobre el corpus.

Como parte del desenlace, se presentan las conclusiones, que recogen los principales resultados obtenidos en el desarrollo de la investigación en función del cumplimiento de los objetivos, así como el trabajo futuro, que propone un conjunto de ideas a ser exploradas en el futuro como parte de la continuación del actual trabajo.

Para concluir se lista la bibliografía utilizada para sustentar la base científica de la solución propuesta y facilitar la búsqueda de temas relacionados.

Capítulo 1

Estado del Arte

En el presente capítulo se discuten brevemente aquellos conceptos que constituyen el fundamento teórico para el desarrollo de este trabajo.

1.1. Sistemas Pregunta/Respuesta

Los sistemas Preguntas/Respuestas han sido uno de los problemas planteados por los humanos como primer paso para el entendimiento del lenguaje natural, de ahí que los diferentes enfoques para intentar resolver este tipo de problemas se remonten al siglo anterior.

Los primeros enfoques se caracterizaban por el uso de reglas predefinidas, estas soluciones aunque logran dar una respuesta muy exacta a determinadas preguntas en dominios específicos, no escalan bien cuando se trata de textos con formatos y temas variados. Posteriormente, los avances se centran en el uso de *feature engineering*, herramientas linguísticas o recursos de la lengua externos como WordNet. Yih et al. (2013), Yao et al. (2013) y Heilman and Smith (2010) son algunos de los exponentes de esta tendencia. Aunque estos métodos muestran efectividad, pueden verse afectados por la disponibilidad de recursos adicionales, el esfuerzo de la ingeniería de características y la complejidad sistemática al introducir herramientas lingüísticas, como árboles de análisis y árboles de dependencia.

Los avances más recientes en Q/A, y en el área de procesamiento del lenguaje natural en general, han sido protagonizados por modelos basados en redes neuronales. Muchas propuestas han sido enfocadas desde el aprendizaje supervisado, de ahí que existan varios conjuntos de datos como son bAbI y SQuAD. En algunos de estos datasets los sistemas artificiales han

llegado a obtener un rendimiento cercano a los resultados alcanzados por humanos.

Sin embargo, estos modelos se adaptan a los datasets de manera que los sistemas pueden lograr alcanzar buenos resultados con un conocimiento muy superficial. Los sistemas QA de múltiples opciones surgen para contrarrestar este efecto y son conocidos en la literatura como sistemas de selección de respuestas (en inglés, sistemas *Answer Selection*).

Adicionalmente, la mayor parte de estos datasets antes mencionados están enfocados en el idioma inglés y en dominios del conocimiento generales. Esto ha ocasionado que la mayoría de los avances en el área del descubrimiento de conocimiento a partir de texto en lenguaje natural se hayan concentrado en este idioma, provocando incluso que autores de origen hispano enfoquen sus trabajos y aportes en el idioma inglés. Adicionalmente y por la dificultad que conlleva la construcción de datasets de este tipo, existen muy pocos datasets sobre dominios específicos como Medicina, Biología, entre otros. Aunque existen trabajos en estos dominios como Abacha et al. (2015) y Nentidis et al. (2018), los autores Vilares and Gómez-Rodríguez (2019) hallan en esta problemática la motivación para la presentación de Head QA, un dataset Q/A específicamente de selección de respuestas en español e inglés, que combina la necesidad de conocimiento y razonamiento en dominios complejos y que resulta difícil responder correctamente, incluso para humanos con años de entrenamiento.

Dado que los mejores resultados en esta área han sido alcanzados mediante la aplicación de modelos de aprendizaje profundo, este trabajo se concentrará en el desarrollo e implementación de modelos neuronales aplicados al conjunto de datos Head QA. Por esta razón, en la siguiente sección serán abordados los avances más recientes en el aprendizaje profundo aplicado al procesamiento del lenguaje natural con el fin de sentar las bases de los modelos a proponer.

1.2. Recuperación de Información

En Vilares and Gómez-Rodríguez (2019), se presentan un conjunto de modelos que pueden ser utilizados como *baselines* para el conjunto de datos en inglés y español. Todos los *baselines* siguen el paradigma de recuperación de información clásica utilizando Wikipedia como fuente de información externa.

Un sistema de recuperación de información clásico tiene como base un corpus compuesto por un conjunto de documentos D. El objetivo es a partir de una consulta q expresada en lenguaje natural, obtener un ordenamiento de los documentos de acuerdo a su relevancia o nivel de similaridad con dicha consulta. De manera que, los primeros lugares del *ranking* correspon-

dan a los documentos más relevantes a la consulta.

Desde el punto de vista de recuperación de información, un sistema Q/A de selección puede ser planteado como:

Sean $(q_i, [a_{i0}, a_{i1}, ..., a_{im}])$ una pregunta y las posibles respuestas. Se crean m consultas de la forma $q_i + a_{im}$ que son enviadas al motor de búsqueda de manera independiente. El motor de búsqueda devuelve para cada consulta enviada un número que indica la mayor relevancia obtenida, es decir, el *score* del documento que mayor puntuación alcanzó en cada caso. Luego, la respuesta correcta a_{im} es seleccionada en base a la consulta $q_i + aim$ que mayor probabilidad alcanzó, lo que indica que es la respuesta donde se encontró el documento con mayor relevancia de todo el corpus.

En Vilares and Gómez-Rodríguez (2019) para el conjunto de datos en español se utilizó como motor de búsqueda el presentado en Chen et al. (2017), el cual calcula la similitud entre la consulta y los documentos representados vectorialmente con el tf_idf como la suma ponderada de los vectores palabra, además también tiene en cuenta el orden y el conteo de bigramas.

Otro de los trabajos enfocados en este conjunto de datos específico se presenta en Liu et al. (2020). Los autores presentan un sistema multipasos basado en un framework de extraccción de conocimiento llamado MurKe. Inicialmente, extrae información de un gran corpus conformado por documentos de salud. Para encontrar la cadena de razonamiento y elegir la respuesta correcta, MurKe itera entre seleccionar los documentos de respaldo, reformular la representación de la consulta utilizando los documentos de respaldo y obtener la puntuación de vinculación para cada elección utilizando el modelo de vinculación. El módulo de reformulación aprovecha documentos seleccionados para evidencia faltante, lo que mantiene la interpretabilidad. Además, hacen un uso completo de los modelos pre-entrenados listos para usar. Con menos peso entrenable, el modelo previamente entrenado puede adaptarse fácilmente a las tareas de atención médica con ejemplos de entrenamiento limitados. A partir de los resultados experimentales, este sistema es capaz de superar varias líneas de base sólidas en el conjunto de datos HeadQA.

En el dataset no ha sido aplicado ninguna técnica de selección de respuestas sobre la base del aprendizaje supervisado y sin el uso de fuentes de conocimiento externas, por lo que este trabajo se enfocará en la aplicación de modelos supervisados. Dada la dificultad de la tarea en el ámbito del entendimiento del lenguaje humano y teniendo el cuenta el estado del arte en procesamiento del lenguaje natural, los modelos a implementar estarán orientados al uso de técnicas de aprendizaje profundo. Es importante destacar que la naturaleza de este tipo de pro-

blemas conlleva a que modelos que utilizan conocimiento externo alcancen mejores resultados, pues los modelos supervisados encuentran muy difícil adaptarse a la complejidad y especificidad de las preguntas, sobre todo cuando no se cuenta con un conjunto de datos suficientemente extenso.

1.3. Aprendizaje Profundo en NLP

En la presente sección se introduce brevemente el concepto de *word embedding*, el cual constituye un enfoque clave en gran parte de las propuestas contemporáneas. De igual manera se presenta una revisión de los principales resultados logrados en los sistemas Q/A utilizando el enfoque del aprendizaje profundo.

Uno de los principales retos en los problemas de procesamiento del lenguaje natural es encontrar una representación vectorial de un texto que sea lo suficientemente rica y expresiva. En otras palabras, una forma matemática de representar la información relevante y el conocimiento contenidos en el lenguaje humano. Intuitivamente, mientras más rica sea la representación, más información estarán utilizando los modelos de aprendizaje y estarán más cerca de lograr buenos resultados en el entendimiento del lenguaje natural.

Un documento textual es representado matemáticamente por un vector numérico. El primer acercamiento en este sentido fue la representación *one-hot*, donde a cada *token*[1] se le asocia un índice y es representado por un vector con tantas componentes como *tokens* tiene el vocabulario (Goyal et al. (2018)). De esta manera es posible hacerle corresponder a cada palabra una componente del vector, el cual tiene valor 0 en todas las componentes, excepto en la posición correspondiente a la palabra que representa, donde toma valor 1. La estrategia tradicional empleada sigue el mismo principio con la diferencia de que la componente de la palabra activa en lugar de 1, toma un valor mayor que puede ser la cantidad de veces que aparece la palabra en el documento. Otra variante muy utilizada es utilizar el resultado de la expresión conocida como $tf^2 - idf^3$. Ambas representaciones requieren vectores de gran dimensión y son insuficientes para capturar la semántica de las palabras.

Como alternativa, se incorporan otras características del lenguaje como anotaciones de las

[1]En los idiomas inglés y español, en el área de procesamiento del lenguaje natural, generalmente se utiliza el término *token* para referirse a una palabra. En la presente tesis se seguirá este convenio.

[2]del inglés, *term frequency*

[3]del inglés, *inverse document frequency*

partes de la oración (POS, del inglés *part-of-speech*), funciones gramaticales y análisis sintáctico. Según establece Chollet (2017), estas herramientas no son exactas y muchas veces cometen errores que afectan el rendimiento de la tarea principal. Además, al basarse solamente en elementos sintácticos de una oración tampoco se logra atrapar toda la riqueza semántica de las palabras.

Los *word embeddings*, introducidos por Mikolov et al. (2013), son una forma diferente de representación. Un *word embedding* es una forma de representación distribuida de las palabras de un vocabulario en un vector. Cada palabra es representada por un vector de pequeñas dimensiones respecto al tamaño del vocabulario. La gran novedad de esta representación es su capacidad para capturar el significado de una palabra teniendo en cuenta el contexto en que aparece, captando tanto rasgos sintácticos como semánticos que permiten establecer similitud entre las palabras.

La representación embebida de una palabra, como también se puede llamar en español, es aprendida en tareas auxiliares enfocadas en predecir la palabra que mejor se ajusta al contexto, entrenadas sobre grandes cantidades de texto. A estas tareas auxiliares se les conoce como *language modeling* y a los modelos resultantes se les llama en la literatura *language models* (Rao and McMahan (2019)). Word2vec (Mikolov et al. (2013)), GloVe (Pennington et al. (2014)) y FastText (Bojanowski et al. (2016)) son algunos de los *frameworks* que brindan técnicas para el aprendizaje de los *word embeddings* y, además, facilitan el uso de *embeddings* pre-entrenados. Al enfoque de utilizar *embeddings* preentrenados en una tarea auxiliar en la solución de un problema diferente se le llama *fine tuning* (Rao and McMahan (2019)), también es conocido en la literatura como *transfer learning*.

Otro enfoque que ha tenido gran aceptación en el área del procesamiento del lenguaje es el aprendizaje profundo por el poder de aprendizaje y generalización que han demostrado estos modelos en diferentes áreas del conocimiento. Los primeros trabajos en utilizar el enfoque de aprendizaje profundo tratan el problema de selección de respuestas como un problema de clasificación binario, intentando asignar una clase a cada par (pregunta, respuesta).

Dentro del aprendizaje profundo, las Redes Neuronales Convolucionales (CNN, del inglés *Convolutional Neural Networks*) y las Redes Recurrentes (RNN, del inglés *Recurrent Neural Networks*) son ampliamente utilizadas para hallar representaciones vectoriales ricas semánticamente de secuencias textuales. Ambas son tipos especiales de redes neuronales, con diferencias en sus arquitecturas que las hacen más adecuadas para un dominio u otro.

Intuitivamente, las Redes Recurrentes se adaptan mejor que otras arquitecturas de redes

neuronales a las tareas de procesamiento del lenguaje natural porque son capaces de aceptar como entrada secuencias de tamaño variable (característica inherente de las oraciones) y analizar cada uno de los *tokens* secuencialmente. Al analizar una oración *token* por *token*, al final de la secuencia logran captar información de cada palabra vista y de esa manera, obtienen una representación de toda la oración. Sin embargo, en la práctica, cuando las secuencias son muy largas se ha demostrado que la representación final pierde información de las primeras palabras analizadas (Rao and McMahan (2019)). Con el fin de superar este problema surgen las redes LSTM (del inglés, *Long-Short Term Memory*) las cuales tienen una arquitectura más compleja que permite atrapar dependencias a lo largo de la secuencia.

Vaswani et al. (2017) propone una nueva arquitectura de redes que recibe el nombre de *Transformer* basada únicamente en mecanismos de atención. Aunque la arquitectura fue probada por primera vez en la tarea de traducción automática, alcanzando resultados superiores en calidad, paralelización y tiempo de entrenamiento, los autores afirman que la nueva arquitectura *Transformer* generaliza bien a otras tareas y con datos de entrenamiento diferentes. Aprovechando la nueva arquitectura propuesta por Vaswani et al. (2017), Devlin et al. (2018) propone el modelo BERT (del inglés, *Bidirectional Encoder Representations*) el cual aplica una arquitectura bidireccional basada en *Tranformer* para resolver la tarea de *Language Modeling* y defiende que obtienen representaciones pre-entrenadas del lenguaje mucho más ricas semánticamente que los *word embeddings* vistos anteriormente.

En la siguiente sección se analizan las propuestas existentes en la literatura que siguen el enfoque del aprendizaje profundo para resolver el problema de selección de respuestas.

1.3.1. Aprendizaje profundo en sistemas Q/A

Los avances más recientes en procesamiento del lenguaje natural, casi en su mayoría, están enfocados en el empleo del aprendizaje profundo. La razón principal es que superan las técnicas tradicionales sin depender de ningún recurso externo. Además, no necesitan ningún esfuerzo de ingeniería de funciones o recursos codificados a mano más allá de algunos grandes sin etiqueta corpus en el que aprender las incrustaciones de palabras iniciales, como los *word embeddings*. Este hecho no es ajeno a los sistemas Q/A. El aprendizaje profundo constituye una opción más acertada porque la recuperación de información tradicional se basa en la coincidencia, exacta o parcial, de palabras. Sin embargo, en el problema de selección de la respuesta correcta un enfoque de coincidencia no es suficiente, porque las respuestan guardan cierto parecido sintáctico entre sí, y la clave para diferenciar la correcta está en la comprensión y

razonamiento de la pregunta. Aunque aún no se puede afirmar que la computadora logre razonar, los modelos de aprendizaje profundo han demostrado un cierto conocimiento del idioma y el campo específico en que sean entrenados.

El problema puede ser planteado en término generales como:

Dada una pregunta q_i en el dataset y un conjunto de oraciones candidatas $c_{i1}, c_{i2}, ..., c_{im}$, la tarea es identificar oraciones candidatas que contienen la respuesta correcta a la pregunta. A partir de la definición, el problema se puede formular como un problema de *ranking*, donde el objetivo es dar una mejor posición a las oraciones candidatas que son relevantes a la pregunta.

Desde el punto de vista del aprendizaje automático, según Lai et al. (2018) existen tres tipos de enfoques para tratar el problema de selección de respuestas. A continuación, se explican brevemente cada uno de ellos.

- Pointwise: El problema de *ranking* se transforma en un problema de clasificación binario, donde las instancias de entrenamiento tienen la forma (q_i, c_{ij}, y_{ij}) y y_{ij} puede tomar valores 0 o 1 si la respuesta es correcta o no. En la etapa de inferencia, el resultado de la función h_0 toma valores en el rango $0 \leq h_0 \leq 1$, donde cada número salida puede ser interpretado como la probabilidad de que la respuesta sea correcta o no, de manera que la respuesta correcta es entre las candidatas aquella que mayor valor de h_0 tenga.

- Pairwise: La función de *ranking* se entrena explícitamente para puntuar las respuestas correctas con un mayor valor que las incorrectas. Dada una pregunta, la instancia entrenante toma dos respuestas candidatas, una correcta y la otra incorrecta, hecho que es recogido por la función de pérdida, de manera que el modelo aprende cuál de las respuestas es más relevante a la pregunta.

- Listwise: A diferencia de los métodos anteriores que ignoran que la selección se hace sobre una lista de respuestas candidatas, en este enfoque una instancia consiste en la pregunta y la lista de respuestas. La función de pérdida se adapta, de igual manera, para determinar cuál de las oraciones candidatas es la más relevante a la pregunta.

Existen tres tipos de arquitecturas de red generales para calcular la relevancia de una oración candidata a una pregunta. El matiz que las diferencia es la forma de aprender la función de *ranking* h_0. Según Lai et al. (2018), los límites entre cada una de las arquitecturas no están siempre bien definidos.

Arquitecturas siamesas

La arquitectura siamesa en esencia consiste en que el codificador o en inglés *encoder* crea por separado las representaciones vectoriales de las oraciones de entrada, en este caso, de la pregunta y la respuesta que está siendo analizada. De manera que, las oraciones no influyen en el cálculo de la representación de cada una de ellas. Luego, las oraciones codificadas se comparan a través de una medida de similutid, que puede ser el coseno como en Feng et al. (2015), cualquier operación elemento a elemento Tai et al. (2015) o combinaciones basadas en redes neuronales como proponen Bowman et al. (2015).

Uno de los primeros modelos de aprendizaje profundo aplicado a la selección de respuestas fue el propuesto por Yu et al. (2014). En esta primera propuesta el modelo genera la representación vectorial de cada secuencia tomando la media de todos los vectores palabras de la oración, además integra características adicionales como la cantidad de palabras que se superponen en ambas secuencias. Este modelo logró un mejor rendimiento que los modelos tradicionales basados en la extracción e ingeniería de características lingüísticas.

Una arquitectura innovadora fue la presentada por Tan et al. (2015) con el nombre de QA-LSTM, la cual utiliza una red bidireccional de tipo LSTM (del inglés, *Long Short-Term Memory*), que constituye un tipo especial de red neuronal recurrente, y una capa *pooling* para obtener una representación vectorial distribuida de las secuencias de entrada de manera independiente. Posteriormente, el modelo utiliza la similitud del coseno para computar la distancia entre las representaciones.

Un ejemplo del uso de las redes convolucionales para generar una representación de las oraciones de entrada es el publicado por Severyn and Moschitti (2015), donde las representaciones se obtienen tras aplicar diferentes tipos de *pooling* en varios niveles de granularidad de la red. La comparación entre las secuencias se realiza utilizando varias métricas de similitud también en diferentes niveles de granularidad de la red. Posteriormente, los resultados de las métricas son utilizados como entrada a una capa *fully connected* para obtener la relevancia final.

Arquitectura basada en mecanismos de atención

En una arquitectura siamesa, las oraciones de entrada se codifican en representaciones vectoriales de longitud fija e independientes y luego, se comparan. A pesar de su sencillez conceptual, una desventaja es la ausencia de interacción explícita entre las oraciones de entrada durante el proceso de codificación. A una pregunta siempre se le asigna al mismo vector inde-

pendientemente de la respuesta candidata en consideración, y viceversa.

Tan et al. (2015) enriquece el modelo QA-LSTM presentado con un mecanismo de atención. Conceptualmente, un mecanismo de atención da mayor importancia a ciertas palabras en la respuesta candidata, y los pesos se calculan de acuerdo con la pregunta. En este caso, el mecanismo de atención se aplica solo en una única dirección, pues solo se emplea para codificar la respuesta candidata en función de la pregunta. Por tra parte, dos Santos et al. (2016) emplea un mecanismo de atención en doble sentido, de manera que tanto la pregunta como la respuesta candidata influyen en la representación de la otra.

Arquitectura *Compare-Aggregate*

La idea de esta arquitectura es comparar unidades de la red más pequeñas que las oraciones como comúnmente se hace. Se comparan unidades como palabras o incluso letras de las oraciones de entrada y luego los resultados de la comparación se agregan mediante redes recurrentes o convolucionales para tomar una decisión definitiva, como se presenta en Tran et al. (2018).

De manera general un modelo *Compare-Aggregate* consta de 5 capas, como se detalla en Wang et al. (2017). La primera capa se encarga de la representación de las palabras (*word representation layer*), su fin es representar cada palabra en un vector d-dimensional. El objetivo de la capa de representación del contexto (*context representation layer*) es obtener una nueva representación para cada posición en las oraciones de entrada que capture alguna información contextual además de la posición de la palabra. La capa de comparación (*matching layer*) compara cada representación contextual de una oración con todas las representaciones contextuales de la otra oración. La salida de esta capa son dos secuencias de vectores coincidentes, donde cada vector coincidente corresponde al resultado de la comparación de una posición de una oración con todas las posiciones de la otra oración. Posteriormente, la capa de agregación (*aggregation layer*) se encarga de agregar los resultados de la comparación de la capa anterior. Por último, la capa de predicción (*prediction layer*) encargada de obtener el valor final que indica la relevancia de la respuesta para la pregunta dada.

Las arquitecturas *Compare-Aggregate* pueden capturar más características interactivas entre las oraciones de entrada que las arquitecturas siamesas y las arquitecturas *Attentive*, por lo tanto, por lo general, tienen un mejor rendimiento como se ha demostrado cuando se evalúan en conjuntos de datos públicos como TrecQA.

Aunque el uso del modelo de lenguaje pre-entrenado BERT ha sido ampliamente utilizado

en varias tareas de procesamiento del lenguaje natural, tras esta revisión bibliográfica, se ha podido constatar que su aplicación no abunda cuando se trata de sistemas Q/A de selección de respuestas, por lo que se considera interesante incluirlo en algunos de los modelos a desarrollar en el presente trabajo.

Tras analizar las principales propuestas en la tarea de selección de respuestas se puede concluir que es un tema muy actual y una tarea todavía no resuelta. Las aplicaciones son indiscutibles y sin duda alguna, se hace necesario transferir los resultados obtenidos en el idioma inglés al idioma español.

1.4. Consideraciones generales

La selección de respuestas es un problema importante en el procesamiento del lenguaje natural y muchos métodos de aprendizaje profundo han sido propuestos para la tarea. En este capítulo, se ha brindado una revisión integral y sistemática de varios métodos de aprendizaje profundo para la selección de respuestas en dos dimensiones: enfoques de aprendizaje (*pointwise*, *pairwise* y *listwise*) y arquitecturas de redes neuronales (arquitectura siamesa, arquitectura basada en atención, y arquitectura *Compare-Aggregate*).

El escaso avance en el idioma español es perceptible. La mayoría de los esfuerzos se han concentrado en sistemas Q/A para la lengua inglesa. Aunque se carece de antecedentes directos que puedan ser utilizados como punto de partida, los avances en el idioma inglés pueden ser aprovechados teniendo en cuenta las diferencias de complejidad gramatical de ambos idiomas.

A pesar de que el trabajo está orientado al idioma español, la presente propuesta puede ser generalizada a cualquier idioma pues no se hace uso de recursos propios de la gramática española.

Capítulo 2

Análisis y preprocesamiento

Este capítulo está dedicada a detallar el análisis exploratorio sobre el conjunto de datos HeadQA y el preprocesamiento requerido. El capítulo está organizado en dos partes principales. La primera sección está dedicada a la planificación del proyecto presentando un esquema de pasos y tareas que fueron llevados a cabo en la realización de este trabajo, y que engloba desde el análisis del problema inicial hasta la presentación de los resultados.

Las siguientes secciones están dedicadas al análisis descriptivo, transformación y modelización de los datos a utilizar con el objetivo de prepararlos para que sean una entrada válida a los modelos de aprendizaje que se presentan más adelante.

2.1. Descripción CRISP-DM

Esta sección está dedicada a la presentación de las diferentes fases del proyecto. Para la organización de este trabajo se sigue la metodología CRISP-DM (del inglés, *Cross Industry Standard Process for Data Mining*), seguida con el objetivo de organizar el proceso de desarrollo. A continuación, se detalla a grandes rasgos el ciclo de vida del proyecto y las diferentes fases.

El proyecto consta de las siguientes fases:

- Comprensión del negocio
- Comprensión de los datos
- Preparación de los datos

- Modelado
- Evaluación

A continuación se describen brevemente las diferentes fases del proyecto, de igual manera en la Figura 2.1 se muestran los pasos de manera gráfica y se detallan las actividades desarrolladas en cada caso.

Comprensión del Negocio: En esta fase inicial se define y entiende el problema en cuestión, planteado en la Introducción del trabajo. Asimismo, se trazan el objetivo general y los objetivos específicos para dar respuesta a la pregunta científica. Se realiza un estudio minucioso del estado del arte de los sistemas Q/A que permite no solo la comprensión del problema sino las principales formas de solución.

Comprensión de los Datos: La fase de entendimiento de datos comienza con la colección de datos inicial y continúa con las actividades que permiten familiarizarse con los datos, identificar los problemas de calidad y descubrir conocimiento preliminar sobre los datos a través de un análisis descriptivo.

Preparación de los Datos: La fase de preparación de datos cubre todas las actividades necesarias para construir el conjunto final a partir de los datos en bruto iniciales. Las tareas incluyen la transformación, la limpieza y la vectorización de datos para la modelación.

Modelado: En esta fase, se seleccionan y aplican las técnicas de modelado que sean pertinentes al problema en cuestión. Se diseñan e implementan las diferentes arquitecturas de red y algoritmos de aprendizaje automático.

Evaluación: Finalmente, en esta etapa del proyecto se evalúan los modelos anteriormente implementados y se comparan entre sí. Se analizan los resultados obtenidos cuantitativamente y también cualitativamente teniendo en cuenta su alineación con los objetivos inicialmente propuestos.

En la Figura 2.1 se pueden visualizar cada una de las fases de desarrollo del proyecto, incluyendo las actividades específicas que se desarrollaron en cada etapa y la cantidad de horas dedicadas. A las fases anteriormente señaladas se añade la escritura de la memoria, haciendo un total de 100 horas.

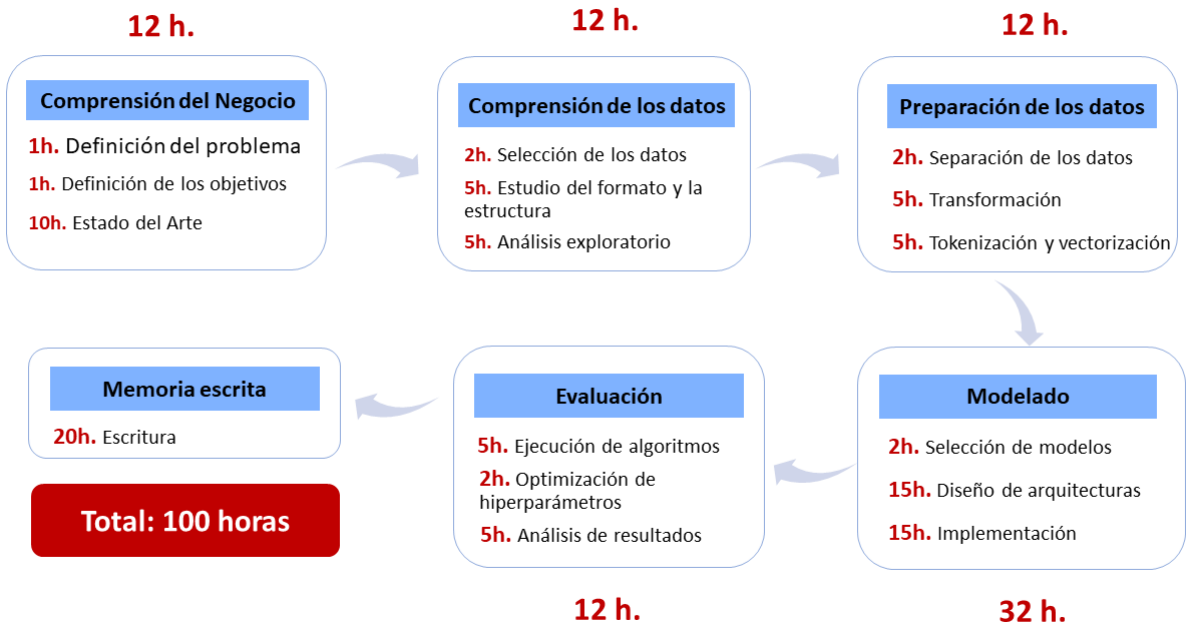

Figura 2.1: Etapas y tiempos de desarrollo del proyecto

2.1.1. Descripción del conjunto de datos

El conjunto de datos Head-QA presentado por Vilares and Gómez-Rodríguez (2019) y orientado al dominio biomédico, es un conjunto de preguntas y respuestas de múltiples opciones. El conjunto se crea a partir de los exámenes confeccionados por el Ministerio de Sanidad, Consumo y Bienestar Social de España [1] y realizados cada año para acceder a distintas especialidades dentro del sistema de salud. Más información sobre la confección del *dataset* puede ser encontrada en Vilares and Gómez-Rodríguez (2019).

Cada instancia del conjunto de datos está compuesta en esencia por una pregunta y las posibles respuestas, de las cuales solo una es correcta. Las preguntas y respuestas están organizadas en las categorías: Medicina, Enfermería, Biología, Química, Psicología, y Farmacología, correspondientes a los diferentes exámenes que se realizan. Aunque todas las propuestas publicadas hasta el momento se han construido sobre el paradigma no supervisado, el conjunto de datos está dividido en *train*, *development* y *test*.

[1] https://www.mscbs.gob.es

```
{'answers': [{'aid': 1, 'atext': 'Son de tipo todo o nada.'},
  {'aid': 2, 'atext': 'Son hiperpolarizantes.'},
  {'aid': 3, 'atext': 'Se pueden sumar.'},
  {'aid': 4, 'atext': 'Se propagan a largas distancias.'},
  {'aid': 5, 'atext': 'Presentan un periodo refractario.'}],
 'category': 'biology',
 'image': '',
 'name': 'Cuaderno_2013_1_B',
 'qid': 1,
 'qtext': 'Los potenciales postsinápticos excitadores:',
 'ra': 3,
 'year': '2013'}
```


Figura 2.2: Ejemplo de instancia del conjunto de datos

En la Figura 2.2 se muestra un ejemplo de pregunta/respuesta del dataset.
Cada instancia del conjunto está en formato JSON y contiene los siguientes atributos:

- name: nombre del examen
- qid: identificador de la pregunta
- qtext: texto de la pregunta
- category: categoría del examen (Medicina, Enfermería, Biología, ...)
- year: año del examen
- answers: lista de posibles respuestas, conformada por su número y el texto
- ra: número de la respuesta correcta
- image: camino a la imagen si lo hay

Aunque en el idioma inglés se han realizado avances en la selección de respuestas y existen varios conjuntos de datos que permiten la evaluación de los modelos propuestos, incluso en ese idioma, los *datasets* de dominios específicos y tan complejos como el actual son escasos.

Tabla 2.1: Distribución de las preguntas en HeadQA

Categoría	**Distribución total**	**Train**	**Dev**	**Test**
Biología	1.132	452	226	454
Enfermería	1.069	384	230	455
Farmacología	1.139	457	225	457
Medicina	1.149	455	231	463
Psicología	1.134	453	226	455
Química	1.142	456	228	458
Total	**6.765**	**2.657**	**1.366**	**2.742**

2.2. Análisis descriptivo

En esta sección se presenta un análisis exploratorio sobre el conjunto de dato sobre el cual se construirán los modelos. Aunque los algoritmos diseñados sobre estos datos hasta el momento han seguido el enfoque no supervisado y/o supervisado a distancia, los autores han presentado una partición con el fin de que pueda ser utilizado por algoritmos supervisados.

En total, Head-QA cuenta con 6.765 preguntas con sus respuestas posibles, organizadas en 6 categorías según la temática. La Tabla 2.2 muestra la distribución de las instancias teniendo en cuenta las categorías y los subconjuntos de datos.

Asimismo, la Figura 2.3 muestra la distribución de preguntas por categorías en los subconjuntos *train*, *dev* y *test* de manera gráfica.

Se puede apreciar que las instancias están distribuidas de manera uniforme entre todas las categorías. Aunque el conjunto de datos está balanceado en cuanto a la cantidad de instancias por categoría, existen diferencias entre las muestras en las categorías como se muestra más adelante en cuanto a las palabras más comunes empleadas en los diferentes dominios del conocimiento.

La Figura 2.4 muestra la frecuencia de las palabras más comunes en el conjunto de entrenamiento. Mientras que que la Figura 2.5 muestra la misma información mediante una nube de palabras por cada una de las categorías que definen el dominio de cada pregunta.

La nube de palabras es un tipo de representación comúnmente utilizada cuando los datos son textuales y muestra las palabras que más se repiten en el dataset en mayor tamaño. De manera, que las palabras que destacan pueden ser interpretadas como las más importantes del

dataset. Es importante destacar que para este análisis se eliminaron los *stopwords* con el objetivo de centrarnos en las palabras propias del contexto.

Como se puede observar, las palabras más comunes difieren en gran medida de categoría a categoría. Esto sugiere que puede ser recomendable construir modelos independientes para cada uno de los dominios. Además, tras analizar la frecuencia de las palabras en el conjunto general se puede notar que existe una gran coincidencia entre las palabras más comunes en todo el conjunto de datos con las palabras de mayor mención en el subconjunto correspondiente a la temática Medicina, lo que sugiere que las preguntas de Medicina utilizan un vocabulario muy repetitivo o que estos términos son también empleados en las preguntas del resto de materias.

La Tabla 2.2 muestra la máxima y media cantidad de palabras que tienen las preguntas y las respuestas respectivamente.

Este análisis denota que las preguntas suelen tener una longitud media mayor que las respuestas. La longitud media varía en dependencia de la categoría, siendo los textos de Medicina los de mayor tamaño. Este hecho puede ser aprovechado en la construcción de los modelos independientes.

Tras un análisis exploratorio sobre el conjunto de datos es pertinente recurrir al procesamiento con el fin de convertir los datos textuales en entradas válidas y comprensibles para los modelos que se presentarán en el capítulo siguiente.

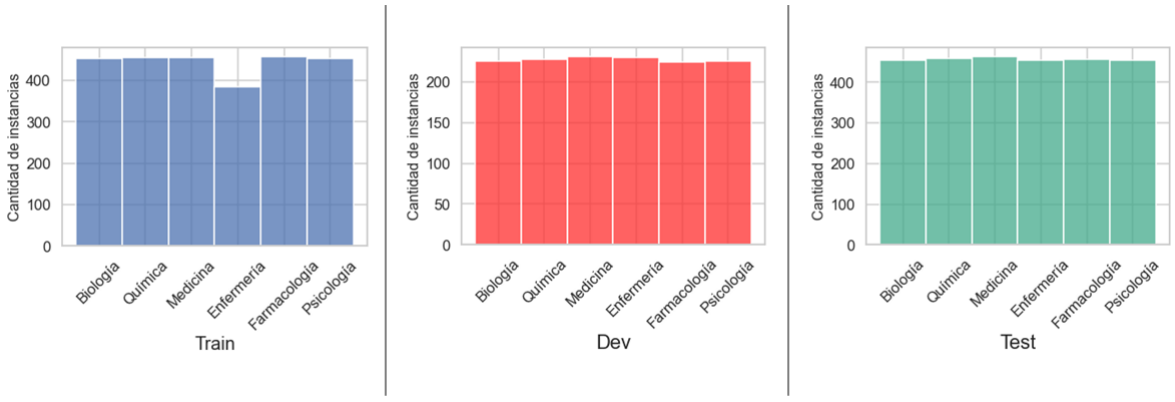

Figura 2.3: Distribución de instancias por categoría

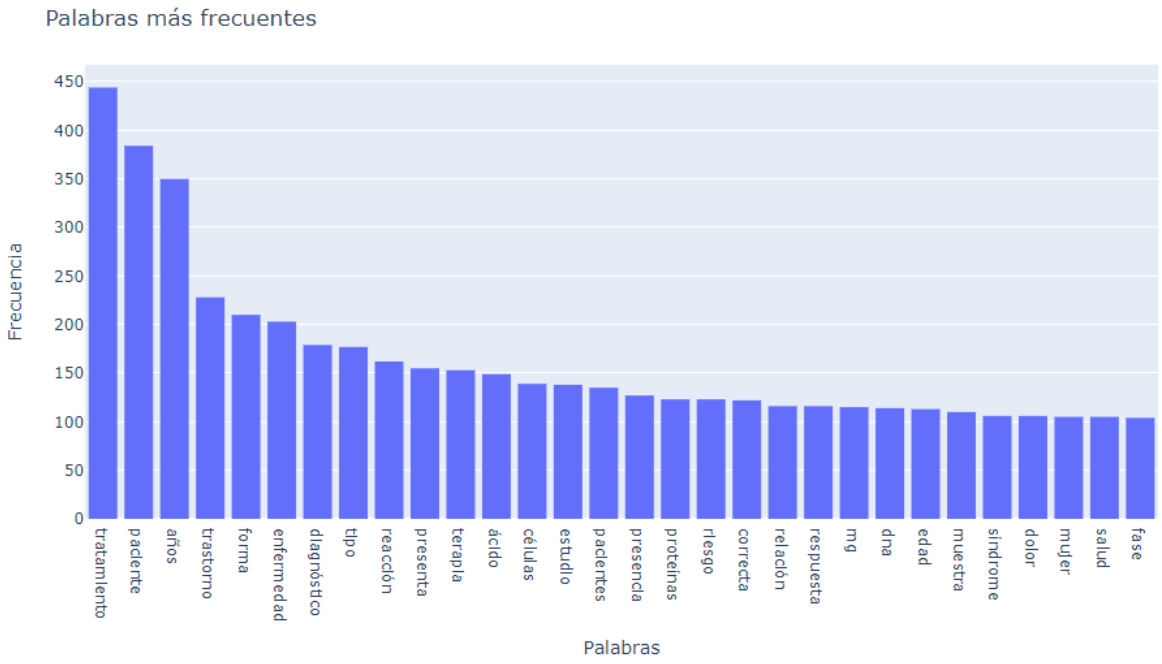

Figura 2.4: Palabras más frecuentes

Tabla 2.2: Tamaño de las preguntas y respuestas

Categoría	**Max. pregunta**	**Avg. pregunta**	**Max. respuesta**	**Avg. respuesta**
Biología	43	11	40	5
Enfermería	187	29	94	9
Farmacología	104	18	43	6
Medicina	308	55	85	9
Psicología	103	21	43	7
Química	63	15	52	7

2.3. Preprocesamiento

Como se plantea en el capítulo anterior, una instancia del corpus en esencia está conformada por la oración correspondiente a la pregunta, las correspondientes a las posibles respuestas y

Figura 2.5: Nube de palabras por categoría

la respuesta correcta. Con el propósito de utilizar el corpus antes descrito como entrada a los modelos matemáticos que se presentan es necesario transformar cada una de las instancias en un vector numérico. En este caso, es necesario transformar las oraciones en una secuencia de *tokens*.

En el idioma español, un *token* constituye una cadena de caracteres consecutivos entre dos espacios, o entre un espacio y un signo de puntuación. Todos los signos constituyen *tokens* excepto las comillas alrededor de una palabra, sin espacios intermedios, que simbolizan citas textuales. El conjunto de los *tokens* presentes en el corpus constituye el vocabulario de palabras, donde a cada *token* del vocabulario se le hace corresponder un número entero único. De esta manera, cada *token* es representado por el índice de la palabra en el vocabulario de palabras. En este caso, el vocabulario de palabras está compuesto por todas las palabras que aparecen en el conjunto de datos, tanto en preguntas como en respuestas, junto a un conjunto de *tokens* especiales.

Acorde con los modelos a desarrollar más adelante, se hace necesario emplear dos enfoques diferentes para vectorizar una instancia del conjunto de datos. El primero se ajusta a arquitecturas propias del aprendizaje supervisado, mientras que el segundo combina técnicas de recuperación de información en arquitecturas de aprendizaje supervisado. Cada uno de los en-

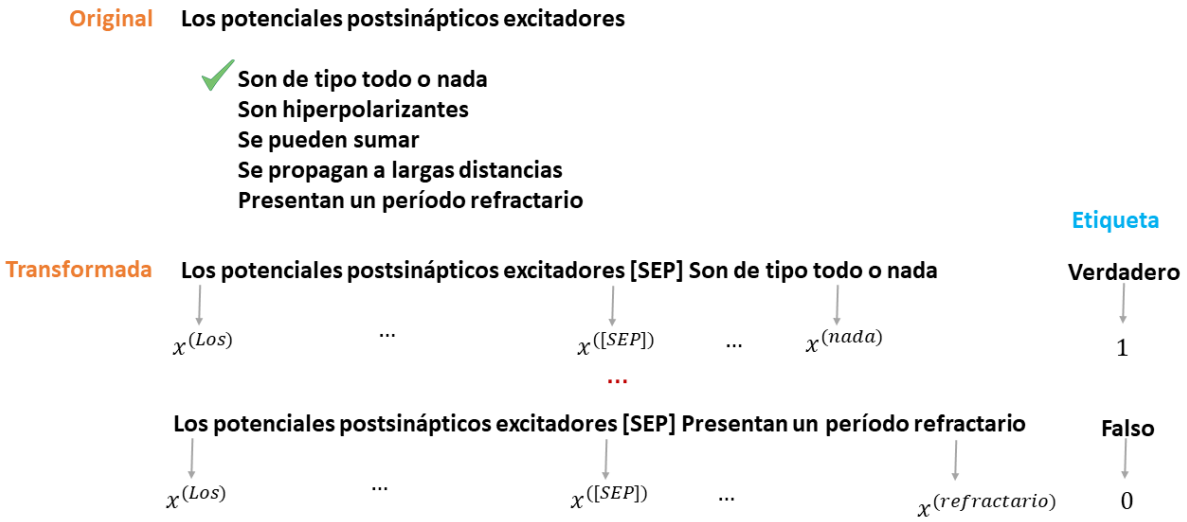

Figura 2.6: Ejemplo de instancia vectorizada según el enfoque supervisado clásico

foques requiere un preprocesamiento y representación de la información diferentes, los cuales se detallan a continuación.

2.3.1. Representación supervisada clásica

Este enfoque consiste en que los modelos reciban un único vector numérico, por lo tanto el objetivo es representar en un mismo vector la pregunta y cada respuesta.

Siguiendo los métodos *Pointwise* donde el problema de selección de respuestas se transforma en un problema de clasificación binario, a partir de una instancia del conjunto de datos original se construyen varios ejemplos vectorizados (ver Figura 2.6).

Cada instancia original se transforma en tantos vectores como respuestas posibles haya (pueden ser 4 o 5, dependiendo del examen). De manera que cada nuevo ejemplo vectorizado está formado por el texto de la pregunta y el texto de la respuesta separados por un carácter especial, introducido en el vocabulario de palabras con ese objetivo. Se le asigna 1 como etiqueta si el par $(pregunta, respuesta)$ es verdadero, en caso contrario, se le asigna la etiqueta negativa.

En la Figura 2.6 se muestra como una instancia del conjunto original es transformada en vectores numéricos, teniendo en cuenta la pregunta y si la respuesta es correcta o no.

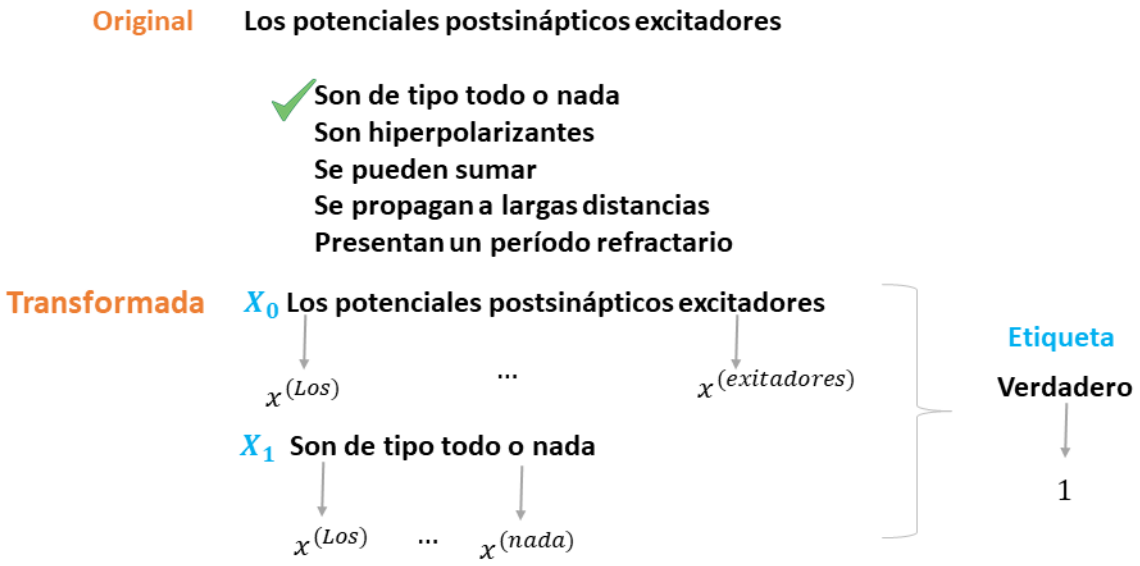

Figura 2.7: Ejemplo de instancia vectorizada según el enfoque RI

Con el objetivo de transformar una oración en una secuencia de *tokens* es necesario utilizar un *tokenizer*. Un *tokenizer* no es más que un algoritmo cuyo objetivo es separar texto en bruto en una secuencia de tokens. En este caso, se utiliza el tokenizador de la librería Spacy[2], específicamente el modelo `es_core_news_sm`. Este modelo es el más pequeño disponible en español, se selecciona el más pequeño porque la tarea de tokenizar no es muy complicada en términos lingüísticos por lo que la autora considera que no es necesario utilizar otro modelo más grande y complejo solo para realizar esta tarea.

2.3.2. Representación supervisada RI

El otro enfoque constituye una combinación entre el modelo vectorial clásico en el área de recuperación de información y un modelo de aprendizaje supervisado genérico. En este caso, los modelos de este perfil en lugar de un único vector numérico que represente ambos textos, la pregunta y la respuesta, están preparados para recibir como entrada una representación

[2]https://spacy.io/

vectorial de la pregunta y otra de la respuesta.

La transformación es muy similar a la vista en la sección anterior, con la diferencia de que al admitir dos entradas no es necesario concatenar ambas representaciones y por tanto, tampoco es necesario el *token* especial que actúa como separador.

En la Figura 2.7 se muestra un ejemplo de cómo es transformada una instancia del conjunto original, en varias instancias vectorizadas.

Al igual que en la situación anterior es necesario transformar las oraciones en bruto en una secuencia de *tokens* y se utiliza el mismo *tokenizer*. Una vez transformado el texto en bruto en representaciones numéricas, los datos están preparados para ser tratados por los modelos matemáticos.

Después del análisis, preprocesamiento y transformación del conjunto de datos original, los datos están listos para ser utilizados como entrada a los modelos. En el próximo capítulo se presentan varios modelos de aprendizaje supervisado, específicamente bajo el paradigma de aprendizaje profundo, los cuales constituyen propuestas de solución para resolver el problema de selección de respuestas en el idioma español sobre la base del corpus HeadQA detallado en el presente capítulo.

Capítulo 3

Modelos propuestos

En el presente capítulo se propone interpretar el problema de selección de respuestas como un problema de clasificación binaria, según el enfoque *Pointwise*, explicado en el Capítulo 1. Se ha seleccionado este enfoque por ser el más sencillo de los tres, y se propone para trabajos futuros incursionar en los demás métodos mencionados. En esta metodología cada par $< pregunta, respuesta >$ se convierte en una instancia y se clasifica en positiva o negativa dependiendo si la respuesta es correcta. Habiendo obtenido el corpus, el objetivo desde este punto es concebir modelos matemáticos-computacionales de aprendizaje profundo principalmente. En este capítulo se presentan diferentes propuestas de modelos para la selección automática de respuestas.

La decisión de implementar modelos de aprendizaje profundo, existiendo modelos supervisados más sencillos y explicables, se basa en que tras un estudio de los modelos del estado del arte, se puede concluir que la mayor parte con diferencia emplea este enfoque. La razón se debe a que estos conjuntos de datos son más complejos semánticamente por lo que requieren un mayor entendimiento del lenguaje humano y por lo tanto, modelos mucho más complejos.

De hecho, el aprendizaje supervisado puede no ser un enfoque adecuado para este conjunto de datos, puesto que como se afirma en Liu et al. (2020) incluso un modelo tan poderoso como BERT puede funcionar de manera insatisfactoria en el conjunto de datos HeadQA. La principal razón es que la pregunta inicial no contiene suficientes pistas recuperables para encontrar la respuesta correcta que contiene la respuesta y además, aunque exista una similitud entre los exámenes anteriores, puede no ser suficiente para generar patrones por un modelo supervisado.

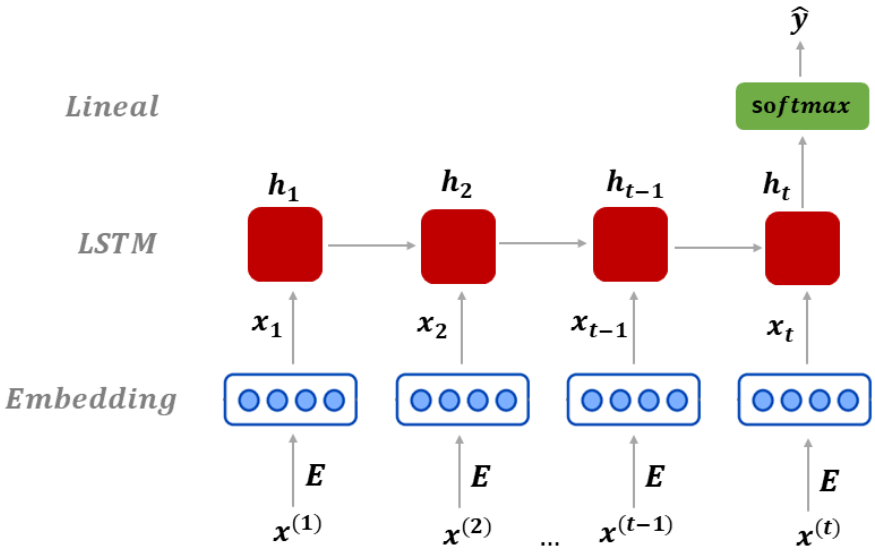

Figura 3.1: Arquitectura del modelo LSTM básico

3.1. *LSTM* Básico

La primera propuesta se puede considerar un modelo de aprendizaje profundo relativamente sencillo cuyo objetivo principal es verificar las ventajas que proporcionan las redes recurrentes en la resolución del problema de selección de respuestas.

La primera propuesta consiste en una red neuronal recurrente de tipo *LSTM*, la cual recibe como entrada una oración vectorizada. La arquitectura se divide en tres capas: representación de los *tokens*, representación de la oración y predicción de la etiqueta (respuesta correcta). En la Figura 3.1 se presenta la arquitectura de la red correspondiente al modelo matemático que se expone a continuación. El modelo matemático solo se incluirá en este primer caso, puesto que su sencillez lo permite y a la vez, permite familiarizarnos con las arquitecturas recurrentes.

Sea $S = [x^{(1)}, x^{(2)}, ..., x^{(T)}]$ la representación vectorial de una oración S, donde $x^{(t)}$ representa el t-ésimo *token* con $(t = 1, 2, ..., T)$ y T, la cantidad de *tokens* de la oración. El modelo puede ser definido formalmente como:

$$x_t = Ex^{(t)} \tag{3.1}$$

$$i_t = \sigma(W^{(i)}x_t + U^{(i)}h_{t-1}) \tag{3.2}$$
$$f_t = \sigma(W^{(f)}x_t + U^{(f)}h_{t-1}) \tag{3.3}$$
$$o_t = \sigma(W^{(o)}x_t + U^{(o)}h_{t-1}) \tag{3.4}$$
$$\tilde{c_t} = \tanh(W^{(c)}x_t + U^{(c)}h_{t-1}) \tag{3.5}$$
$$c_t = f_t c_{t-1} + i_t \tilde{c_t} \tag{3.6}$$
$$h_t = o_t \tanh c_t \tag{3.7}$$

$$\hat{y} = sigmoid(Uh_T + b) \tag{3.8}$$

En una primera etapa (Ecuación 3.1), el modelo obtiene una representación más rica semánticamente de cada uno de los *tokens* que conforman una oración. Esto se logra a través de una capa de *embeddings* E que transforma cada *token* $x^{(t)}$ en un vector $x_t \in \mathbb{R}^d$, donde d es la dimensión de los *embeddings* que constituye un hiperparámetro del modelo.

En la segunda etapa (Ecuaciones 3.2 - 3.7), el modelo procesa la secuencia de *tokens* para obtener una representación final de la oración. Esto se logra con la inclusión de una capa *LSTM*, la cual analiza de manera secuencial cada una de las salidas x_t de la capa de *embeddings*. El centro de una red *LSTM* es el funcionamiento de sus celdas, una red *LSTM* tiene tantas celdas como *tokens* tiene una oración. Las Ecuaciones 3.2 - 3.7 describen el comportamiento de una celda de la red *LSTM*. En la tabla 3.1 se describe la notación empleada.

Una celda *LSTM* está compuesta por tres componentes fundamentales:

- La *input gate*, en español "válvula de entrada", expresada en la Ecuación 3.2, tiene la función de determinar qué información debe estar presente en el estado de la celda (en inglés, *cell state*) representado por c_t, teniendo en cuenta la salida final de la celda anterior h_{t-1}.
- La *forget gate*, en español "válvula del olvido", representada en la Ecuación 3.3, tiene la función de determinar qué información es irrelevante en el estado de la celda c_t; funciona de manera análoga a la *input gate*.
- La *output gate*, en español "válvula de salida", representada en la Ecuación 3.4, tiene la función de determinar el nivel de activación del estado c_t para la salida final.

Tabla 3.1: Descripción de símbolos utilizados en una red *LSTM*

Símbolo	**Significado**
x_t	entrada a la celda *LSTM* en el paso t
h_{t-1}	salida de la celda *LSTM* en el paso $t-1$
i_t	valor de la *input gate* en el paso t
f_t	valor de la *forget gate* en el paso t
o_t	valor de la *output gate* en el paso t
σ	función sigmoidal
$W^{(\alpha)}$	pesos de x_t en α_t ($\alpha = i, f, o$)
$U^{(\alpha)}$	pesos de h_{t-1} en α_t ($\alpha = i, f, o$)
$\tilde{c_t}$	candidato a *cell state* en el paso t
c_t	*cell state* en el paso t
h_t	salida final de la celda *LSTM* en el paso t
h_T	salida final de la celda *LSTM* en el último paso T

En todos estos casos, se utiliza como función de activación la función sigmoide σ con el propósito de que los valores sean positivos y se encuentren entre 0 y 1.

La Ecuación 3.5, conocida como *new memory generation* o *candidate cell*, calcula un estado recurrente temporal $\tilde{c_t}$ teniendo en cuenta la entrada x_t y el estado anterior h_{t-1}. En este caso, para superar el problema de *vanishing gradient*[1] se necesita una función de activación cuya segunda derivada pueda mantenerse durante un largo rango antes de llegar a cero, razón por la cual se utiliza la tangente.

En la Ecuación 3.6 se calcula el estado de la celda actual c_t teniendo en cuenta qué debe olvidar del estado previo a través de la expresión $f_t * c_{t-1}$ y qué debe considerar del estado actual temporal representado en la expresión $i_t * \tilde{c_t}$. En la Ecuación 3.7 se toma el estado de la celda c_t y la *output gate* o_t para determinar qué información queda contenida en el estado último de la celda h_t.

Es importante destacar que los parámetros E, $W^{(i)}$, $W^{(f)}$, $W^{(o)}$, $W^{(c)}$, $U^{(i)}$, $U^{(f)}$, $U^{(o)}$, $U^{(c)}$ y U son aprendidos durante la etapa de entrenamiento en el proceso de *backpropagation*.

Las operaciones presentadas en las ecuaciones 3.2 - 3.7 son ejecutadas T veces, por cada uno de los *tokens* que conforman una oración. Tras el análisis del último *token* T, el estado final h_T

[1] https://towardsdatascience.com/the-vanishing-gradient-problem-69bf08b15484

constituye una representación de la oración S. Finalmente, una capa lineal (Ecuación 3.8) utiliza la representación de la oración obtenida h_T para predecir la relación $\hat{y}$ existente en la oración. Esta salida se convierte en una robabilidad empleando la función *sigmoid*. Esta operación, a diferencia de las ecuaciones anteriores, solo se aplica una vez al estado final h_T de la capa *LSTM*.

La sencillez arquitectónica del modelo permite que se pueda utilizar como base para la comparación con otros modelos más complejos que serán abordados a continuación.

3.2. *BiLSTM+Att*

El segundo modelo propuesto tiene una arquitectura más compleja que el anterior consistiendo, esencialmente, en una red *BiLSTM* integrada con un mecanismo de atención.

En este caso, se adiciona el uso de un modelo del lenguaje (en inglés, *language model*) (Bojanowski et al. (2016)) pre-entrenado en un conjunto de documentos en idioma español con el propósito de ganar riqueza semántica en la representación de la oración. Utilizar un modelo pre-entrenado ofrece la posibilidad de aprovechar el conocimiento del lenguaje contenido en la representación de las palabras (a través de los *embeddings* pre-entrenados en una tarea auxiliar) en nuestra tarea específica que, en este caso, es la selección de respuestas. El modelo del lenguaje empleado es el presentado por Bojanowski et al. (2016), entrenado sobre un conjunto de artículos médicos de la biblioteca electrónica *Scielo*[2] tomados de Soares et al. (2019).

Las redes recurrentes unidireccionales en general, en el análisis de una palabra, tienen en cuenta solo las palabras anteriores (o posteriores), sin embargo sería útil que al hallar la representación de una palabra se tuviera en cuenta tanto las palabras que aparecen antes como las que aparecen después en una oración. Por esta razón se decide utilizar una capa *BiLSTM*, esto posibilita que en cada estado de una secuencia, la red tenga una visión completa y consecutiva de todos los estados anteriores y posteriores. Se propone la inclusión de una capa de atención pues podría ayudar al modelo a darle más peso a ciertas palabras en la oración que pueden ser determinantes en la predicción.

En la Figura 3.2 se expone la arquitectura de la red correspondiente al modelo que se presenta en esta sección.

El modelo contiene una primera capa de *embeddings* pre-entrenada que transforma el *word index* en una representación más rica semánticamente representada como un vector de dimen-

[2]https://scielo.isciii.es/scielo.php

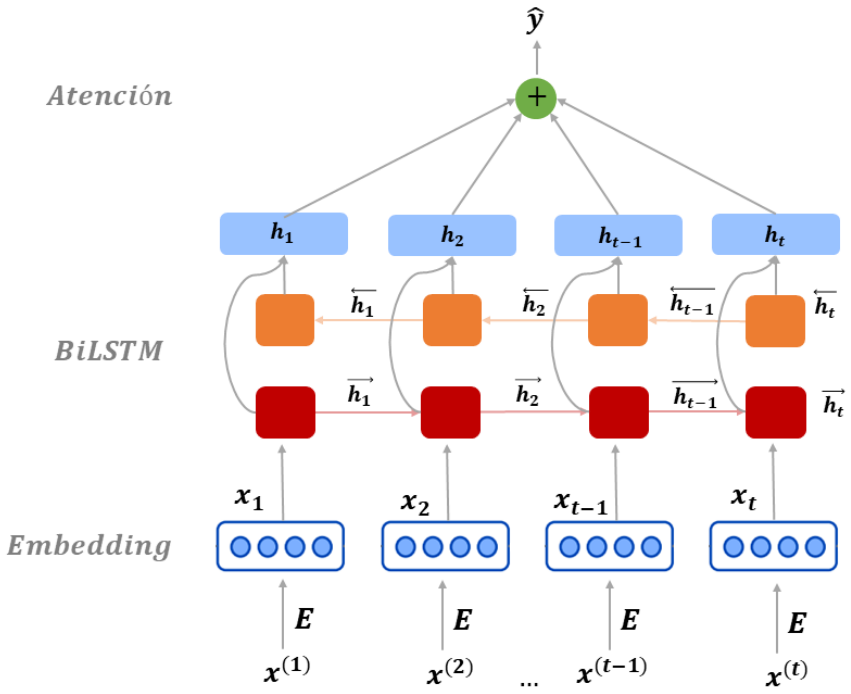

Figura 3.2: Arquitectura del modelo *BiLSTM+Att*

sión establecida. La diferencia en esta primera capa con respecto al modelo anterior se encuentra en que en el modelo *LSTM* los pesos de la capa de *embeddings E* se aprenden durante la etapa de entrenamiento, mientras que, en este caso, los pesos que se utilizan pertenecen a *embeddings* pre-entrenados tomados de Soares et al. (2019) que ya contienen un conocimiento del idioma español y del dominio médico específicamente, adquirido previamente.

Una segunda capa está conformada por una red *Bi-LSTM*, la cual permite tener en cuenta para el cómputo de un estado no solo las palabras anteriores sino también las siguientes. Posteriormente, se incluye la capa de atención. El objetivo de incluir un mecanismo de atención es que el modelo dé mayor peso a aquellas palabras que tienen una mayor influencia en la predicción final, de manera que su resultado es un vector de pesos, llamado vector de pesos de atención, que es combinado linealmente con la representación de las oraciones que se tenía en la capa anterior dando como resultado un nuevo vector, conocido como vector contexto, que constituye la representación final de la oración. Finalmente, una capa lineal utiliza la

representación de la oración obtenida para predecir si la respuesta es correcta o no.

3.3. QA-LSTM

Este modelo hace uso de las redes bidireccionales y a su vez emplea un enfoque clásico del área de recuperación de información, está inspirado en el introducido por Tan et al. (2015), aunque con algunas diferencias. En la Figura 3.3 se presenta la arquitectura del modelo. Es importante destacar que mientras las arquitecturas presentadas anteriormente combinaban la pregunta y la respuesta en una misma entrada, los modelos que se presentan a partir de este momento utilizan una representación separada para pregunta y respuesta. Por esta razón, en la figura se presentan dos veces la misma arquitectura, cuyo objetivo es codificar el texto perteneciente a la pregunta y a la respuesta por separado, aunque en la práctica se utiliza la misma red para ambos textos.

Ambas arquitecturas, notables en la base de la red por capas, comparten la misma arquitectura. Se ha visualizado de esta forma para lograr una mejor comprensión por parte del lector.

El modelo contiene una primera capa de *embeddings* pre-entrenada que transforma el *word index* en una representación más rica semánticamente representada como un vector de una dimensión fija para cada palabra. Al igual que en el modelo *BiLSTM+Attn* se utilizan embeddings pre-entrenados, se toman como base para la capa de *Embeddings* los vectores palabras orientados al dominio biomédico citados anteriormente y publicados por Soares et al. (2019).

Estos embeddings se utilizan como entrada a una capa LSTM bidireccional. La red BiLSTM genera representaciones distribuidas tanto para la pregunta como para la respuesta de forma independiente, y luego utilizan la similitud del coseno para medir su distancia.

Como se muestra en la figura y se explicó anteriormente en la introducción de las redes recurrentes de tipo LSTM, al aplicar este tipo de redes es posible obtener una representación de cada palabra a través de los estados intermedios y que brindan el atributo de recurrencia. De esta manera, en cada palabra tenemos un vector resultado de analizar las palabras anteriores y que se puede interpretar como una representación de dicha palabra en el contexto de sus antecesoras. Como estamos en presencia de una red recurrente bidireccional, en cada momento se cuenta con dos vectores, uno que constituye la representación de esa palabra a partir de sus anteriores y otros, a partir de sus posteriores.

En el modelo que se expone se utilizan todos los estados intermedios de la red para representar cada oración. Esto implica que tenemos dos vectores para la pregunta y dos para

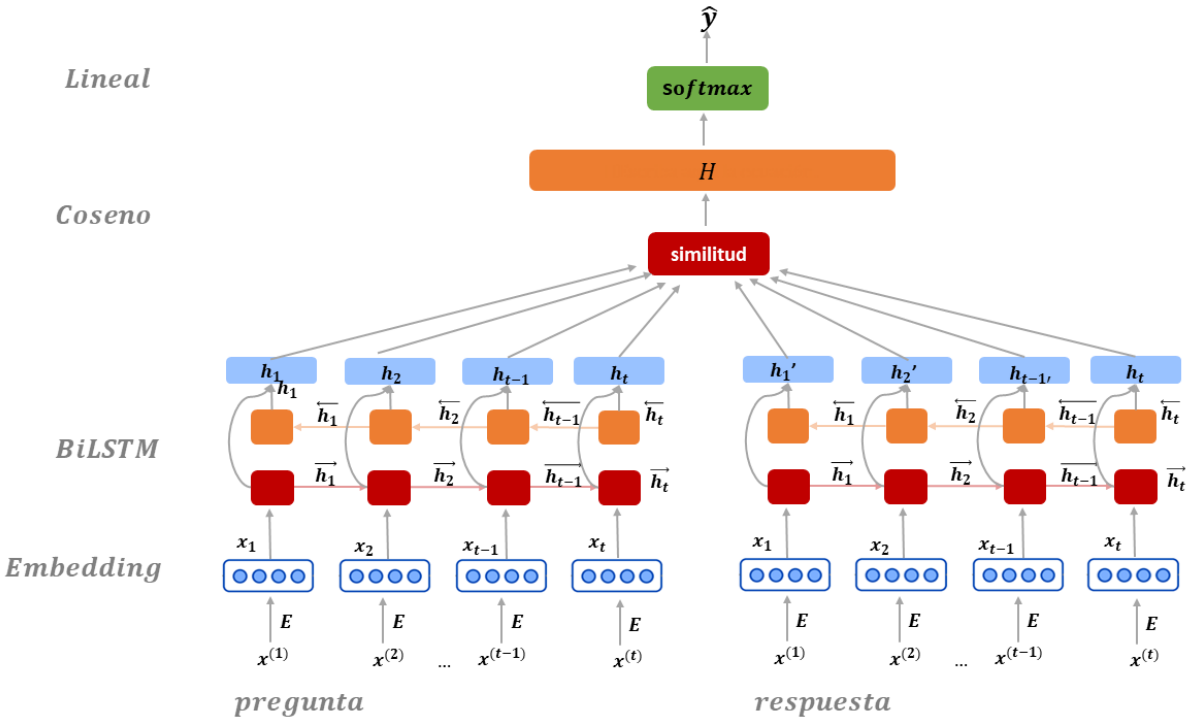

Figura 3.3: Arquitectura del modelo QA-LSTM

la respuesta, la representación final de cada una de las oraciones se obtiene concatenando los pares con que se contaba. A partir de estas representaciones de calcula la similitud del coseno resultando igualmente en un vector. Por último, este vector es conectado con una capa de regresión linear que utiliza *sigmoid* como función de activación, de manera que la salida en un número entre 0 y 1. Esta salida puede ser interpretada como el nivel de relevancia de la respuesta con la pregunta, y es necesario establecer un punto de corte para determinar si la instancia es positiva o negativa.

3.4. QA-LSTM/CNN

En esta sección se expone el modelo QA-LSTM/CNN, el cual es muy similar al anterior y su arquitectura fue también introducida por Tan et al. (2015). En la Figura 3.4 se muestra de

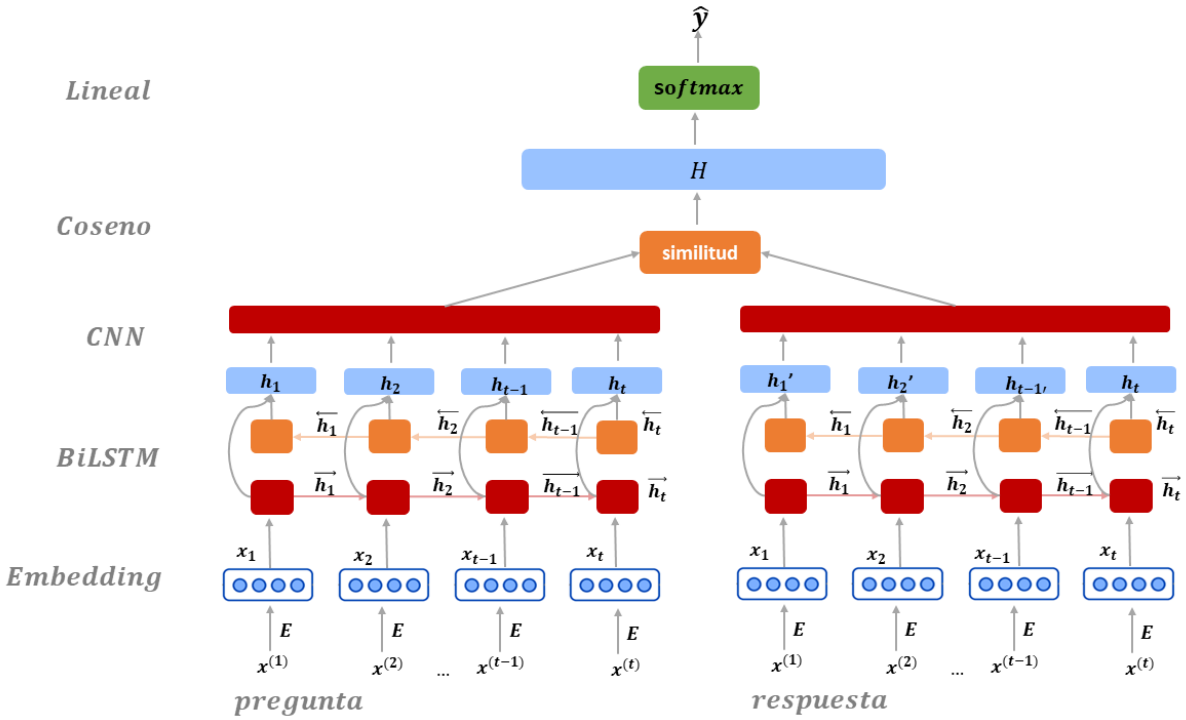

Figura 3.4: Arquitectura del modelo QA-LSTM/CNN

manera gráfica la arquitectura de este modelo. Como se puede apreciar es muy parecida a la arquitectura presentada anteriormente, la única diferencia es la inclusión de una capa convolucional tras las redes bidireccionales.

En la sección anterior, las representaciones de preguntas y respuestas se generan solo por simple operaciones, en este caso, la concatenación. Este nuevo modelo recurre a una estructura de CNN construida sobre los resultados de BiLSTM, con el fin de ofrecer una representación más compacta de las preguntas y las respuestas. A diferencia de la red neuronal tradicional, donde cada salida es interactiva con cada entrada, la estructura convolucional solo impone interacciones locales entre las entradas dentro de un tamaño de filtro m. Eesto quiere decir que la representación resultante tiene en cuenta la interacción de las palabras más cercanas, tanto a la derecha como a la izquierda.

Se decide incorporar esta arquitectura porque los autores de Tan et al. (2015) señalan que

alcanzan resultados superiores que con la arquitectura anterior. Sin embargo, aunque esto es una razón para incluirla en la experimentación no asegura el éxito dadas las diferencia de los contextos.

3.5. QA-BERT: *Transfer Learning*

Devlin et al. (2018) plantea que el uso de BERT en el enfoque *fine tuning* obtiene nuevos resultados en el estado del arte en varias tareas de procesamiento del lenguaje natural, entre ellas, el reconocimiento de entidades nombradas y los sistemas pregunta/respuesta.

La diferencia con otros modelos de representación del lenguaje, en cuanto a arquitectura, es que BERT está basado en un *Transformer* bidireccional. En el artículo de Devlin et al. (2018) no se detalla la arquitectura del modelo *Transformer*. Otra diferencia es que modifica la tarea tradicional de *language modeling* sobre la que se entrena introduciendo nuevas tareas. La primera tarea introducida recibe el nombre de *masked language model* y consiste en ocultar palabras aleatorias en un texto y predecir, teniendo en cuenta el contexto, el *token* que corresponde a la posición oculta, mientras que la segunda tarea consiste en, dada una oración en un texto, predecir la oración siguiente.

Los autores de BERT señalan como principal contribución la demostración de la efectividad del pre-entrenamiento bidireccional en la representación del lenguaje sobre las arquitecturas unidireccionales utilizadas hasta ese momento y, además, plantean que con el uso de esta nueva representación se puede reducir la complejidad de la arquitectura del modelo y aún así, obtener mejores resultados.

El modelo que se presenta en esta sección comparte la esencia del modelo QA-LSTM, pero en lugar de utilizar un red de tipo BiLSTM en las capas bases, utiliza un modelo BERT pre-entrenado. Los autores de BERT en su trabajo Devlin et al. (2018) afirman que el uso de BERT trae consigo una riqueza semántica mayor que los embeddings tradicionales. Aunque no fue probado formalmente que BERT tiene conocimiento del lenguaje humano, los resultados sugieren que este modelo posee una capacidad de entendimiento del lenguaje mayor que modelos vistos con anterioridad.

Bajo esta suposición, se considera que la representación que se puede lograr con BERT tanto de las preguntas como de las oraciones, puede tener más información del idioma español en general, dada la cantidad de información con que fue entrenado este sistema. El modelo BERT base que se utiliza fue entrenado sobre un corpus en idioma español presentado por Ca-

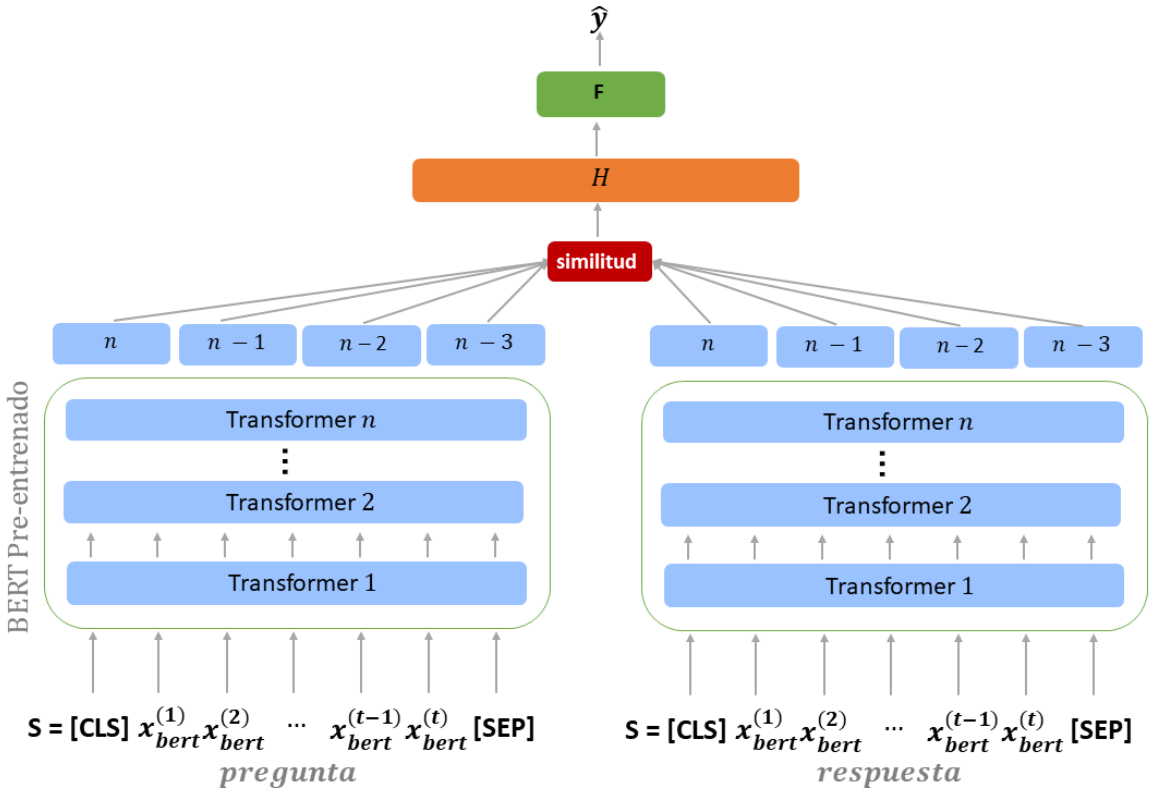

Figura 3.5: Arquitectura del modelo BERT-QA

ñete et al. (2020). En la Figura 3.5 se muestra gráficamente la arquitectura del modelo que se presenta.

BERT es básicamente una pila de codificadores de arquitectura *Transformer*, como se muestra en la base de la imagen. Una arquitectura *Transformer* es una red de codificador-decodificador basadas en diferentes mecanismos de atención. BERT BASE, la versión más ligera, tiene 12 capas en la pila del codificador, mientras que BERT LARGE tiene 24 capas en la pila del codificador. Cañete et al. (2020), modelo que se utiliza en esta propuesta, utiliza la arquitectura BERT BASE.

En esta arquitectura se propone utilizar la salida de la última capa de BERT como representación tanto de la pregunta como la respuesta y sobre estas representaciones aplicar las mismas operaciones que en el caso de LSTM-QA. Luego de obtener las representaciones contextuales

se aplica una medida de similitud, obteniendo un vector que es utilizado para dar la predicción final de la instancia, a través de una capa *fully connected* con función de activación sigmoidal, cuya salida puede interpretarse nuevamente como la relevancia de la posible respuesta a la pregunta.

3.5.1. Especificidad del preprocesamiento

Con el objetivo de transformar una oración en un vector computacionalmente interpretable, anteriormente se propuso un preprocesamiento que fue aplicado a los modelos de aprendizaje presentados. Sin embargo, en el caso de BERT, las oraciones no pueden ser traducidas a vectores numéricos de la misma forma que se hizo en las propuestas anteriores, sino que BERT requiere un formato diferente. Con el fin de obtener los resultados esperados, el corpus original siempre debe ser modificado en función de las pautas planteadas por los autores en Devlin et al. (2018). Asimismo, el *tokenizer* empleado no puede ser el mismo que el utilizado para los modelos anteriores, puesto que BERT tiene un vocabulario de palabras diferentes.

BERT define los siguientes *tokens* especiales para la representación de una oración. Dichos *tokens* reciben una interpretación diferente en la etapa de entrenamiento.

- **[CLS]**: Indica el inicio de una oración
- **[SEP]**: Indica el fin de una oración
- **[MASK]**: Se utiliza para denotar *tokens* que deben ser ignorados por el modelo

BERT no acepta secuencias de tamaño variable por lo que se recurre a la estrategia de *padding*, en la que se establece un tamaño máximo. Las oraciones que tienen una cantidad menor de *tokens* son completadas con el *token* especial **[MASK]**, para indicar que esas posiciones deben ser ignoradas.

3.6. BERT No Supervisado

Las propuestas no supervisadas publicadas hasta el momento se apoyan en un gran conjunto de documentos textuales de las temáticas relacionadas. Las propuestas presentadas en Vilares and Gómez-Rodríguez (2019) y Liu et al. (2020), las únicas a saber de la autora sobre este conjunto de datos, utilizan como fuente de información externa artículos de Wikipedia, artículos de la revista Scielo, entre otras.

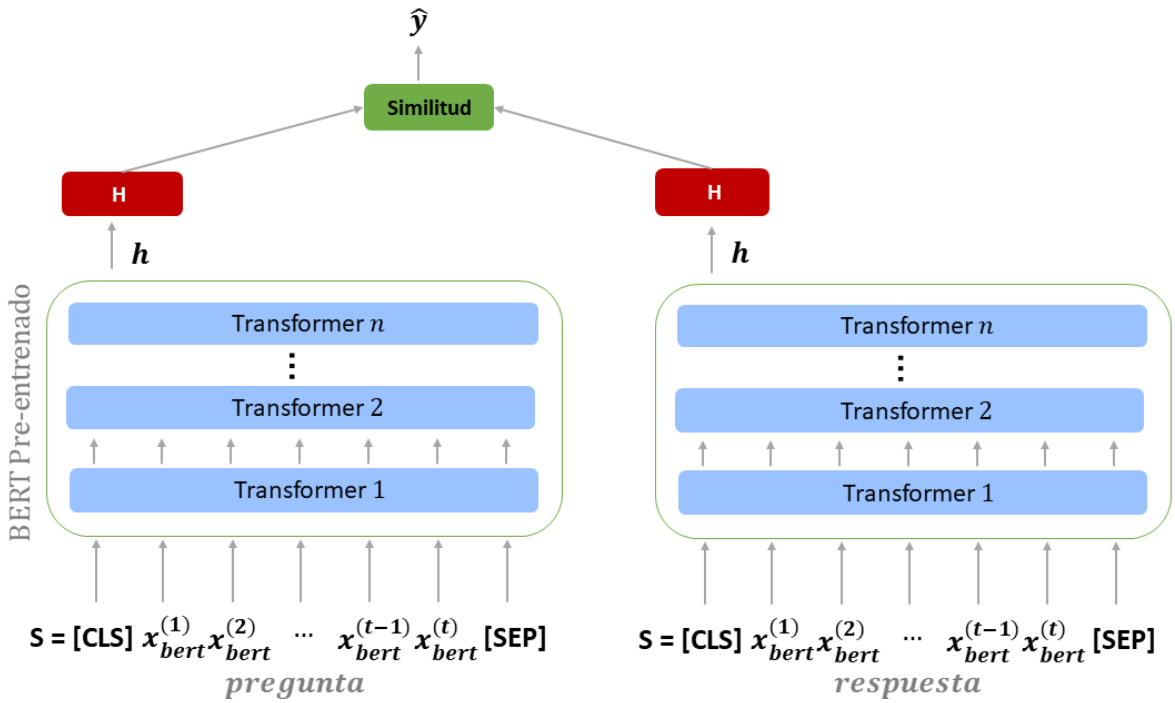

Figura 3.6: Arquitectura del modelo BERT-Sim

El gran volumen de datos que contienen estas bibliotecas hacen que el procesamiento sea muy costoso, tanto de recursos computacionales como de tiempo. Esta propuesta busca aprovechar los recursos y el tipo de procesamiento dedicados anteriormente proponiendo el uso de un modelo pre-entrenado.

Se propone hallar una representación a partir de BERT, esta representación tiene incluida de alguna manera conocimiento sobre los documentos con que fue entrenado BERT. Una limitante es que dado el coste de entrenar desde cero un modelo como BERT no existen modelos entrenados en idioma español especializados en el dominio biomédico por lo que se utiliza el presentado por Cañete et al. (2020), que ha sido entrenado sobre un gran corpus de documentos en idioma español.

En la Figura 3.6 se presenta la arquitectura del modelo en cuestión.

La idea es utilizar BERT como codificador de las oraciones de entrada, tanto la pregunta como la respuesta, y posteriormente, aplicar un modelo vectorial clásico. Como se mencionó

anteriormente BERT es básicamente una pila de codificadores de arquitectura *Transformer*, como se muestra en la base de la imagen, cuya versión más ligera y la utilizada en este caso (publicada en Cañete et al. (2020)), tiene 12 capas en la pila del codificador.

El modelo que se propone codifica la pregunta y la respuesta por separado utilizando la salida de las últimas cuatro capas de BERT. Los vectores salida se concatenan de manera que se obtiene un vector de gran tamaño para cada oración. A partir de estos vectores se aplica el coseno del ángulo como medida de similitud, cuyo resultado es la salida del modelo y puede ser interpretado como el nivel de correspondencia entre la pregunta y la respuesta.

El objetivo de este modelo tan sencillo es comprobar si un modelo tan poderoso como BERT puede a través de la representación contextual de las palabras que obtiene y un modelo vectorial clásico ser aplicado eficazmente a la selección de respuestas.

3.7. Implementación

En esta sección, se abordan los elementos esenciales para la programación y ejecución de los modelos propuestos y se describen de manera breve las herramientas utilizadas en el proceso de implementación.

En toda la solución se utilizó Python[3] como lenguaje de programación. Fue seleccionado porque debido a que el código es libre, existe una gran cantidad de bibliotecas implementadas por la comunidad de desarrolladores para dar respuesta a problemas diversos. De igual manera, Python constituye uno de los lenguajes más populares en el procesamiento del lenguaje natural. Contiene bibliotecas como Spacy[4] y NLTK[5] que cuentan con un gran número de funcionalidades implementadas. La completitud de Spacy y la facilidad de uso fueron determinantes en la selección como biblioteca para el procesamiento textual, empleada en la fase de preprocesamiento de las oraciones.

Asimismo, Python contiene varias alternativas en cuanto a bibliotecas de aprendizaje profundo y ofrece opciones para la visualización y comparación de resultados. Keras[6], TensorFlow[7] y PyTorch[8] se encuentran entre los tres marcos principales preferidos por los científicos

[3]http://www.python.org
[4]https://spacy.io/
[5]https://www.nltk.org/
[6]https://keras.io/
[7]https://www.tensorflow.org/
[8]https://pytorch.org/

de datos y los principiantes en el campo del aprendizaje profundo. En este caso, se utiliza PyTorch para implementar los modelos de aprendizaje pues es una herramienta de más bajo nivel que posibilita el trabajo con tensores, facilitando la manipulación interna de las arquitecturas neuronales, a diferencia de Keras y TensorFlow, que ofrecen APIs más generales.

En el capítulo 3 se presentó el proceso de concepción, modelación y diseño de los modelos a aplicar al problema de selección de respuestas como un problema de clasificación binario. Asimismo, se abordaron los detalles del proceso de implementación de los modelos expuestos. El próximo capítulo está dedicado a evaluar y comparar los modelos implementados en términos de métricas estándares para la evaluación de tareas de clasificación.

Capítulo 4

Resultados

A lo largo de esta tesis se han concebido y diseñado varios modelos para resolver la tarea de selección de respuestas a partir de un corpus previamente construido. En el presente capítulo, se expone un conjunto de experimentos desarrollados con el fin de medir el desempeño de los modelos propuestos en dicha tarea.

4.1. Medidas de Evaluación

Con el propósito de medir la eficacia de los modelos propuestos es necesario la aplicación de medidas estándares utilizadas en el paradigma del aprendizaje automático. Para la evaluación de los modelos propuestos se utiliza como métrica la **Exactitud**. Asimismo, teniendo en cuenta el escenario también se utiliza como métrica de evaluación la puntuación característica de estos exámenes y que es la empleada realmente para calificar los exámenes cada año.

Supongamos que estamos en presencia de un problema de clasificación binario cualquiera, donde existen dos clases que llamaremos C^+ y C^-. Los posibles tipos de respuestas de un modelo se definen como:

***True Positive* (TP)**: Un ejemplo se clasifica en **TP**, traducido al español como Verdadero Positivo, cuando la clase predicha coincide con la clase correcta C^+. En este caso, la predicción hecha fue correcta.

***False Positive* (FP)**: Un ejemplo se clasifica en **FP**, en español Falso Positivo, cuando el valor predicho es C^+, sin embargo su valor correcto es C^-. En este caso, el modelo hizo una

predicción equivocada.

***False Negative* (FN)**: Una instancia se clasifica en **FN**, en español Falso Negativo, cuando el valor predicho es C^-, sin embargo su clase correcta es C^+. En este caso, como en el anterior, el modelo hizo una predicción incorrecta.

***True Negative* (TN)**: Una instancia se clasifica en **TN**, en español, Verdadero Negativo, cuando la clase predicha corresponde con la clase correcta C^-, lo que implica que el modelo hizo una predicción acertada.

Estos tipos de respuestas se representan visualmente en la matriz de confusión que se muestra en la tabla 4.1.

Tabla 4.1: Matriz de confusión para un problema de clasificación binario

Valor Correcto/Valor Predicho	$\mathbf{C}^+$	$\mathbf{C}^-$
$\mathbf{C}^+$	TP	FN
$\mathbf{C}^-$	FP	TN

La **Exactitud** (en inglés *Accuracy*) se interpreta como la razón entre las predicciones correctas ($TP + TN$) y la cantidad total de instancias que analizó el modelo. Se computa sobre los resultados generales del modelo y se define como:

$$E = \frac{TP + TN}{\textit{Total de Ejemplos}} \tag{4.1}$$

La métrica estándar para calificar este tipo de exámenes consiste en sumar tres puntos si la respuesta está correcta y -1 si es incorrecta.

4.2. Técnicas de remuestreo

En esta sección se detalla el proceso de entrenamiento de los modelos propuestos anteriormente.

Dado que una instancia del conjunto original se convierte en varios ejemplos diferentes, una por cada posible respuesta y que solo una de las respuestas es correcta, el conjunto de

datos resultante del procesamiento de texto es un conjunto desbalanceado. Por una instancia positiva pueden aparecer al menos 3 instancias negativas.

Un conjunto de datos desbalanceado es el problema en el que los datos que pertenecen a una clase son significativamente más altos o más bajos que los que pertenecen a otras clases, como es el caso. La mayoría de los algoritmos de clasificación ML/DL no están equipados para manejar clases desequilibradas y tienden a inclinarse hacia las clases mayoritarias.

Para enfrentar este problema se han implementado varias técnicas de remuestreo cuyo objetivo es equilibrar la cantidad de muestras en cada una de las clases. La idea más sencilla para balancear un conjunto de datos es añadiendo muestras de la clase minoritaria o eliminando de la clase mayoritaria, sin embargo, estás técnicas tienen riesgo de *overfitting* y *underfitting* respectivamente. En la práctica resultan más efectivas técnicas avanzadas como SMOTE (del inglés, *Synthetic Minority Over-sampling Technique*) que crean datos sintéticos de la case minoritaria. Sin embargo, SMOTE no se desempeña bien en el ámbito de los datos textuales por la alta dimensionalidad de los vectores que se obtienen a partir de texto.

Teniendo en cuenta esto, se han implementado las siguientes técnicas de remuestreo:

- *Random Undersampling*: Este algoritmo consiste en eliminar del conjunto de datos transformados instancias negativas de manera aleatoria hasta igualar la cantidad de instancias positivas y negativas.
- *Random oversampling*: Este proceso es muy similar al anterior desde el punto de vista de la aleatoriedad pero adiciona instancias positivas al *dataset* hasta igualar la cantidad de instancias de cada etiqueta.
- *Mixed oversampling*: El objetivo de esta técnica es reproducir el comportamiento de los algoritmos de generación sintética de datos. El concepto también es similar a *Data Augmentation*, común en el procesamiento de imágenes, que implica la creación de nuevas instancias aplicando transformaciones (como rotar, trasladar, escalar, entre otras) las del conjunto de datos original. En este caso, para generar texto a partir de las oraciones originales se recurre a la traducción y al reemplazamiento por sinónimos. El algoritmo de traducción toma una oración y la traduce al idioma inglés, del inglés al alemán y luego, nuevamente al castellano. Se introduce un idioma intermedio para lograr una mayor diferencia entre la oración original y la resultante. Esto podría resultar contraproducente pues puede cambiar el sentido de la oración original, lo cual se verá en la evaluación de los modelos. Por otra parte, la otra técnica consiste en a partir de una oración obtener otra

compuesta por las palabras más parecidas a cada una de las palabras de la oración original. Las palabras más similares a un témino dado se han determinado a partir del uso de sus *word embeddings*. En este caso, se han utilizado *embeddings* obtenidos aplicando el algoritmo Fasttext sobre el *Spanish Billion Word Corpus*[1] descargado de Pérez (2019).

En las secciones posteriores se muestran los resultados de los modelos propuestos utilizando como entrenamiento las diferentes técnicas de remuestreo.

4.3. Entrenamiento

Además de los modelos de aprendizaje profundo detallados en el capítulo anterior, se implementa un modelo de regresión logística, que por su simplicidad, es utilizado como *baseline* supervisado, al no existir soluciones anteriores que utilicen este tipo de aprendizaje.

La métrica utilizada por los autores del *dataset* es la *Exactitud* y la cantidad de puntos, ambos a nivel de examen, de ahí que se hallan seleccioando como métricas de evaluación en este trabajo. Lo que quiere decir que, aunque los enfoques supervisados implementados interpretan el problema como una tarea de clasificación binaria y se puede calcular la exactitud sobre el conjunto de datos transformado, no se utilizará esta forma para expresar los resultados finales en aras de poder comparar los resultados con los ya existentes en la literatura.

Por esta razón, esta sección se centra en el proceso de entrenamiento y el análisis de los resultados sobre el conjunto de datos transformado. Mientras que la próxima sección se centra en el cálculo de la **Exactitud** sobre los datos originales y presenta los resultados finales comparables con los ya existentes.

Para analizar el proceso de entrenamiento nos centraremos en estudiar la evolución de los algoritmos implementados sobre el conjunto de datos de entrenamiento resultante de aplicar la técnica de remuestreo *Random Oversampling*. Se escoge esta técnica porque logra balancear el conjunto de datos y por otra parte, al duplicar no introduce instancias que pudieran no estar correctas sintácticamente como sí es posible cuando recurrimos a técnicas de traducción y/o reemplazamiento por sinónimos.

El objetivo de este análisis es comprobar si la transformación a un problema de clasificación binario en este escenario es válida y si se corresponden con los resultados finales sobre el conjunto original.

[1] http://crscardellino.github.io/SBWCE/

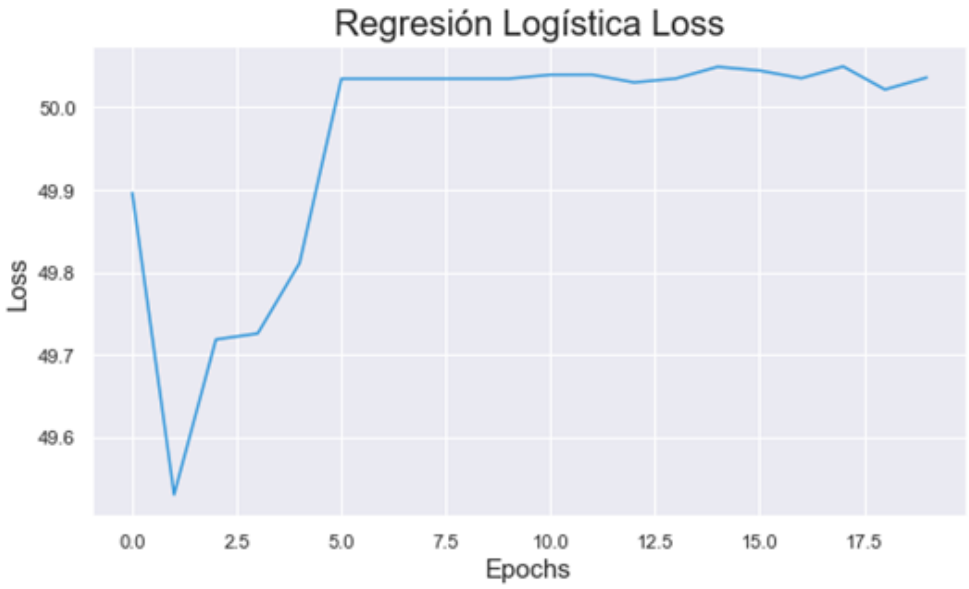

Figura 4.1: Función de pérdida del modelo Regresión Logística durante el entrenamiento

En la Figura 4.1 se muestra la evolución de la función de pérdida durante el entrenamiento del modelo de Regresión Logística, mientras que en la Figura 4.2 del resto de los modelos supervisados, exceptuando el basado en BERT. Este último no se incluyó porque la cantidad de epochs utilizada es muy pequeña. La función de pérdida utilizada fue *Binary Cross Entropy*.

Se puede observar que el modelo de regresión no logra minimizar la función de pérdida a partir de los datos de entrenamiento, por lo que se puede pensar que este modelo no tiene la capacidad o los datos de entrenamiento no son son suficientes, sufriendo de *underfitting*. También se puede notar en la segunda imagen que el modelo BiLSTM+Attn aunque logra minimizar la pérdida, se queda por encima del resto de los modelos que logran minimizar a valores cercanos a 0,05. A partir del comportamiento de esta función, se espera que los modelos LSTM, LSTM-QA y LSTM-QA/CNN sean los que mejor desempeño alcancen.

En este caso, como estamos en presencia de conjuntos de datos imbalanceados, aunque se

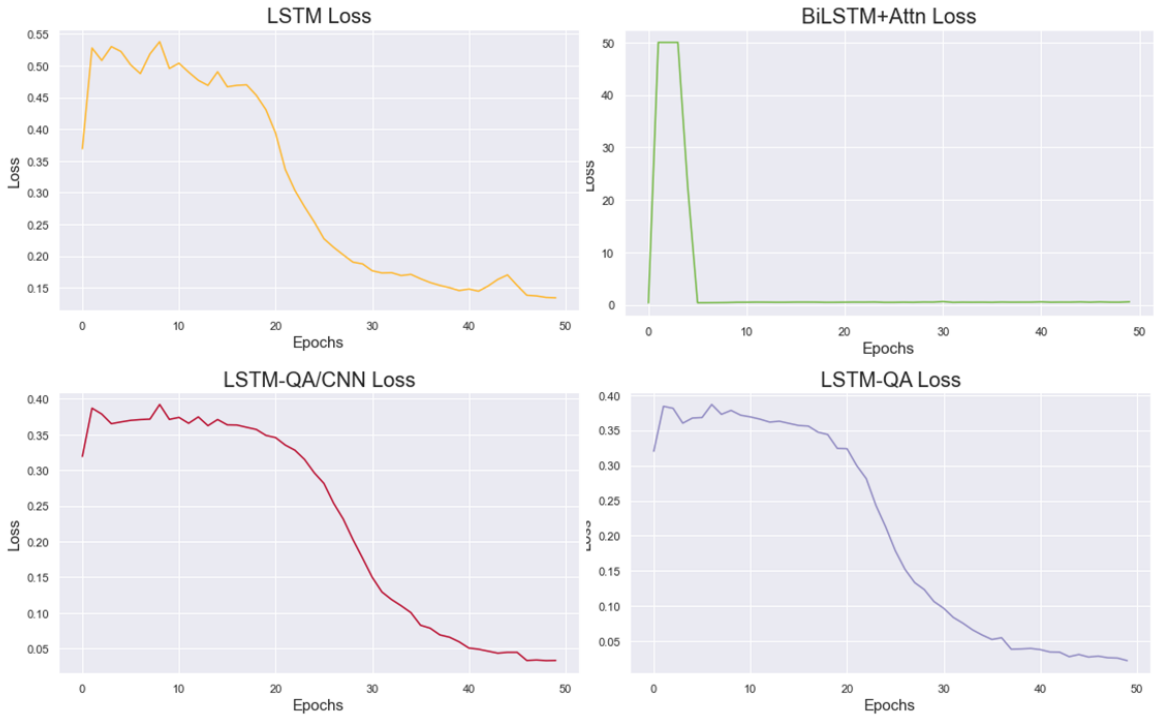

Figura 4.2: Función de pérdida de los modelos supervisados durante el entrenamiento

balanceó el de entrenamiento, el de validación y test no se modificaron en ese sentido, por lo que no es recomendable utilizar la exactitud como métrica de evaluación. Por esta razón, se visualiza la matriz de confusión para cada uno de los modelos entrenados.

En la Figura 4.3 se muestra la matriz de confusión de los modelos sobre el conjunto de validación, mientras que la Figura 4.4 corresponde a los resultados sobre el conjunto de test.

En la Figura 4.3 se puede notar como el algoritmo de regresión a pesar de no haber logrado optimizar la función de pérdida, da algunos resultados correctos. Por otra parte, es notable que LSTM y BiLSTM predicen siempre la clase positiva. Los últimos tres modelos son los que más han logrado aprender del conjunto de entrenamiento, sin embargo también se puede notar una tendencia a la clase positiva. Este comportamiento es muy similar en el conjunto de test como se puede observar en la Figura 4.4. Predecir la clase postiva significa que los modelos dan una gran relevancia a la mayoría de las posibles respuestas. Sin embargo, esto necesariamente no

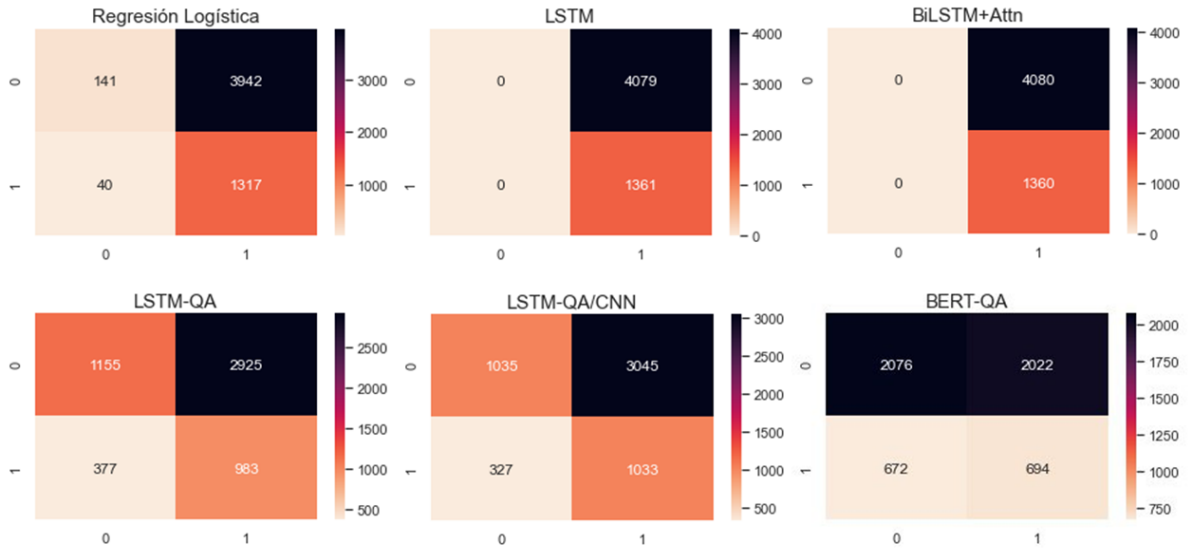

Figura 4.3: Matriz de confusión DEV

tiene por qué ser negativo al ser aplicado en el problema original, pues en ese caso se toma como correcta solo la respuesta con mayor relevancia.

Tras este análisis se puede concluir que si bien algunos de los algoritmos han logrado aprender de los datos, especialmente los que incluyen el enfoque de recuperación de información, el desempeño no ha sido muy bueno de manera general, aunque era esperado dados los resultados tan bajos que se han obtenido también en la literatura. Esto puede tener varias causas: los algoritmos sufren de *underfitting* y necesitan más entrenamiento sobre los mismos datos o necesitan más datos. El proceso de entrenamiento durante la experimentación se detuvo cuando se observó una estabilidad en la pérdida, por lo que existe una mayor posibilidad de que los datos no sean suficientes dada la complejidad del problema.

En la siguiente sección nos centraremos en exponer los resultados finales sobre el conjunto original.

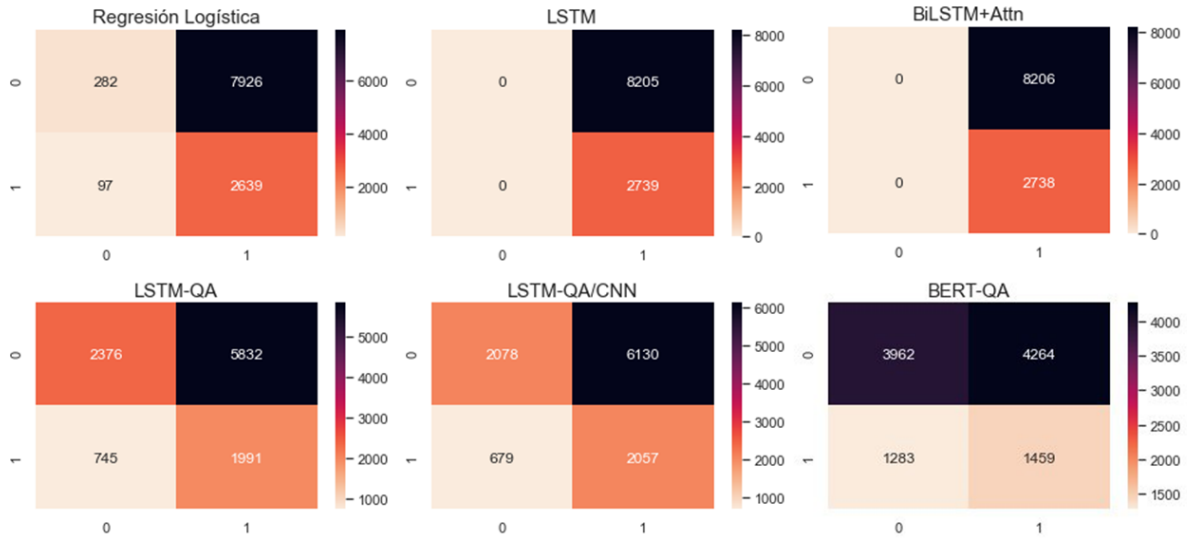

Figura 4.4: Matriz de confusión TEST

4.4. Evaluación

Como se mencionó anteriormente, las preguntas del conjunto de datos están separadas por categorías. Tras el análisis descriptivo inicial, se pudo notar que las palabrás más comunes al separar los datos diferían según la temática, o sea, el vocabulario varía de acuerdo al dominio. Por esta razón, se decide entrenar y evaluar los modelos tanto en el conjunto de datos completo, como en los subconjuntos resultantes de filtrar por materia.

La métrica utilizada por los autores del *dataset* es la *Exactitud* y la cantidad de puntos, ambos a nivel de examen. Lo que quiere decir que, aunque los enfoques supervisados implementados interpretan el problema como una tarea de clasificación binaria y se puede calcular la exactitud sobre el conjunto de datos transformado, no se utilizará esta forma en aras de poder comparar los resultados con los ya existentes en la literatura.

De manera que, la métrica **Exactitud** no se computará sobre el conjunto transformado a binario, sino sobre los exámenes originales. Por lo tanto, una instancia se considera correcta si la pregunta como un todo fue respondida correctamente y no si cada una de las respuestas fueron clasificadas en positivo o negativo correctamente. La **Exactitud**, además, se considera adecuada

ya que las clases en el problema actual (pregunta respondida correctamente o incorrectamente) tienen el mismo peso.

Dado que los modelos supervisados anteriores en su mayoría aplican la función sigmoide en la última capa, devuelven un número entre 0 y 1, que indica cuán relevante es la posible respuesta a la pregunta. Dada una pregunta del conjunto de datos original con sus posibles respuestas, cada par se transforma en una instancia vectorizada y se da como entrada al modelo de aprendizaje. La salida del modelo para cada una de las posibles respuestas es organizada y se elige como respuesta correcta la de mayor relevancia.

A continuación, se presentan los resultados alcanzados en este trabajo. Inicialmente se presentan los resultados alcanzados por categorías y posteriormente en el conjunto de datos general.

4.4.1. Evaluación por categorías

En esta sección, se analizan los resultados de aplicar las métricas antes mencionadas sobre el conjunto de evaluación obteniendo un conjunto de valores finales y concluyentes para cada modelo, de manera que se puedan comparar con los resultados actuales. El conjunto de validación está compuesto solamente por 6 exámenes, uno de cada categoría, mientras que el conjunto de evaluación está confomado por 12, 2 de cada rama.

En la Tabla 4.4.1 se presentan los resultados obtenidos en función de la cantidad de puntos por todos los modelos implementados, teniendo en cuenta las categorías. Asimismo en la Tabla 4.4.1, se comparan en función de la exactitud. Adicionalmente, en ambos casos, en las líneas inferiores se incluyen las métricas reportadas en Vilares and Gómez-Rodríguez (2019) sobre el conjunto de datos en español.

Las columnas de la tabla corresponden al nombre del modelo, el método de remuestreo aplicado sobre el conjunto de entrenamiento y los nombre de los exámenes de acuerdo a la categoría: Biología (MIR), Enfermería (EIR), Farmacología (FIR), Medicina (MIR), Psicología (PIR) y Química (QIR).

Los modelos incluidos en la parte inferior de la tabla consisten en:

- $RANDOM$: Seleccionar una respuesta de manera aleatoria.
- $BLIND_3$: Seleccionar siempre la tercera opción.
- $LENGTH$: Seleccionar la respuesta más larga.

Tabla 4.2: Comparación de los modelos por categorías por Puntos

Modelo	**Entrenamiento**	**BIR**	**EIR**	**FIR**	**MIR**	**PIR**	**QIR**
Regresión Logística	*Oversampled*	-3	7	-9	-12	-8	-7
LSTM	*Oversampled*	**43**	23	-17	-29	9	51
BiLSTM+Attn	*Oversampled*	-22	-6	-37	-33	-8	-9
QA-LSTM	*Oversampled*	41	17	-3	3	13	-5
QA-LSTM/CNN	*Oversampled*	-2	-16	-1	9	17	21
Regresión Logística	*Mixed*	3	-6	-9	-6	-8	-13
LSTM	*Mixed*	27	19	**24**	-40	-2	-65
BiLSTM+Attn	*Mixed*	-3	-6	-13	-6	-8	-9
QA-LSTM	*Mixed*	7	7	-11	47	-2	-45
QA-LSTM/CNN	*Mixed*	-15	1	6	35	29	**51**
QA-BERT	*Undersampled*	38	**25**	3	41	**49**	-26
Sim-BERT	*No*	-33	-4	-40	17	-26	7
RANDOM	-	-7	2.5	-10,5	-17,5	26,5	25
$BLIND_3$	-	9	44.5	19,5	22,5	-1,5	25
$LENGTH$	-	67	70,5	47,5	18,5	50,5	47
IR	-	105	100,5	139,5	12,5	98,5	103,5

- *IR*: Modelo de RI implementado en Vilares and Gómez-Rodríguez (2019) que utiliza como corpus artículos de Wikipedia.

Al analizar los resultados obtenidos en función de la cantidad de puntos, se puede notar que los algoritmos supervisados si bien alcanzan resultados superiores al modelo aleatorio la mayoría de ellos, no superan al modelo basado en recuperación de información. Los modelos que más baja puntuación alcanzan son el de regresión logística y el modelo no supervisado construido sobre BERT. Tras el análisis anterior de los modelos durante la etapa de entrenamiento, el comportamiento del regresor era esperado. Por otra parte, el fallo del modelo BERT-SIM reafirma que un simple *matching* o coincidencia entre las palabras de la pregunta y la respuesta no es suficientes para resolver este problema. Los resultados en función de la exactitud reflejan un comportamiento muy similar, por encima del modelo aleatorio pero por debajo del modelo IR. Los modelos que mejor desempeño han alcanzado son el LSTM, QA-LSTM y QA-BERT,

Tabla 4.3: Comparación de los modelos por categorías por Exactitud

Modelo	**Entrenamiento**	**BIR**	**EIR**	**FIR**	**MIR**	**PIR**	**QIR**
Regresión Logística	*Oversampled*	0,25	0,26	0,24	0,23	0,24	0,24
LSTM	*Oversampled*	**0,30**	**0,27**	0,23	0,22	0,26	**0,31**
BiLSTM+Attn	*Oversampled*	0,23	0,24	0,21	0,22	0,24	0,24
QA-LSTM	*Oversampled*	0,25	**0,27**	0,25	0,24	0,26	0,24
QA-LSTM/CNN	*Oversampled*	0,29	0,23	0,25	0,23	0,27	0,27
Regresión Logística	*Mixed*	0,25	0,24	0,24	0,24	0,24	0,23
LSTM	*Mixed*	0,28	**0,27**	**0,28**	0,20	0,25	0,32
BiLSTM+Attn	*Mixed*	0,25	0,24	0,24	0,24	0,24	0,24
QA-LSTM	*Mixed*	0,26	**0,27**	0,24	0,30	0,25	0,20
QA-LSTM/CNN	*Mixed*	0,23	0,25	0,26	0,29	0,28	**0,31**
QA-BERT	*Undersampled*	0,27	0,26	0,25	0,27	**0,28**	0,24
Sim-BERT	*No*	0,21	0,25	0,21	0,27	0,22	0,26
$RANDOM$	-	0,24	0,25	0,24	0,23	0,28	0,28
$BLIND_3$	-	0,26	0,29	0,27	0,27	0,24	0,27
$LENGTH$	-	0,32	0,32	0,30	0,27	0,30	0,30
IR	-	0,36	0,36	0,40	0,26	0,36	0,36

mostrando una correspondencia con el entrenamiento que confirma que el enfoque *Pointwise* es válido a pesar de no alcanzar buenos resultados.

4.4.2. Evaluación general

En esta sección se exponen los resultados tras entrenar y evaluar los modelos sobre todo el conjunto de datos, sin tener en cuenta las materias de cada pregunta. En este caso, solamente se exponen los resultados alcanzados por nuestros modelos, pues en el artículo Vilares and Gómez-Rodríguez (2019) no se desarrollaron modelos siguiendo este enfoque.

En la Tabla 4.4.2 se presentan los resultados obtenidos en función de la cantidad de puntos por todos los modelos implementados. De manera análoga, en la Tabla 4.4.2 se muestra en función de la exactitud. En cada caso se muestras las puntuaciones media y máximas obtenidas.

Nuevamente los modelos que menor desempeño muestran son el regresor logístico y BERT-

Tabla 4.4: Comparación general por cantidad de puntos TEST

Modelo	**Entrenamiento**	**Media Pts.**	**Max Pts.**
Regresión Logística	Desbalanceado	-8	29
LSTM	Desbalanceado	1	37
BiLSTM+Attn	Desbalanceado	15	93
QA-LSTM	Desbalanceado	15	74
QA-LSTM/CNN	Desbalanceado	11	36
Regresión Logística	*Oversampled*	-5	37
LSTM	*Oversampled*	2	56
BiLSTM+Attn	*Oversampled*	-2	21
QA-LSTM	*Oversampled*	9	45
QA-LSTM/CNN	*Oversampled*	7	61
Regresión Logística	*Mixed*	-9	21
LSTM	*Mixed*	11	39
BiLSTM+Attn	*Mixed*	-3	40
QA-LSTM	*Mixed*	**23**	**101**
QA-LSTM/CNN	*Mixed*	4	49
QA-BERT	*Undersampled*	3	41
Sim-BERT	*No*	-13	29

SIM. Mientras que el que mayor puntuación alcanza es LSTM-QA cuando fue entrenado con el conjunto balanceado empleando el algoritmo *Mixed Oversampling*. Se puede notar que la media de las puntuaciones al entrenar con todos los datos es muy similar a la alcanzada por los modelos independientes. Mientras que la puntuación máxima generalmente muestra un incremento.

Sin embargo, se debe tener en cuenta que el conjunto de datos de evaluación consta de 12 exámenes distribuidos en las 6 categorías y el de validación, con solo 6. Por esta razón y atendiendo a los resultados, no se considera que este *dataset* esté preparado aún para ser utilizado por algoritmos supervisados. Si bien es cierto que la cantidad de preguntas es grande, la probabilidad de que una pregunta sea similar a preguntas de años anteriores es baja cuando solo se cuenta con un histórico de alrededor de 5 años, teniendo en cuenta que cada examen tiene entre 200 y 250 preguntas y que los exámenes están separados por categorías.

Tabla 4.5: Comparación general por Exactitud TEST

Modelo	**Entrenamiento**	**Media Exact.**	**Max Exact.**
Regresión Logística	Desbalanceado	0, 24	0,28
LSTM	Desbalanceado	0,25	0,29
BiLSTM+Attn	Desbalanceado	**0,27**	**0,35**
QA-LSTM	Desbalanceado	0,26	0,33
QA-LSTM/CNN	Desbalanceado	0,26	0,29
Regresión Logística	*Oversampled*	0.24	0,29
LSTM	*Oversampled*	0,25	0,31
BiLSTM+Attn	*Oversampled*	0,25	0,27
QA-LSTM	*Oversampled*	0,26	0,30
QA-LSTM/CNN	*Oversampled*	0,26	0,32
Regresión Logística	*Mixed*	0,24	0,27
LSTM	*Mixed*	0,26	0,29
BiLSTM+Attn	*Mixed*	0,25	0,29
QA-LSTM	*Mixed*	**0,27**	**0,36**
QA-LSTM/CNN	*Mixed*	0,25	0,30
QA-BERT	*Undersampled*	0,25	0,30
Sim-BERT	*No*	0,23	0,28

De manera general, se puede afirmar que no hay grandes diferencias en los resultados en función de los datos de entrenamiento y las técnicas de remuestreo utilizadas.

Los algoritmos de clasificación, a diferencia de los algoritmos de regresión, durante el entrenamiento minimizan una función de pérdida seleccionada y no directamente, la exactitud de los modelos, aunque ambas estén relaciondas. De manera que, el desempeño de un modelo, depende en gran medida de la función de pérdida seleccionada. La autora considera que una función de pérdida diseñada para maximizar la importancia de una respuesta correcta y penalizar una incorrecta podrá ofrecer notables ventajas y mejorar el desempeño de los algoritmos supervisados y se incluye como recomendación para trabajos futuros el empleo de una función de pérdida diferente.

4.5. Discusión de los resultados

Tras el análisis de los resultados se puede afirmar que el conocimiento adquirido por los modelos durante la etapa de entrenamiento es válido y puede ser utilizado para la selección de respuestas, dado que en la mayoría de los casos sobrepasan al modelo aleatorio aunque por muy poco. Los modelos supervisados no mostraron un desempeño superior a los modelos del estado del arte sobre este conjunto de datos, pero tampoco obtuvieron resultados muy por debajo. Sin embargo y aunque los resultados son comparables con los modelos previamente existentes, continúan estando muy lejos de las puntuaciones alcanzadas por humanos. Teniendo en cuenta que en el año 2016 la media de las 10 mejores notas según Vilares and Gómez-Rodríguez (2019) está alrededor de los 600 puntos y las notas de corte cercanas a los 200, los modelos automáticos aún están lejos de resolver este problema.

Los modelos han demostrado que no tienen la capacidad de resolver la selección de respuestas automáticamente sobre este conjunto de datos. Dada la complejidad del problema en cuestión, se considera que la cantidad de datos no es suficiente para construir un modelo robusto basado solamente en aprendizaje supervisado.

Conclusiones

Como resultado del presente trabajo se logró la creación de una solución computacional orientada a la selección de respuestas de manera automática en idioma español.

Se logró la concepción y el diseño de diversos modelos matemáticos de aprendizaje profundo enfocados en resolver la tarea de selección de respuestas. La comparación de los modelos a través de métricas descritas permitieron establecer la validez de la solución. Si bien los modelos supervisados no lograron sobrepasar a los modelos no supervisados ya existentes, sí demostraron que el enfoque supervisado es válido en este escenario, aún cuando sería conveniente ampliar el conjunto de datos actual, y constituyen una base para el desarrollo de futuras propuestas supervisadas.

Al constituir el primer método que aplica el enfoque de supervisión no hay referentes directos para la comparación. Sin embargo, la comparación con el desempeño humano permite afirmar que aún el problema de selección de respuestas en idioma español y sobre dominios del conocimiento complejos no está resuelto. Esta solución constituye un primer paso puede ser aprovechada por futuras investigaciones.

Los resultados de la investigación permiten responder afirmativamente a la pregunta científica, ya que ha sido posible crear un modelo matemático computacional de selección de respuestas en textos en idioma español.

Trabajo Futuro

Como trabajo futuro se propone:

- Aplicar otros enfoques para la resolución del problema de selección de respuestas, como los mencionados en el capítulo primero que se correspondan igualmente con los técnicas del estado del arte.
- Enriquecer el conjunto de datos actual con la inclusión de exámenes similares obtenidos de otras fuentes.
- Diseñar y/o aplicar una función de pérdida que tenga en maximice la importancia de una respuesta correcta y penalice una incorrecta.
- Diseñar algoritmos híbridos que combinen las técnicas anteriormente empleadas en la literatura bajo el enfoque de recuperación de información no supervisada o supervisada a distancia, con modelos del aprendizaje supervisado.

Bibliografía

Abacha, A. B., Dinh, D., and Mrabet, Y. (2015). Semantic analysis and automatic corpus construction for entailment recognition in medical texts. In Holmes, J. H., Bellazzi, R., Sacchi, L., and Peek, N., editors, *Artificial Intelligence in Medicine*, pages 238–242, Cham. Springer International Publishing.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2016). Enriching word vectors with subword information. *CoRR*, abs/1607.04606.

Bowman, S. R., Angeli, G., Potts, C., and Manning, C. D. (2015). A large annotated corpus for learning natural language inference. *CoRR*, abs/1508.05326.

Cañete, J., Chaperon, G., Fuentes, R., Ho, J.-H., Kang, H., and Pérez, J. (2020). Spanish pre-trained bert model and evaluation data. In *PML4DC at ICLR 2020*.

Chen, D., Fisch, A., Weston, J., and Bordes, A. (2017). Reading wikipedia to answer open-domain questions. *CoRR*, abs/1704.00051.

Chollet, F. (2017). *Deep Learning with Python*. Manning.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

dos Santos, C. N., Tan, M., Xiang, B., and Zhou, B. (2016). Attentive pooling networks. *CoRR*, abs/1602.03609.

Feng, M., Xiang, B., Glass, M. R., Wang, L., and Zhou, B. (2015). Applying deep learning to answer selection: A study and an open task. *CoRR*, abs/1508.01585.

Goyal, P., Pandey, S., and Jain, K. (2018). Deep learning for natural language processing. *Deep Learning for Natural Language Processing: Creating Neural Networks with Python [Berkeley, CA]: Apress*, pages 138–143.

Heilman, M. and Smith, N. A. (2010). Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, page 1011–1019, USA. Association for Computational Linguistics.

Lai, T. M., Bui, T., and Li, S. (2018). A review on deep learning techniques applied to answer selection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2132–2144, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Liu, Y., Chowdhury, S., Zhang, C., Caragea, C., and Yu, P. S. (2020). Interpretable multi-step reasoning with knowledge extraction on complex healthcare question answering. *CoRR*, abs/2008.02434.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA. Curran Associates Inc.

Nentidis, A., Krithara, A., Bougiatiotis, K., Paliouras, G., and Kakadiaris, I. (2018). Results of the sixth edition of the BioASQ challenge. In *Proceedings of the 6th BioASQ Workshop A challenge on large-scale biomedical semantic indexing and question answering*, pages 1–10, Brussels, Belgium. Association for Computational Linguistics.

Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Pérez, J. (2019). Spanish word embeddings. `https://github.com/dccuchile/spanish-word-embeddings`. Consultado el 2021-08-28.

Rao, D. and McMahan, B. (2019). *Natural language processing with PyTorch: build intelligent language applications using deep learning*. .ºReilly Media, Inc.".

Severyn, A. and Moschitti, A. (2015). Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, page 373–382, New York, NY, USA. Association for Computing Machinery.

Soares, F., Villegas, M., Gonzalez-Agirre, A., Krallinger, M., and Armengol-Estapé, J. (2019). Medical word embeddings for Spanish: Development and evaluation. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 124–133, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Tai, K. S., Socher, R., and Manning, C. D. (2015). Improved semantic representations from tree-structured long short-term memory networks. *CoRR*, abs/1503.00075.

Tan, M., Xiang, B., and Zhou, B. (2015). Lstm-based deep learning models for non-factoid answer selection. *CoRR*, abs/1511.04108.

Tran, Q. H., Lai, T., Haffari, G., Zukerman, I., Bui, T., and Bui, H. (2018). The context-dependent additive recurrent neural net. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1274–1283, New Orleans, Louisiana. Association for Computational Linguistics.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *CoRR*, abs/1706.03762.

Vilares, D. and Gómez-Rodríguez, C. (2019). HEAD-QA: A healthcare dataset for complex reasoning. *CoRR*, abs/1906.04701.

Wang, Z., Hamza, W., and Florian, R. (2017). Bilateral multi-perspective matching for natural language sentences. *CoRR*, abs/1702.03814.

Yao, X., Van Durme, B., Callison-Burch, C., and Clark, P. (2013). Answer extraction as sequence tagging with tree edit distance. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 858–867, Atlanta, Georgia. Association for Computational Linguistics.

Yih, W.-t., Chang, M.-W., Meek, C., and Pastusiak, A. (2013). Question answering using enhanced lexical semantic models. In *Proceedings of the 51st Annual Meeting of the Association for*

Computational Linguistics (Volume 1: Long Papers), pages 1744–1753, Sofia, Bulgaria. Association for Computational Linguistics.

Yu, L., Hermann, K. M., Blunsom, P., and Pulman, S. (2014). Deep learning for answer sentence selection. *CoRR*, abs/1412.1632.