



ESCUELA DE INGENIERÍA DE FUENLABRADA

GRADO EN INGENIERÍA BIOMÉDICA

TRABAJO FIN DE GRADO

**DESARROLLO DE UN CHATBOT ESPECIALIZADO EN
EQUIPAMIENTO BIOMÉDICO BASADO EN GPT-3**

Autor: Miriam Cristina Vallejos Rojas

Tutor: Borja Rodríguez Vila

Co-tutor: David Viar Hernández

Curso académico 2022/2023

Nunca tengas miedo a no lograrlo. Ten miedo, mucho miedo, a no llegar a intentarlo.

– Dr. Francisco Salmerón

La vida es 10% lo que te pasa y 90% cómo reaccionas

– Charles Swindoll

Agradecimientos

Para comenzar, quiero agradecer a mi profesor y tutor del TFG, Borja. Gracias por ofrecerme tu apoyo y ánimos que me han alentado durante esta etapa. Por supuesto, también agradecer a David Viar por proponer la idea de este trabajo y a Ángel Torrado por confiar en mí y asignarme la realización de este proyecto.

Tras cuatro años de carrera cierro otro capítulo de mi vida, y es incalculable el número de personas involucradas en esta etapa. Gracias a todos los profesores que habéis compartido vuestra sabiduría, pasión y dedicación. Y sobre todo, gracias a los amigos que he hecho durante el camino, vuestra presencia ha sido fundamental. Son muchas las horas que hemos pasado juntos, horas de alegría, estrés y preocupación que nos han unido y permitido conocernos mejor. Llevo conmigo algo de cada uno de vosotros, sois gente increíble.

A mis amigos de toda la vida, porque siempre habéis estado presentes, incluso cuando apenas disponía de tiempo de calidad para dedicaros. Agradezco y valoro todo el interés por cada paso que daba y vuestra alegría por mis triunfos. Seguiremos creciendo juntos como hasta ahora.

A ti, Volker, por acompañarme y darme fuerzas en esta etapa final. Por aportarme la tranquilidad que tanta falta me hacía y ayudarme a despejar la mente siempre que necesitaba un respiro. Gracias por atender mis preocupaciones y enseñarme que los problemas se enfrentan de uno en uno.

Para finalizar, quiero agradecer a las personas más influyentes en mi vida, mi familia. Por confiar tanto en mí y admirar cada paso que daba. Aunque me gustaría dedicarle un reconocimiento especial a la persona más fuerte y valiente que conozco, mi madre. Tus palabras y abrazos han sido de vital importancia cuando las circunstancias que tenía que enfrentar parecían complicadas. Gracias por ayudarme a respirar y enseñarme que las cosas son más simples de lo que parecen en primera instancia, todo depende de la perspectiva.

Resumen

En los últimos años, la Inteligencia Artificial está experimentando un auge sin precedentes. Son muchas las empresas que gracias a la investigación e interés en el campo han desarrollado diversos modelos de procesamiento del lenguaje natural capaces de comprender y generar tanto lenguaje natural como código. El progreso de estos modelos ha sido crucial para el desarrollo de los chatbots, permitiendo una interacción más natural y eficiente con los usuarios.

El objetivo de este Trabajo Final de Grado (TFG) radica en el entrenamiento específico de un modelo de lenguaje de OpenAI, concretamente el modelo Davinci, con el propósito de desarrollar un chatbot capacitado para responder preguntas sobre equipamiento biomédico. Como plataforma de comunicación entre el usuario y el modelo se utilizará la aplicación de mensajería instantánea y gratuita, Telegram.

El estudio comienza con la recopilación de datos de Wikipedia sobre varios equipos biomédicos como Tomografía Computarizada, Resonancia Magnética, Rayos X, Ultrasonidos y Medicina Nuclear. Se realiza el preprocesamiento y preparación adecuado de los datos para el entrenamiento específico del modelo de lenguaje.

Posteriormente, se lleva a cabo el proceso de entrenamiento, mejorando la capacidad del modelo para proporcionar respuestas cortas y precisas sobre los temas relacionados con el equipamiento biomédico mencionado anteriormente.

Finalmente, se crea un bot a través de la API de Telegram que permite utilizar como interfaz los mensajes de la aplicación. Entonces, se establece la conexión del modelo ajustado con el bot para lograr obtener el chatbot especializado en equipamiento biomédico, capaz de responder a las preguntas de los usuarios de Telegram.

Los resultados obtenidos muestran un buen rendimiento y gran satisfacción por parte de los usuarios en lo que a utilidad y usabilidad se refiere. A pesar de las posibles mejoras, estos modelos de lenguaje preentrenados allanan el camino considerablemente, simplificando el desarrollo de chatbots, que en circunstancias originales requerían elaborar el modelo desde el inicio.

Índice general

Agradecimientos

Resumen

Índice de figuras III

Índice de tablas V

Lista de acrónimos y abreviaturas VII

1. Introducción 1

1.1. Contexto y motivación 1

1.2. Objetivos 2

1.3. Estructura de la memoria 3

2. Fundamentos teóricos 5

2.1. Conceptos previos 5

2.2. Estado del arte 9

3. Diseño del proyecto 13

3.1. Planificación temporal 13

3.2. Diseño general de la propuesta 14

3.3. OpenAI y los modelos GPT-3 14

3.4. Python	17
3.5. Wikipedia	18
3.6. Postman	19
3.7. Telegram	20
4. Desarrollo del proyecto	21
4.1. Recopilación y preparación de datos	21
4.1.1. Extracción de datos de Wikipedia	21
4.1.2. Formulación de preguntas y respuestas sintéticas	23
4.1.3. Ajuste de formato de los datos	25
4.2. Entrenamiento específico	26
4.3. Implementación en Telegram	28
4.4. Presupuesto	29
5. Experimentos y resultados	31
5.1. Evaluación interna	31
5.1.1. Impacto de la temperatura	31
5.1.2. Capacidad de generalización	35
5.2. Evaluación externa	37
5.2.1. Interacción con usuarios	37
5.2.2. Cuestionario	39
6. Discusión	43
6.1. Análisis de resultados	43
7. Conclusión y líneas futuras	47
7.1. Conclusión	47
7.2. Líneas futuras	48
Bibliografía	51

Índice de figuras

2.1. Arquitectura del modelo <i>transformer</i> . Figura extraída de [1].	7
2.2. Ejemplo tokenización de una frase. Separación en <i>tokens</i> y asignación de identificadores. Figura extraída/modificada de [2].	8
2.3. Proyección en 2D de Word Embeddings. Se muestran cinco clusters diferentes. Figura extraída de [3].	8
3.1. Diagrama de Gantt.	13
3.2. Esquema de componentes del proyecto.	14
3.3. Ejemplo de JSONL.	17
3.4. Visualización de la estructura de Wikipedia: (a) subcategorías de la categoría principal “ <i>Ultrasound</i> ”; (b) primeras nueve secciones del índice de la página “ <i>Medical Ultrasound</i> ” perteneciente a la subcategoría “ <i>Medical Ultrasonography</i> ”. Figura extraída de [4] [5].	19
4.1. Ejemplo de DataFrame donde se muestra el encabezado y las cinco primeras entradas que corresponden con las cinco primeras secciones de la página de Wikipedia “ <i>Medical ultrasound</i> ”.	23
4.2. Lectura del archivo CSV con la biblioteca Pandas donde se muestran las preguntas y respuestas sintéticas.	25
4.3. Ejemplo de la estructura de un JSONL listo para el reentrenamiento del modelo GPT-3.	26
4.4. Ejemplo del cuerpo de una solicitud POST para el proceso de entrenamiento específico de un modelo de lenguaje en OpenAI.	28

5.1. Gráfico de la primera pregunta del cuestionario realizado a los usuarios.	39
5.2. Gráfico de la segunda pregunta del cuestionario realizado a los usuarios.	40
5.3. Gráfico de la tercera pregunta del cuestionario realizado a los usuarios.	40
5.4. Gráfico de la cuarta pregunta del cuestionario realizado a los usuarios.	41

Índice de tablas

3.1. Características de los modelos GPT-3 disponibles.	16
3.2. Coste de los modelos GPT-3 disponibles.	16
4.1. Inversión realizada en la elaboración del conjunto de datos de entrenamiento.	29
5.1. Similitud entre las respuestas generadas con diferentes valores de temperatura y la respuesta original de entrenamiento.	35

Lista de acrónimos y abreviaturas

AA	Aprendizaje Automático
API	<i>Application Programming Interface</i>
CSV	<i>Comma Separated Values</i>
GPT	<i>Generative Pre-trained Transformer</i>
GPU	<i>Graphics Processing Unit</i>
GUI	<i>Graphical User Interface</i>
HTTP	<i>Hypertext Transfer Protocol</i>
IAG	Inteligencia Artificial General
IA	Inteligencia Artificial
JSON	<i>JavaScript Object Notation</i>
LSTM	<i>Long Short-Term Memory</i>
ML	<i>Machine Learning</i>
NLTK	<i>Natural Language Toolkit</i>
TFG	Trabajo Fin de Grado

Capítulo 1

Introducción

Este capítulo constituye un preámbulo al desarrollo de un chatbot en Telegram especializado en equipamiento biomédico y basado en el entrenamiento específico del modelo GPT-3. A continuación, se definen los objetivos de este Trabajo Fin de Grado (TFG) y se presenta la estructura que sigue esta memoria.

1.1. Contexto y motivación

En los últimos años se ha visto un avance significativo en el campo de la Inteligencia Artificial (IA), en particular en lo que respecta a los modelos de procesamiento del lenguaje natural. Estos modelos han experimentado una evolución exponencial, demostrando una extraordinaria capacidad para la comprensión y la generación de texto en lenguaje natural. Los avances en los métodos de aprendizaje automático, especialmente en el campo del aprendizaje profundo, han permitido que estos modelos comprendan y reproduzcan textos coherentes similares a los escritos por humanos.

Aprovechando estos avances, el objetivo principal de este TFG es desarrollar un chatbot que, a través de la plataforma de mensajería Telegram, sea capaz de responder preguntas relacionadas con equipamiento biomédico. Para ello, se propone realizar *fine-tuning* o entrenamiento específico del modelo GPT-3 (del inglés, *Generative Pre-trained Transformer 3*) Davinci de OpenAI. Este tipo de modelos de lenguaje han sido preentrenados con una cantidad masiva de datos, otorgándoles capacidad para localizar, aprender y seguir patrones en el texto que le permite generar contenido en base en las entradas que reciba.

Otro campo en constante desarrollo es el campo de la ingeniería biomédica. Esta disciplina

pretende investigar nuevas tecnologías que, aplicadas al ámbito sanitario, mejoren la calidad de vida de las personas. En este contexto, el equipamiento biomédico destaca por su contribución en la prevención, diagnóstico, tratamiento y seguimiento de enfermedades, favoreciendo el bienestar de la población. En vista de los avances realizados en este campo, a medida que aumenta la complejidad de los diferentes equipos, se vuelve de vital importancia la existencia de una fuente de información precisa y actualizada que permita acceder de manera sencilla a contenidos relacionados con equipos biomédicos.

Apreciando las notables mejoras que la IA ha proporcionado al ámbito biomédico, creando nuevas herramientas y tecnologías especializadas en mejorar el estado de salud de la sociedad, se puede explorar y aprovechar su potencial para comprender la complejidad y diversidad del equipamiento biomédico existente.

La elección del tema del presente TFG responde a la necesidad de una herramienta que permita obtener información específica y veraz sobre equipos biomédicos. Para el estudiantado que se está formando en la materia (más concretamente, al estudiantado matriculado en la asignatura Equipamiento Biomédico del grado en Ingeniería Biomédica de la Universidad Rey Juan Carlos), contar con un asistente virtual capacitado para brindar respuestas precisas sobre la materia puede suponer una gran aportación para complementar el método de aprendizaje.

Con la realización de este TFG se pretende desarrollar un chatbot especializado en equipamiento biomédico que aumente la accesibilidad a la información relacionada con este ámbito. Se busca que el chatbot sea capaz de proporcionar respuestas concretas utilizando las capacidades lingüísticas del modelo GPT-3. Así, se pretende investigar y evaluar las capacidades de los modelos de lenguaje natural en el campo biomédico y su utilidad para desarrollar herramientas que favorezcan el aprendizaje.

1.2. Objetivos

El objetivo principal del presente TFG es entrenar un modelo de procesamiento de lenguaje natural capacitándolo para proporcionar respuestas cortas y precisas a cuestiones relacionadas con equipos médicos a través de Telegram. Esto permitirá a los usuarios mantener una conversación escrita con una IA, integrada en un bot de Telegram, que ha sido entrenada con la finalidad de dar respuesta a cuestiones relacionadas con Tomografía Computarizada, Resonancia Magnética, Rayos X, Ultrasonidos y Medicina Nuclear. Este objetivo tiene un carácter multidisciplinar, pudiendo dividirse en los siguientes objetivos específicos que se redactan a continuación:

1. Compresión de los conceptos del dominio de conocimiento, profundizando en temas de *Deep Learning*.
2. Investigación y selección adecuada de la infraestructura de modelo de lenguaje basada en *transformer*.
3. Recopilación de información relevante sobre diversos equipos médicos.
4. Elaboración automática de preguntas y respuestas sobre los datos recogidos.
5. Ajuste del formato del conjunto de datos destinados al entrenamiento específico de modelos de lenguaje natural.
6. Empleo de APIs para establecer la comunicación e interacción entre diferentes sistemas y aplicaciones, y de éstas con el usuario final.

1.3. Estructura de la memoria

A continuación, se describe brevemente el contenido de cada capítulo de la memoria:

- **Capítulo 1: Introducción y objetivos.** En primer lugar, se contextualiza y justifica la elección del tema de estudio. En segundo lugar, se presentan los objetivos abordados a lo largo del TFG. Por último, se proporciona una breve descripción de la estructura empleada en la elaboración de la memoria.
- **Capítulo 2: Fundamentos teóricos.** Se proporcionará información relacionada con los métodos de Aprendizaje Automático, especialmente en el campo del *Deep Learning* permitiendo una mejor comprensión del contenido del presente TFG. Finalmente, se discuten varios trabajos previos a este TFG en los que también se utilizan *transformers* como arquitectura para desarrollar modelos de lenguaje.
- **Capítulo 3: Diseño del proyecto.** Se familiariza al lector con los recursos utilizados para el desarrollo del chatbot realizando una presentación y justificación de la elección de los mismos.
- **Capítulo 4: Desarrollo del proyecto.** Se realiza una explicación concreta sobre el procedimiento utilizado para implementar la solución que permita alcanzar los objetivos propuestos.

- **Capítulo 5: Experimentos y resultados.** En este capítulo se presenta el componente experimental y los resultados obtenidos tras evaluar el rendimiento del modelo de lenguaje. Se realiza una evaluación tanto interna como externa con la participación y retroalimentación de usuarios reales.
- **Capítulo 6: Discusión.** Se analizan los resultados recopilados para medir el desempeño del chatbot, proporcionando información valiosa que se debe tener en cuenta para mejorar y perfeccionarlo en etapas posteriores.
- **Capítulo 7: Conclusiones y líneas futuras.** Se resumen el trabajo y las conclusiones extraídas. Además, se presentan posibles líneas de trabajo futuras que podrían llevar al modelo a su máximo rendimiento.

Capítulo 2

Fundamentos teóricos

En este capítulo se introducen una serie de conceptos relevantes asociados al Procesamiento del Lenguaje Natural y al Aprendizaje Automático, los cuales facilitarán la comprensión de las bases teóricas de este TFG.

2.1. Conceptos previos

Los chatbots o asistentes conversacionales son programas informáticos preparados para participar en conversaciones escritas o grabadas con seres humanos. Son diversas las aplicaciones o funciones que pueden desempeñar, sin embargo, el uso principal ha sido enfocado al servicio al cliente.

Para poder establecer comunicación entre el chatbot y el usuario que desea utilizarlo, es imprescindible que el agente conversacional sea capaz de comprender y generar lenguaje humano. En consecuencia, el chatbot se fundamenta en el Procesamiento del Lenguaje Natural (NLP, por sus siglas en inglés, *Natural Language Processing*). NLP hace referencia a la especialidad de la IA destinada a proporcionar a los ordenadores la capacidad de comprensión de textos, siguiendo el estilo de comunicación de los seres humanos. La esencia del funcionamiento radica en la identificación de patrones contextuales en las entradas de texto que proporciona el usuario. De esta manera, se trata de entender su significado completo, además de la intención y el sentimiento del escritor, con el propósito de ofrecer una respuesta apropiada y útil [6].

El campo de la IA destaca por su amplio alcance, dentro del cual se puede encontrar el Aprendizaje Automático (AA) o *Machine Learning* (ML). Las técnicas de ML permiten a los algoritmos identificar patrones complejos entre gran cantidad de datos, infiriendo así sus pro-

pías reglas para detectar patrones similares en nuevos conjuntos de datos. Dentro de esta rama surge el enfoque *Deep Learning* o Aprendizaje Profundo, un modelo de aprendizaje por capas cuya base son las redes neuronales artificiales. Estos sistemas imitan el funcionamiento de las neuronas y mecanismos básicos del cerebro, intentando aproximarse a la forma que tienen los humanos de aprender [7].

Dentro del *Deep Learning*, surgieron arquitecturas como las Redes Neuronales Recurrentes (RNN, del inglés, *Recurrent Neural Nets*) adecuadas para procesar datos secuenciales o secuencias temporales. Existen diferentes tipos de RNN, pero entre sus aplicaciones se puede distinguir la traducción de texto, predicción de ventas, desarrollo de asistentes virtuales o chatbots inteligentes [8]. Esto se debe a su estructura recurrente que les permite obtener memoria artificial, ya que la información fluye de una capa a otra a través de un “estado oculto” que almacena información sobre los estados anteriores e influye en la salida [9]. Sin embargo, la memoria solo funciona para secuencias relativamente cortas, impidiendo tener memoria a largo plazo. A esto se suma el problema del gradiente. El gradiente es una medida de la pendiente de una función, que en el contexto de redes neuronales hace referencia a cómo cambian los pesos y los sesgos de la red en función del error calculado durante su entrenamiento. En las RNN, los gradientes pueden volverse muy pequeños, llegando a tener velocidades de aprendizaje muy lentas que impliquen detener el entrenamiento; o gradientes excesivamente grandes, que llevan a un aprendizaje rápido, pero con el riesgo de que los pesos se alejen de la solución óptima [10]. Para solucionar estos problemas aparecieron las redes LSTM (por sus siglas en inglés, *Long Short-Term Memory*).

En este contexto, aparece en 2017 el concepto *transformer*, una nueva propuesta de arquitectura de redes neuronales superando a las redes LSTM. A diferencia de los modelos secuenciales dominantes, que utilizaban un mecanismo de atención para conectar el codificador y el decodificador, el *transformer* está basado únicamente en mecanismos de atención. Así se especifica en el artículo “*Attention is all you need*” de Vaswani et al [1], donde se presenta esta arquitectura (véase Figura 2.1).

La arquitectura que sigue un modelo *transformer* consta de dos grandes bloques, uno específico para la codificación y otro para la decodificación de los datos. Para comprender el mecanismo de un *transformer* es necesario aclarar conceptos como *tokens* y *embeddings*.

En primer lugar, resulta imprescindible disponer de un método de representación de palabras con la finalidad de que los modelos tengan capacidad de procesarlas, puesto que sólo procesan números. Los *transformers* son modelos de lenguaje que utilizan técnicas de *Text Tokenization*. La tokenización es el proceso de dividir un fragmento de texto en unidades más pequeñas deno-

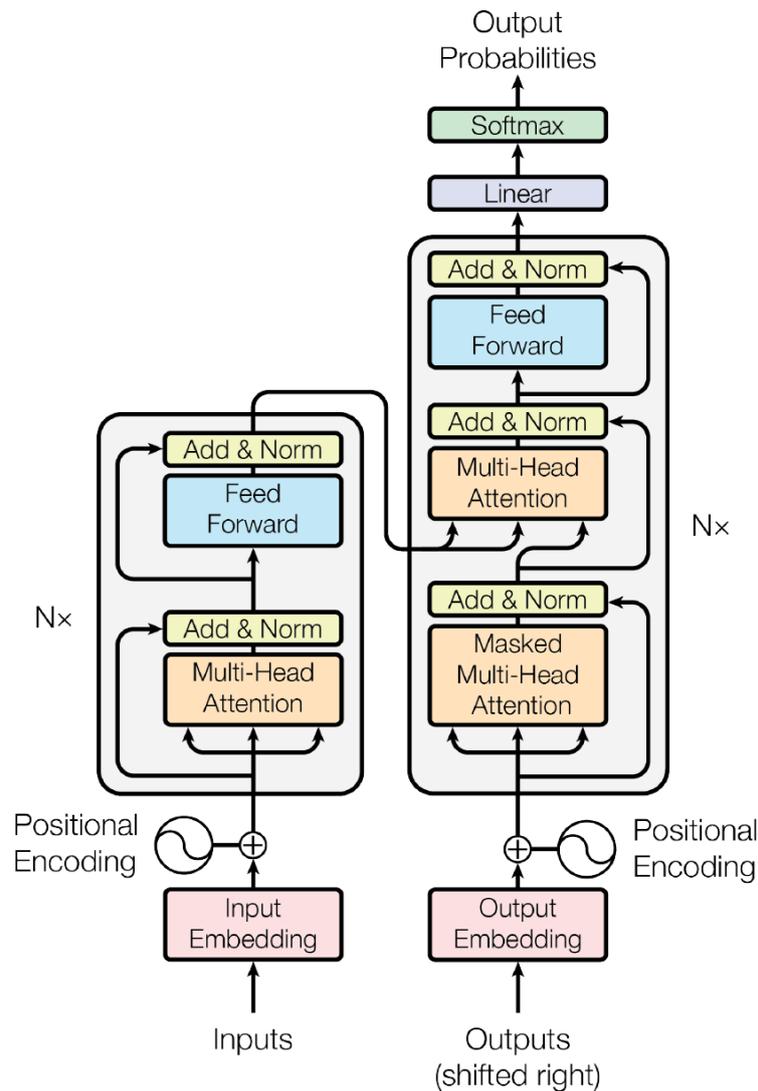


Figura 2.1: Arquitectura del modelo *transformer*. Figura extraída de [1].

minadas *tokens*. Los *tokens* pueden ser palabras completas, subpalabras o caracteres [11]. Los tokenizadores tienen un archivo de vocabulario donde se almacena una lista de *tokens* predefinidos y sus correspondientes índices numéricos que permiten representar ese *token* de manera única. Dada una frase, el tokenizador busca en el corpus de su vocabulario los *tokens* y asigna el identificador asociado (véase Figura 2.2).

Tras la tokenización de la frase, se utiliza *Word Embedding* o “incrustación de palabras” para representar palabras mediante vectores numéricos reales. De esta manera, se reemplaza el identificador de cada palabra por un vector que permite identificar las palabras semánticamente similares por estar situadas en lugares cercanos en el espacio (véase Figura 2.3) [12].

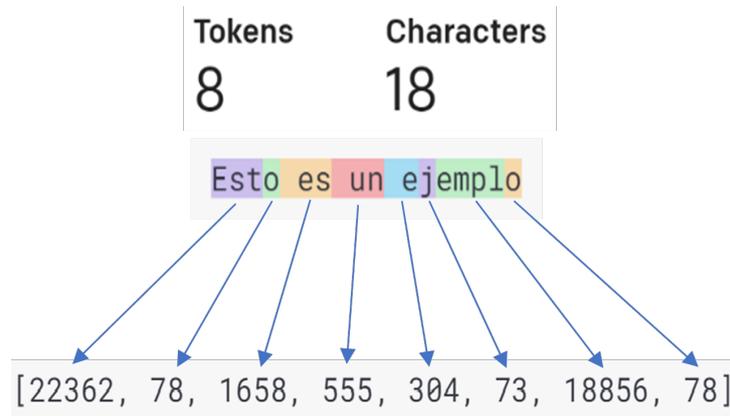


Figura 2.2: Ejemplo tokenización de una frase. Separación en *tokens* y asignación de identificadores. Figura extraída/modificada de [2].

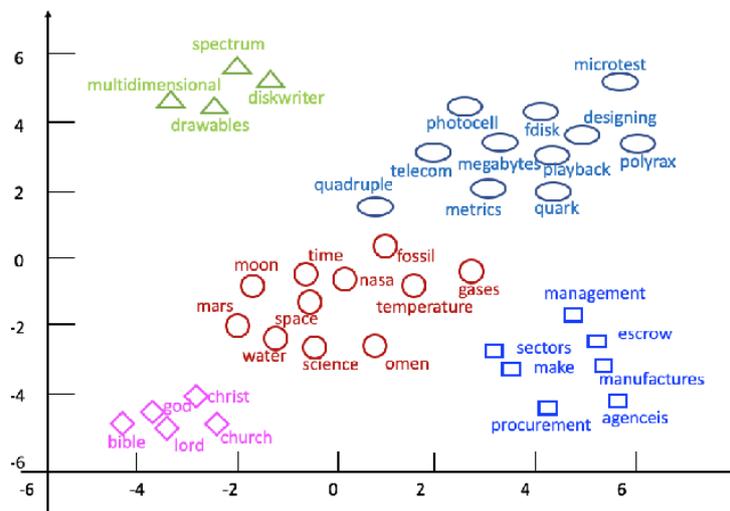


Figura 2.3: Proyección en 2D de Word Embeddings. Se muestran cinco clusters diferentes. Figura extraída de [3].

Posteriormente, se captura la información posicional de cada palabra, para poder reconocer qué posición ocupa cada palabra en la frase introducida, puesto que una alteración en el orden de las palabras puede suponer una alteración en el significado de la frase de entrada. Esta información numérica entra al codificador, que es en realidad una pila de seis codificadores, cada uno dividido en dos capas [13]:

- Capa de *self-attention*: encargada de asignar importancia a las palabras en función de su relevancia en la frase.
- Capa de *feed forward*: proporciona capacidad de identificar relaciones complejas y extraer características significativas de los datos.

El decodificador también está formado por una pila de seis decodificadores con ambas capas; no obstante, entre ellas existe una capa de atención que ayuda al decodificador a centrarse en las partes relevantes de la secuencia de entrada [13].

Siguiendo esta arquitectura, los *transformer* ofrecen la capacidad de entrenar en paralelo aprovechando la GPU (del inglés, *Graphics Processing Unit*), reduciendo así tiempos y costes de entrenamiento. En consecuencia, los *transformers* se aplicaron a diversas áreas de programación convirtiéndose en el modelo estándar en tareas de NLP. Surgieron modelos que fueron entrenados con conjuntos de datos cada vez más grandes, permitiendo realizar diversas tareas entre las cuales se encuentran la traducción de idiomas, clasificación y resumen de textos, así como tareas de preguntas-respuestas como la que se aborda a lo largo del presente TFG [14].

2.2. Estado del arte

En los años previos al desarrollo de este TFG se han realizado varios estudios empleando diversas arquitecturas que han permitido el desarrollo de modelos de lenguaje con diferentes aplicaciones. Con el tiempo han aparecido varios sistemas de chatbot centrados en temas de salud, abordando cuestiones de atención psicológica, diagnóstico médico, seguimiento de enfermedades crónicas o educación sanitaria, entre otros.

Entre los trabajos previos se consideran los más relevantes y recientes que han contribuido al avance de los modelos de lenguaje, observando especialmente el proceso de recopilación y preparación del conjunto de datos, así como la arquitectura elegida para el modelo. La elaboración del conjunto de datos es un paso fundamental para conseguir desarrollar un modelo de lenguaje exitoso, ya que permitirá optimizar el modelo utilizando datos de alta calidad. Por ello, se deben elegir fuentes que proporcionen información veraz y diversa sobre la temática elegida. De la misma manera, se observará la arquitectura empleada para desarrollar los modelos, que ofrecerán mejores o peores prestaciones en función de su configuración.

En [15] se propone un método para utilizar modelos de lenguaje en el ámbito de la biomedicina. Concretamente, se plantea incorporar BERT (del inglés, *Bidirectional Encoder Representation from Transformers*), un método de NLP desarrollado por Google en 2018 y basado en los modelos de sólo codificador de la arquitectura *transformer*. BERT es un modelo diseñado con un preentrenamiento bidireccional, a diferencia de los modelos anteriores, que entrenaban de manera unidireccional. Esto le permite procesar una palabra capturando información tanto de las palabras que la preceden como de las que la siguen, atendiendo así al contexto general de la frase [16].

En el estudio se seleccionaron cuatro modelos BERT preentrenados:

- RoBERTa: modelo robusto con un preentrenamiento más prolongado y con más datos que BERT.
- XML-R: modelo extensión de RoBERTa con capacidad para comprender y procesar texto multilingüe.
- BERT Large: modelo con mejor rendimiento que BERT debido a una mayor complejidad en su arquitectura y un mayor coste asociado a su uso.
- BioBert: modelo de lenguaje preentrenado con datos del ámbito biomédico.

Estos modelos permiten realizar un entrenamiento específico para mejorar su rendimiento y aplicarlo en dominios específicos, en este caso, en el ámbito de la biomedicina. Para llevar esto a cabo se utilizó como fuente de información 1,100 resúmenes de artículos publicados en PubMed relacionados con temas médicos como enfermedades, genética, microorganismos y técnicas generales para el tratamiento de enfermedades. Además, se utilizaron datos de SQuAD v2.0, una base de datos con un amplio conjunto de pares pregunta-respuesta sobre ciencia, historia, tecnología y deportes.

Tras el entrenamiento específico de los cuatro modelos se concluyó que el modelo BERT Large con su compleja arquitectura obtiene los mejores resultados siendo las respuestas generadas más similares a la respuesta real, seguido de BioBERT, RoBERTa y XML-R en último lugar. En este caso, ninguno de los modelos disponía de una plataforma interactiva para los usuarios.

Por otro lado, en [17] se explora un nuevo modelo de lenguaje, SciFive. Se utilizó un modelo *transformer* de transferencia de texto a texto (T5), un modelo preentrenado con capacidad para realizar entrenamiento específico y adaptar este modelo a tareas relacionadas con la literatura biomédica. El modelo T5 es muy similar al modelo codificador-decodificador explicado en la Sección 2.1, con la peculiaridad de que este último se enfoca en la traducción automática y en la generación de texto, mientras que el modelo T5 se centra de manera general en la generación de texto, abordando tanto traducciones y resúmenes de documentos como elaboraciones de respuestas.

A continuación, se enuncian los conjuntos de datos utilizados para el entrenamiento completo de SciFive:

- **Preentrenamiento:** SciFive utiliza el modelo T5 que ha sido preentrenado con el Colossal Clean Crawled Corpus (C4), un corpus que es la versión masiva y limpia de Common Craw Corpus, que contiene datos en bruto de páginas web, extractos de metadata y de texto [18].
- **Entrenamiento específico:** en esta ocasión se utilizan resúmenes de artículos biomédicos publicados en PubMed, así como artículos biomédicos completos de PubMed Central para mejorar el aprendizaje.

Se determina que Scifive aporta respuestas científicamente correctas, suponiendo un avance significativo. Se comparan las respuestas generadas por SciFive con las generadas por BioBert determinando que SciFive proporciona respuestas más claras y completas que BioBERT.

Finalmente, en [19] se presenta BiomedGPT, un modelo entrenado con grandes conjuntos de datos para procesar entradas multimodales que permitan llevar a cabo gran cantidad de tareas relacionadas con el área de la biomedicina. BiomedGPT sigue la arquitectura basada en *transformer* con codificador-decodificador. Concretamente, es una combinación de un codificador BERT y un decodificador GPT (por sus siglas en inglés, *Generative Pre-trained Transformer*). Esto se combina con estructuras de redes neuronales convolucionales (CNN, del inglés, Convolutional Neural Network) para permitir que las entradas también puedan ser imágenes.

El proceso de entrenamiento de BiomedGTP se divide en dos fases:

- **Preentrenamiento:** se preentrena al modelo para que realice tareas de texto basadas en preguntas y respuestas, y tareas de visión centradas en la detección de objetos. También se incluyen tareas multimodales como la descripción de imágenes o la respuesta a preguntas visuales.
- **Entrenamiento específico:** se incluye la clasificación de imágenes según sean patológicas o no, así como resúmenes de texto o inferencia del lenguaje natural (determinar si un texto A implica el texto B).

Para entrenar al modelo en estas tareas se utilizaron 20 conjuntos de datos diferentes, procedentes de diversas fuentes, desde bases de datos de imagen médica, pasando por plataformas de atención médica en línea para recuperar interacciones médico-paciente, así como notas clínicas. Por esto, BiomedGPT abarca un amplio espectro de áreas médicas: coloproctología, dermatología, histopatología, tomografía de coherencia óptica, radiografía de tórax, ultrasonido mamario y tomografía computarizada abdominal.

BiomedGPT supone un enfoque revolucionario en el campo de la medicina por ser un modelo versátil con capacidad para abordar tareas multimodales de diversas áreas médicas. Sin embargo, los resultados obtenidos con este nuevo modelo de lenguaje son deficientes en el área de la sumariazación de texto donde tienen mejor rendimiento BioBERT y SciFive. Quizás esta diferencia en el rendimiento se debe a la menor cantidad de parámetros que definen BiomedGPT, 182 millones frente a los 400 millones de parámetros del modelo BioBERT. A esto se añade que el corpus utilizado para el entrenamiento de BiomedGPT es mucho menor en comparación con el utilizado en SciFive. Aun así, BiomedGPT obtuvo un alto rendimiento en tareas de clasificación y descripción de imágenes, así como en respuestas a preguntas visuales, superando a los modelos de última generación.

El presente TFG pretende realizar un entrenamiento especializado del modelo GPT, utilizado en BiomedGPT. Sin embargo, en este caso las entradas serán unimodales con capacidad única de procesar texto para abordar una tarea de preguntas y respuestas relacionadas con equipos médicos, incluyendo Tomografía Computarizada, Resonancia Magnética, Rayos X, Ultrasonidos y Medicina Nuclear.

Capítulo 3

Diseño del proyecto

Este capítulo tiene como finalidad presentar y justificar la elección de los recursos utilizados durante el presente TFG.

3.1. Planificación temporal

A continuación, en la Figura 3.1 se incluye un diagrama de Gantt con la previsión estimada de tiempo dedicado a cada tarea principal del proyecto.

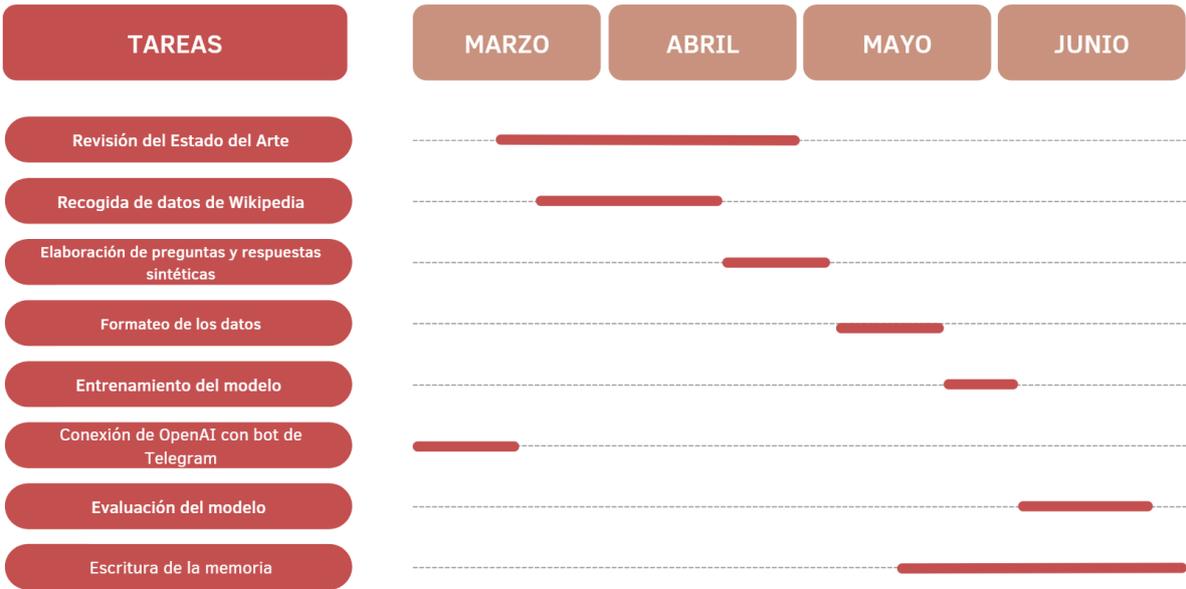


Figura 3.1: Diagrama de Gantt.

3.2. Diseño general de la propuesta

Para una mejor comprensión de los componentes que permiten el desarrollo del proyecto, se presenta el siguiente esquema que resume el proceso llevado a cabo para el desarrollo del chatbot (véase Figura 3.2).

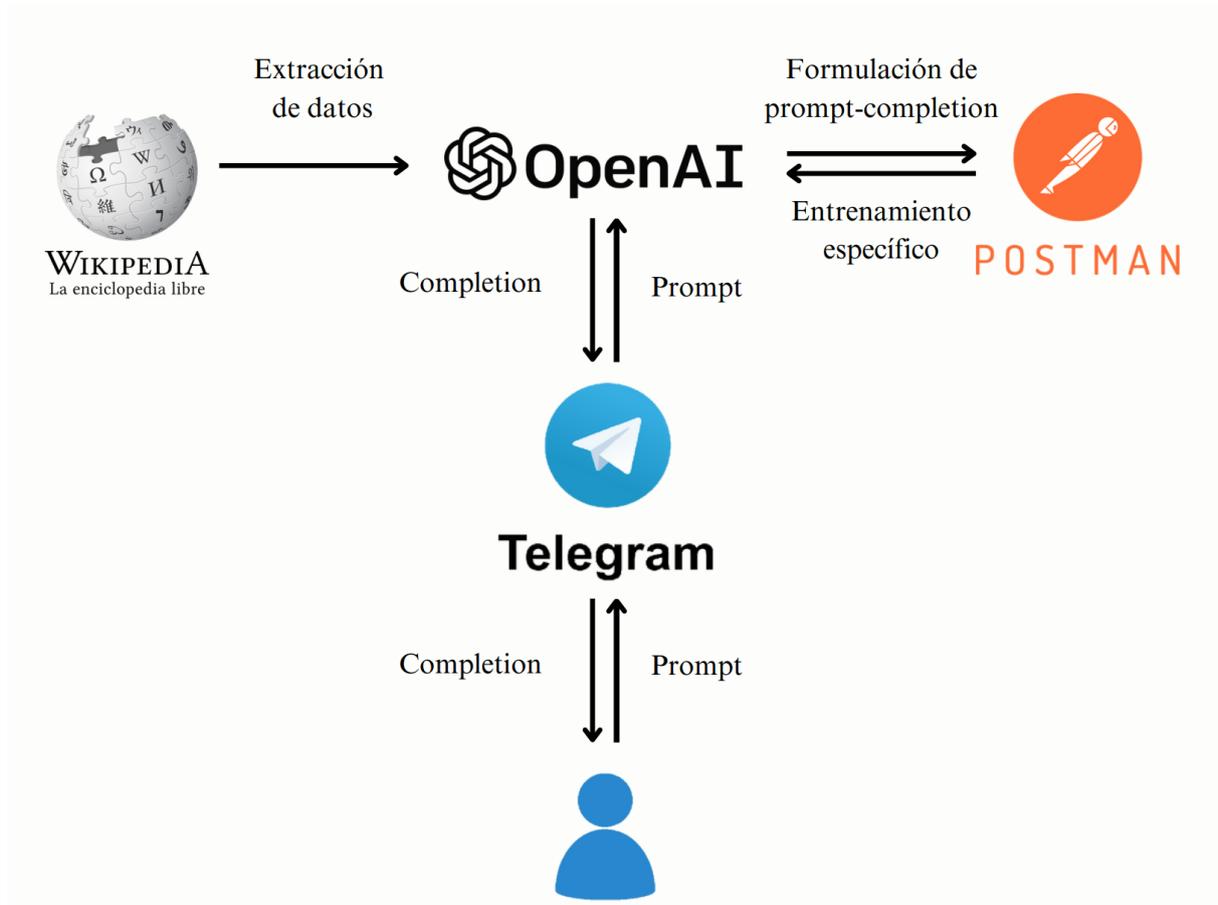


Figura 3.2: Esquema de componentes del proyecto.

3.3. OpenAI y los modelos GPT-3

OpenAI es una empresa de investigación y desarrollo de Inteligencia Artificial (IA) estadounidense fundada en 2015 por Elon Musk, principal accionista, junto con otros nombres como Sam Altman, Ilya Sutskever, Greg Brockman, Wojciech Zaremba y Andrej Karpathy [20].

La misión de la empresa es asegurar que la Inteligencia Artificial General (IAG) beneficie a la humanidad [21]. Musk caracteriza a la IA como la mayor amenaza existencial para los hu-

manos [22], por esta razón, el principal objetivo de OpenAI se basa en guiar su investigación de manera que la IAG pueda ayudar a “elevar a la humanidad al aumentar la abundancia, impulsar la economía global y ayudar en el descubrimiento de nuevos conocimientos científicos que cambien los límites de la posibilidad” [23].

Durante los trece años de trayectoria de la empresa se ha investigado y realizado publicaciones científicas sobre ética y seguridad de la IA, avances en transferencia de aprendizaje y en aprendizaje por refuerzo, así como el desarrollo de modelos de lenguaje avanzados [24]. Entre estos últimos se encuentran los modelos GPT, basados en la arquitectura *transformer* descrita en el Capítulo 2.

En este TFG se utiliza el modelo GPT-3, la tercera generación de los modelos de lenguaje GPT, con el objetivo de entender y generar lenguaje natural en el ámbito del equipamiento biomédico. La diferencia clave entre este modelo y los demás existentes es su tamaño. Durante su preentrenamiento se utilizaron 175 mil millones de parámetros, lo que supone un tamaño hasta 17 veces mayor que el modelo anterior, GPT-2 [25]. Por tanto, la tercera generación implica mayor tamaño y capacidad, suponiendo un mayor coste computacional, pero mejorando el rendimiento del modelo en las tareas a desempeñar. Actualmente, existen versiones superiores como GPT-3.5, GPT-3.5 turbo o GPT-4, pero durante la realización del presente TFG aún no se habían terminado de desarrollar o bien no existían claves API (del inglés, *Application Programming Interface*) que permitieran la interacción con los modelos.

Como se define en [25] el entrenamiento de los modelos GPT consta de dos fases:

- **Preentrenamiento:** primera etapa donde los investigadores utilizan una combinación de datos no supervisados procedentes de páginas de internet, de una colección inmensa de libros y de todo Wikipedia.
- **Entrenamiento específico:** segunda etapa que supone la actualización de los pesos del modelo preentrenado con un nuevo conjunto de datos. Se ajusta el modelo a tareas específicas mediante un entrenamiento supervisado con conjuntos de datos más pequeños y específicos.

A través de la API de OpenAI, la empresa ofrece la oportunidad de comunicarse con ella llevando a cabo tareas relacionadas con NLP. Creando una cuenta en OpenAI se pueden obtener claves API con las que interactuar con los modelos de lenguaje GPT-3 de la plataforma. Entre ellos se pueden encontrar los modelos *instructGPT* y los modelos base (véase Tabla 3.1).

Modelo	Descripción	Tokens máximos	Datos actualizados hasta
text-davinci-003	Modelo instructGPT	4,097	octubre de 2019
text-curie-001	Modelo instructGPT	2,049	octubre de 2019
text-babbage-001	Modelo instructGPT	2,049	octubre de 2019
text-ada-001	Modelo instructGPT	2,049	octubre de 2019
Davinci	Modelo base	2,049	junio de 2021
Curie	Modelo base	2,049	octubre de 2019
Babbage	Modelo base	2,049	octubre de 2019
Ada	Modelo base	2,049	octubre de 2019

Tabla 3.1: Características de los modelos GPT-3 disponibles.

Los modelos *instruct* son modelos de lenguaje que han sido entrenados con técnicas específicas para desarrollar la capacidad de procesar texto prestando especial atención a las instrucciones que proporciona el usuario. Estos modelos demuestran un mejor desempeño ya que están diseñados para entender la intención de las peticiones, por ello están especializados en generar texto respondiendo a instrucciones concretas y no ofrecen posibilidad de realizar entrenamiento específico [26]. Para esto ya existen los modelos base, que serán los que permitan el ajuste sobre un campo concreto para realizar tareas tanto de clasificación como de preguntas-respuesta.

Como se puede observar, en función de las capacidades y precios existen diferentes versiones dentro de los modelos *instruct* y de los modelos base. Interactuar con los modelos de lenguaje de OpenAI no es gratuito, la plataforma establece una forma de pago basada en el consumo de cada 1,000 tokens. Siguiendo la documentación de OpenAI, los modelos más competentes son los más caros (véase Tabla 3.2), ya que ofrecen una mayor calidad en las tareas desempeñadas. En base a esto, los modelos seleccionados para desarrollar el presente TFG serán “text-davinci-003” y “Davinci”.

Modelo	Precio por uso del modelo (\$/1,000 tokens)	Precio por entrenamiento (\$/1,000 tokens)	Precio por uso del modelo entrenado (\$/1,000 tokens)
text-davinci-003	0.0200	No ofrece entrenamiento	
text-curie-001	0.0020	No ofrece entrenamiento	
text-babbage-001	0.0005	No ofrece entrenamiento	
text-ada-001	0.0004	No ofrece entrenamiento	
Davinci	0.0200	0.0300	0.1200
Curie	0.0020	0.0030	0.0120
Babbage	0.0005	0.0006	0.0024
Ada	0.0004	0.0004	0.0016

Tabla 3.2: Coste de los modelos GPT-3 disponibles.

Para realizar el entrenamiento específico de los modelos base se deben preparar y cargar los datos que se utilizarán en el entrenamiento. Siguiendo la documentación en [27] los modelos de lenguaje de OpenAI se entrenan con un archivo en formato JSONL (JSON Lines), también denominado JSON (JavaScript Object Notation) delimitado por saltos de línea [28]. En la Figura 3.3 se observa la estructura que siguen este tipo de archivos de texto, donde cada línea es un valor JSON válido con pares clave-valor. El objeto JSON se encuentra encerrado entre llaves y con una coma separando cada par clave-valor. El valor de la clave *prompt* es una cadena de caracteres (string) que contiene un ejemplo de pregunta que podría ser formulada por el usuario al modelo de lenguaje, mientras que el valor de la clave *completion* es otro string con la respuesta que se espera obtener a dicha pregunta. Con esto se proporcionan al modelo ejemplos de conversaciones entre el usuario y el modelo de lenguaje que le permiten forjar su comportamiento para interacciones posteriores.



Figura 3.3: Ejemplo de JSONL.

De esta forma, la API de OpenAI permite acceder a sus modelos de lenguaje para construir nuevos modelos especializados, a través del entrenamiento específico, sin necesidad de definir la estructura de la red neuronal o entrenarla desde cero, facilitando así la creación de chatbots.

3.4. Python

Python es un lenguaje de programación *Open-Source* interactivo y orientado a objetos que incorpora módulos, excepciones, tipos de datos dinámicos de muy alto nivel y clases. Además, se puede utilizar como interfaz programable para desarrollar muchas aplicaciones [29].

Para la realización de este proyecto se propone utilizar Python (versión 3.9.13) por su amplia gama de bibliotecas y librerías y su sintaxis clara. Durante el desarrollo del presente TFG se

han utilizado las siguientes librerías, mejor explicadas a lo largo de la memoria: *wikipedia*, *wikipediaapi*, *OpenAI*, *re*, *transformers*, *numpy*, *nltk.tokenize*, *pandas*, *csv*, *json*, *telebot*, *spacy*.

3.5. Wikipedia

Wikipedia es una enciclopedia en línea de acceso gratuito donde millones de personas han ido añadiendo contenido sobre diversas áreas de conocimiento. Desde que Jimmy Wales impulsó su lanzamiento en 2001, son muchos los profesionales que han contribuido al tráfico de información favoreciendo la evolución de esta enciclopedia que se ha extendido por todo el mundo [30].

En la actualidad, la versión en español de la plataforma cuenta con una significativa cantidad de publicaciones, alcanzando aproximadamente los dos millones de artículos [31]. Mientras tanto, en la versión en inglés se encuentra disponible un mayor número de publicaciones, superando los seis millones [32]. En este TFG se plantea utilizar la versión en inglés como método de adquisición de información, en vista de la vasta cantidad de publicaciones realizadas en dicho idioma.

El amplio contenido de Wikipedia está organizado y estructurado mediante diferentes categorías. Estas se ordenan de manera jerárquica, subdividiendo categorías principales más generales en subcategorías más especializadas. Por ejemplo, una categoría principal como “*Ultrasound*” puede tener subcategorías tal y como se muestra en la Figura 3.4 (a), para simplificar el acceso a la información relacionada con una temática concreta.

Tras ingresar a la categoría o subcategoría escogida, se presentan en orden alfabético todas las páginas existentes hasta la fecha. Se observa que cada página publicada sigue una estructura basada en un índice que organiza la presentación de la información, facilitando la navegación por la página (véase Figura 3.4 (b)).

Considerando la buena estructura, el práctico acceso a la información de una determinada temática y la extensa cantidad de publicaciones disponible en la versión en inglés, se utiliza Wikipedia como recurso para recabar información relacionada con equipos médicos, tales como:

- Tomografía Computarizada
- Resonancia Magnética
- Rayos X

- Ultrasonidos
- Medicina Nuclear

Category:Ultrasound

Category [Talk](#)

From Wikipedia, the free encyclopedia

The main article for this category is *[Ultrasound](#)*.

Subcategories

This category has the following 3 subcategories, out of 3 total.

M

► [Medical ultrasonography](#) (1 C, 85 P)

S

► [Sonar](#) (6 C, 38 P)

U

► [Ultrasonographers](#) (3 P)

(a)

☰ Medical ultrasound

Contents

(Top)

[Types](#)

[Advantages and drawbacks](#)

▼ [Uses](#)

[Anesthesiology](#)

[Angiology \(vascular\)](#)

[Cardiology \(heart\)](#)

[Emergency medicine](#)

[Gastroenterology/Colorectal surgery](#)

[Gynecology and obstetrics](#)

[Hemodynamics \(blood circulation\)](#)

(b)

Figura 3.4: Visualización de la estructura de Wikipedia: (a) subcategorías de la categoría principal “*Ultrasound*”; (b) primeras nueve secciones del índice de la página “*Medical Ultrasound*” perteneciente a la subcategoría “*Medical Ultrasonography*”. Figura extraída de [4] [5].

3.6. Postman

Postman es una plataforma gratuita, en el caso de trabajar de manera individual, que posibilita y agiliza tanto la creación como el uso de APIs [33]. Entre sus funciones se encuentra la posibilidad de realizar solicitudes HTTP a un recurso específico.

Las solicitudes HTTP, son métodos que se utilizan en el protocolo HTTP (Hypertext Transfer Protocol) para establecer comunicación entre el cliente y un servidor web. En el caso de este proyecto, es necesario establecer conexión con el servidor de OpenAI, realizando el siguiente tipo de solicitudes [34]:

- **POST:** este método envía información al servidor en el cuerpo de la solicitud. Es comúnmente utilizado para enviar datos en formularios, formato JSON o texto sin formato (raw), entre otros. Una vez los datos son procesados, el servidor puede enviar una respuesta.

- GET: se realiza una petición al servidor para recuperar información, por tanto, se puede retornar una respuesta en formato JSON u otros formatos adicionales.

La principal diferencia entre emplear lenguaje específico de programación y utilizar Postman en la realización de estas peticiones HTTP es que Postman proporciona una interfaz intuitiva donde enviar dichas peticiones a APIs específicas de manera sencilla. Permite establecer los parámetros necesarios, así como adjuntar datos en el cuerpo de la solicitud y visualizar la respuesta a los métodos HTTP de forma clara.

Por tanto, con el propósito de facilitar la comunicación cliente-servidor se determina utilizar esta plataforma como recurso para realizar el entrenamiento específico del modelo de lenguaje de OpenAI.

3.7. Telegram

Una interfaz gráfica de usuario (GUI, por sus siglas en inglés, *Graphical User Interface*) permite a los usuarios interactuar con una máquina empleando un conjunto de objetos e imágenes para representar la información disponible durante la conexión. El principal objetivo es crear un entorno visual que permita la comunicación de una manera sencilla e intuitiva [35]. En este sentido se puede aprovechar la GUI que utiliza Telegram para establecer la comunicación entre el usuario y el modelo reentrenado en OpenAI.

Telegram es una plataforma de mensajería gratuita lanzada en 2013, donde la GUI está diseñada con un enfoque sencillo para su uso intuitivo [36]. Además, cuenta con una API especializada en la creación de bots, permitiendo que los programas usen Telegram como una interfaz [37]. Esto la convierte en una plataforma atractiva para establecer la comunicación entre el modelo de lenguaje y los usuarios porque evita la necesidad de entender lenguaje de programación que permite utilizar el modelo.

Al complementar el uso de la API de OpenAI con la API de Telegram, se puede integrar el modelo de lenguaje previamente entrenado en equipamiento biomédico en el bot para crear un chatbot en Telegram, creando una experiencia eficaz similar a la interacción con otro ser humano.

Capítulo 4

Desarrollo del proyecto

A continuación, se explica el procedimiento seguido para entrenar el modelo, proporcionándole capacidad para responder preguntas relacionadas con equipamiento biomédico.

4.1. Recopilación y preparación de datos

4.1.1. Extracción de datos de Wikipedia

Tal y como se especificó en el Capítulo 3, los datos relacionados con equipamiento biomédico requeridos para el entrenamiento específico del modelo de lenguaje son extraídos de la enciclopedia Wikipedia.

Existen dos bibliotecas en Python, *wikipedia* y *wikipediaapi*, que proporcionan una interfaz para comunicarse con la API de Wikipedia, en el caso de este TFG, con la versión en inglés. Se recopilan todas las páginas de interés accediendo a la categoría o subcategoría que contiene la información sobre el equipamiento biomédico requerido. En total, se ha recopilado contenido de 333 páginas:

- Tomografía Computarizada: 35 páginas.
- Resonancia Magnética: 108 páginas.
- Rayos X: 70 páginas.
- Ultrasonidos: 85 páginas.

- Medicina Nuclear: 35 páginas.

En primer lugar, se guarda y procesa la primera sección de todas las páginas donde aparece resumido el contenido de la misma, etiquetando ese contenido como *Summary*. El resto del contenido se identifica utilizando expresiones regulares de la librería *re* de Python, que permite localizar patrones en el texto para identificar los títulos de las secciones de cada página web.

Es fundamental que las diferentes secciones recopiladas tengan una extensión considerable que posibilite la extracción de contenido de ellas en la etapa de formulación de preguntas y respuestas sintéticas de la sección 4.1.2. Por tanto, se debe contar el número de tokens de cada sección y comprobar que el valor es significativo. Para llevar esto a cabo, es imprescindible utilizar la biblioteca *Natural Language Toolkit* (NLTK) para dividir los párrafos identificados en oraciones. Importando la clase *GPT2TokenizerFast* del módulo *transformers* se puede utilizar el modelo de lenguaje GPT-2 de OpenAI para tokenizar estas oraciones. Posteriormente, con ayuda de la biblioteca *numpy* se calcula la longitud del texto tokenizado para obtener el recuento del número de tokens utilizado en cada sección. Esto permitirá eliminar las secciones cuyos contenidos contienen un valor de tokens igual o inferior a 40, considerando que estas secciones no son lo suficientemente detalladas ni relevantes por ser poco extensas.

En las páginas de Wikipedia existen algunas secciones que carecen de relevancia en cuanto a su contenido y otras que, por ser imágenes, no se pueden incluir en los datos de entrenamiento porque los modelos de lenguaje GPT-3 sólo poseen capacidad para procesar texto. Debido a esto, se ha considerado la eliminación de las siguientes secciones del índice: *see also*, *references*, *external links*, *further reading*, *footnotes*, *bibliography*, *sources*, *citations*, *literature*, *footnotes*, *notes and references*, *photo gallery*, *works cited*, *photos*, *gallery*, *notes*, *references and sources*, *references and notes*.

Gracias a la biblioteca *pandas* se pueden manipular los datos obtenidos siguiendo la estructura de un *DataFrame*. Se trata de una tabla bidimensional en la que los datos están estructurados en filas y columnas (véase Figura 4.1). Se crean cuatro columnas siguiendo el orden descrito a continuación:

- ***Title***: corresponde al título de la página de Wikipedia.
- ***Heading***: incluye el nombre de las secciones de las páginas de Wikipedia.
- ***Content***: almacena el contenido asociado a la sección y página correspondiente.
- ***Tokens***: acumula el número de tokens que tiene el contenido de cada sección.

	title	heading	content	tokens
0	Medical ultrasound	Summary	Medical ultrasound includes diagnostic techniq...	336
1	Medical ultrasound	Types	The imaging mode refers to probe and machine s...	643
2	Medical ultrasound	Advantages and drawbacks	Compared to other medical imaging modalities, ...	95
3	Medical ultrasound	Uses	Sonography (ultrasonography) is widely used in...	206
4	Medical ultrasound	Anesthesiology	In anesthesiology, ultrasound is commonly used...	93

Figura 4.1: Ejemplo de DataFrame donde se muestra el encabezado y las cinco primeras entradas que corresponden con las cinco primeras secciones de la página de Wikipedia “*Medical ultrasound*”.

Por último, haciendo uso del módulo *csv* se almacenan en local los datos del DataFrame en un archivo en formato CSV (Comma Separated Values) que es el formato de importación y exportación más común de bases de datos. El archivo sigue la misma estructura de encabezado, filas y columnas con la peculiaridad de que las columnas están delimitadas por comas.

4.1.2. Formulación de preguntas y respuestas sintéticas

Sabiendo que el conjunto de datos debe estar compuesto por parejas de preguntas-respuestas o *prompt-completion*, se busca automatizar el proceso de creación puesto que sería ineficiente realizarlo manualmente. Con el propósito de realizar esta tarea, será fundamental la función *openai.Completion.create* de la biblioteca OpenAI, que permite generar texto basado en una entrada previamente proporcionada.

A continuación, para una mejor comprensión del mecanismo de generación de texto, se presentan los parámetros más comunes de la función *openai.Completion.create*. Pueden ser obligatorios (**OBL**) u opcionales (**OPT**) [38]:

- **Model**: variable tipo string que hace referencia al identificador del modelo a utilizar (**OBL**).
- **Prompt**: string o array con el texto de entrada que podría ser una instrucción o la pregunta para la que se generará una respuesta (**OPT**).
- **Temperature**: variable tipo number que controla la aleatoriedad de las respuestas generadas. Puede tomar valores entre 0 y 2 aunque por defecto toma el valor 1 (**OPT**).
- **Max tokens**: integer que representa el valor máximo de tokens que son generados en cada respuesta. El valor predeterminado es 16 (**OPT**).

- **Top p**: variable tipo number que controla la probabilidad acumulada de las palabras generadas. Por defecto toma el valor 1 y se recomienda modificar este parámetro o el de temperatura, pero en ningún caso los dos simultáneamente (**OPT**).
- **Frequency penalty**: number comprendido entre -2 y 2 que permite controlar la frecuencia de palabras repetidas en las respuestas que genera el modelo de lenguaje. Por defecto el valor es 0 (**OPT**).
- **Presence penalty**: number comprendido entre -2 y 2 que permite ajustar la presencia de determinadas palabras o frases específicas en las respuestas generadas por el modelo (**OPT**).
- **Stop**: string o array con el que la API dejará de generar más tokens (**OPT**).

En el presente TFG se utiliza esta función para crear sintéticamente las preguntas y sus correspondientes respuestas. En ambos casos se han utilizado los siguientes parámetros:

- **Model**: en esta ocasión se utilizará el modelo instruct seleccionado en el Capítulo 3. Es necesario dar una instrucción como es la de crear preguntas en base a un texto proporcionado o la de proporcionar respuestas a determinadas preguntas. Dado que los modelos *instruct* son especialmente útiles para el desarrollo de este tipo de tareas, en las que la intención es crucial, se proporcionará el identificador "text-davinci-003".
- **Prompt**: si el propósito es la elaboración de preguntas, se debe indicar al modelo que utilice el contexto proporcionado, siendo el contexto el contenido extraído de las secciones de Wikipedia. Sin embargo, si el propósito es crear respuestas, se indicará que se utilicen las preguntas obtenidas anteriormente.
- **Temperature**: se establece un valor nulo para obtener preguntas y respuestas coherentes y precisas que favorezcan la elaboración de un conjunto de datos de calidad.
- **Max tokens**: se limita el máximo de tokens a 250 para evitar generar contenido demasiado largo.
- **Top p**: como se ha modificado la temperatura, este parámetro se restringe a 1, que es el valor predeterminado.
- **Frequency penalty**: toma el valor nulo para indicar que no se aplica penalización por la frecuencia con la que aparece una palabra.

- **Presence penalty:** se establece el valor nulo permitiendo al modelo generar texto libremente, sin tener en cuenta la presencia de ciertas palabras en el texto generado.
- **Stop:** se establece el valor nulo, en este caso no existe ninguna secuencia de parada.

Las parejas sintéticas de pregunta-respuesta se han almacenado en el archivo CSV previo (véase Figura 4.2).

	title	heading	content	tokens	context	questions	answers
0	Medical ultrasound	Summary	Medical ultrasound includes diagnostic techniq...	336	Medical ultrasound\nSummary\n\nMedical ultraso...	1. What is medical ultrasound?\n2. What are t...	1. Medical ultrasound is the use of ultrasound...
1	Medical ultrasound	Types	The imaging mode refers to probe and machine s...	643	Medical ultrasound\nTypes\n\nThe imaging mode ...	1. What is A-mode?\n2. What is B-mode?\n3. Wh...	1. A-mode is a one-dimensional mode in which t...
2	Medical ultrasound	Advantages and drawbacks	Compared to other medical imaging modalities, ...	95	Medical ultrasound\nAdvantages and drawbacks\n...	1. What are some of the advantages of ultraso...	1. Some of the advantages of ultrasound compar...
3	Medical ultrasound	Uses	Sonography (ultrasonography) is widely used in...	206	Medical ultrasound\nUses\n\nSonography (ultras...	1. What are the different medical uses for ul...	1. Ultrasound is used for both diagnostics and...
4	Medical ultrasound	Anesthesiology	In anesthesiology, ultrasound is commonly used...	93	Medical ultrasound\nAnesthesiology\n\nIn anest...	1. What is ultrasound used for in anesthesiol...	1. Ultrasound is used for guiding the placemen...

Figura 4.2: Lectura del archivo CSV con la biblioteca Pandas donde se muestran las preguntas y respuestas sintéticas.

4.1.3. Ajuste de formato de los datos

Antes de establecer las parejas *prompt-completion* para crear el archivo JSONL explicado en la Sección 3.3, se comprueba que existe un equilibrio entre el número de preguntas y respuestas para asegurar que todas las preguntas han sido respondidas con normalidad.

OpenAI establece que cada *prompt* debe ir acompañado de un separador fijo para informar al modelo de cuándo termina la pregunta y cuándo comienza la respuesta. Recomienda utilizar separadores sencillos que no formen parte del contenido del JSONL como: “\n\n###\n\n”. De igual manera, tras cada completion se debe añadir una secuencia de parada fija que indique al modelo cuándo finaliza la respuesta. En este caso se utiliza como token: “END”. Añadiendo estos tokens se ayuda considerablemente al modelo de lenguaje a comprender los datos propuestos y se evita la alucinación del modelo en el momento de generar las respuestas [27]. Como se verá en la Sección 4.3, es importante recordar la existencia de estos separadores y secuencias de parada para utilizar correctamente el modelo especializado.

Finalmente, utilizando la librería *JSON* se elabora el archivo en formato JSONL. Se obtiene como resultado un archivo similar al de la Figura 4.3 con un total de más de 8,000 ejemplos de

pregunta-respuesta o *prompt-completion* que servirán como conjunto de datos para el entrenamiento del modelo Davinci.



Figura 4.3: Ejemplo de la estructura de un JSONL listo para el reentrenamiento del modelo GPT-3.

4.2. Entrenamiento específico

Una vez los datos han sido recopilados y estructurados en el formato adecuado que procesan los modelos de lenguaje de OpenAI, se procede a realizar el entrenamiento específico del modelo DaVinci.

En primer lugar, se debe cargar el archivo JSONL que contiene los pares clave-valor con las preguntas y respuestas relacionadas con equipamiento biomédico. OpenAI establece que cada organización pueda cargar un máximo de 1 GB, pero en el presente TFG el archivo de entrenamiento tiene un tamaño de 1,878 KB [39]. Para subir el JSONL a la plataforma de OpenAI se necesita realizar una petición HTTP tipo POST, donde el cuerpo de la solicitud es un formulario en el que se adjunta el conjunto de datos. Además, será necesario indicar la finalidad con la que se realiza la petición, en este caso, *fine-tune*. En consecuencia, si el proceso se ha realizado de manera correcta, la entidad responde con un JSON donde se indica el identificador para el archivo de entrenamiento.

En segundo lugar, iniciamos el ajuste del modelo realizando otra petición POST. El cuerpo de la solicitud es un texto sin formato cuyo contenido es un objeto JSON, donde se deben indicar los valores de los hiperparámetros expuestos a continuación. Pueden ser obligatorios (**OBL**) u opcionales (**OPT**) [40]:

- **Training file**: string que contiene el identificador obtenido como respuesta en la petición anterior, correspondiente al archivo que contiene el conjunto de datos necesarios para el entrenamiento (**OBL**).

- **Model:** string que indica el nombre del modelo base que buscamos reentrenar. Por defecto se utiliza el modelo Curie (**OPT**).
- **Suffix:** string de hasta cuarenta caracteres que funciona como un sufijo de extensión para completar el nombre e indicar la naturaleza del modelo reentrenado (**OPT**).
- **N epochs:** número entero (integer) que refleja el número de épocas. Hace referencia a la cantidad de veces que se completa un ciclo completo de entrenamiento utilizando todo el conjunto de datos de entrenamiento. El valor predeterminado es cuatro (**OPT**).
- **Batch size:** integer que indica el tamaño del lote, es decir, la cantidad de ejemplos de entrenamiento que se utilizan en cada iteración del proceso de entrenamiento del modelo de lenguaje. Por defecto, es aproximadamente el 0.2 % del número de ejemplos del conjunto de entrenamiento, limitado a 256 (**OPT**).
- **Learning rate multiplier:** variable tipo number que representa el multiplicador de la tasa de aprendizaje para el reentrenamiento. Se trata de la tasa de aprendizaje original para el modelo preentrenado multiplicada por el valor indicado en esta variable, pudiendo así aumentar o disminuir la velocidad de aprendizaje durante el entrenamiento. Por defecto se utiliza una tasa de aprendizaje 0.1 (**OPT**).
- **Prompt loss weight:** variable tipo number que representa el peso de pérdida del *prompt*. Con este hiperparámetro se indica la influencia del *prompt* en la optimización del modelo. El valor predeterminado es 0,01 (**OPT**).

Existen otros hiperparámetros de reentrenamiento acordes a una tarea de clasificación, por tanto, en el caso de este TFG no serán de interés.

Para la realización de este proyecto se establecen los hiperparámetros de la Figura 4.2. En el caso del número de épocas, el tamaño del lote y la tasa de aprendizaje del modelo se consideran los valores por defecto, dado que son los comúnmente utilizados obteniendo buenas prestaciones. Para el hiperparámetro *prompt loss weight* se considera que algunos *prompts* del conjunto de entrenamiento son largos. En tal caso, se busca reducir el peso estableciendo un valor bajo del hiperparámetro (el valor predeterminado ya es bajo) para evitar priorizar el aprendizaje del *prompt*. Es importante apreciar que en esta etapa el modelo que se indica es el modelo base Davinci, que admite el proceso de entrenamiento.

Como respuesta a la petición, el servidor de OpenAI proporciona un identificador del proceso de entrenamiento. Realizando una petición GET en la que se indique dicho identificador se

puede comprobar el estado en el que se encuentra el entrenamiento, que puede tomar algunos minutos, y verificar que se ha realizado con éxito.

```
{
  "training_file": "file-AF1WoRqd3aJAHsqc9NY7iL8F",
  "model": "davinci",
  "suffix": "biomedical-equipment-AI"
}
```

Figura 4.4: Ejemplo del cuerpo de una solicitud POST para el proceso de entrenamiento específico de un modelo de lenguaje en OpenAI.

4.3. Implementación en Telegram

Una vez se ha ajustado el modelo es fundamental aprovechar la GUI de Telegram para disponer de una plataforma que permita a los usuarios acceder y utilizar el modelo de lenguaje de manera simple y directa.

Tras iniciar una conversación con *BotFather* y a través del comando */newbot* se crea un nuevo bot siguiendo las instrucciones indicadas en la conversación. *BotFather* proporcionará un token de acceso al bot recién creado, imprescindible para interactuar con la API de Telegram y controlar el bot.

Utilizando la librería *Telebot*, que proporciona una interfaz para establecer la interacción entre el usuario y el bot, se debe definir el controlador de comando */start* con la finalidad de que el bot envíe un mensaje de bienvenida tan pronto como el usuario inicie la conversación. Dado que el objetivo es que el modelo de lenguaje ajustado responda las preguntas del usuario, se define el controlador de mensajes para que las interacciones posteriores se gestionen a través de la API de OpenAI. Entonces, se llama de nuevo a la función *openai.Completion.create* definiendo los siguientes parámetros:

- **Model:** en esta ocasión introducimos el identificador del modelo ajustado, proporcionado por OpenAI en la petición POST.
- **Prompt:** pregunta recibida por Telegram. Como se estableció durante el proceso de entrenamiento, se debe añadir el separador `"\n\n###\n\n"` a todas las preguntas recibidas.

- **Stop**: se define la secuencia de parada "END" también establecida en el proceso de *fine-tuning*.
- **Max tokens**: se fija el valor máximo en 250 buscando respuestas no excesivamente largas.
- **Temperature**: posteriormente se variará el valor de la temperatura.

Estableciendo estos parámetros, el modelo de lenguaje es capaz de elaborar las respuestas, basándose en el conjunto de datos de entrenamiento. Finalmente, la respuesta generada se transmitirá al usuario por Telegram y este podrá continuar con la comunicación si así lo desea.

4.4. Presupuesto

En esta sección se mostrarán los costes asociados a la elaboración completa de este proyecto, incluyendo el desarrollo, entrenamiento y utilización del modelo de lenguaje.

En primer lugar, se llevó a cabo la elaboración del conjunto de datos necesarios para el entrenamiento del modelo. En este proceso se recopiló la información necesaria de manera gratuita, sin embargo, el proceso de elaboración de preguntas y respuestas sintéticas generó un coste que se ilustra en la Figura 4.1.

Tecnología	Número de parejas <i>prompt-completion</i>	Precio (\$)
Tomografía Computarizada	812	8,73
Resonancia Magnética	2,043	12,74
Rayos X	2,049	13,21
Ultrasonidos	2,468	13,67
Medicina Nuclear	895	9,63
Total		58

Tabla 4.1: Inversión realizada en la elaboración del conjunto de datos de entrenamiento.

Los datos de entrenamiento se almacenaron en un archivo JSONL que fue subido a la plataforma de OpenAI para realizar el entrenamiento específico. En este caso, el proceso de entrenamiento ha incurrido en un gasto de 44\$ aproximadamente, lo que sumado al proceso de creación del conjunto de datos supone un coste total de 102\$ por el desarrollo del proyecto.

Para complementar el presupuesto por desarrollar una herramienta como la de este proyecto, sería conveniente incluir el salario del ingeniero encargado de diseñar y desarrollar el chatbot, en esta situación particular, una ingeniera biomédica. Se estima que un ingeniero biomédico

tiene un salario bruto anual que oscila alrededor de los 27,000 € de media en Madrid [41]. Este valor varía en función de la experiencia y puesto a desempeñar, así como de la duración del proyecto asignado. Teniendo en cuenta que para la fase de desarrollo de este proyecto fue necesario emplear 3 meses, el sueldo de la ingeniera se estima en 6,750 € por 3 meses de trabajo a jornada completa.

Finalmente, considerando estos factores, se estima que el presupuesto final para el desarrollo de un chatbot como el descrito a lo largo del presente TFG serían necesarios 6844 € aproximadamente. Se debe recordar que cada interacción con el bot lleva asociado un coste por tokens como se mostró en el Cuadro 3.2. Por tanto, el precio por mantener el chatbot dependerá de su extensión en la comunidad y del número de interacciones con el modelo de lenguaje de OpenAI.

Capítulo 5

Experimentos y resultados

En este capítulo se presenta la parte experimental de este TFG. Se define la configuración experimental desarrollada y se muestran los resultados de los experimentos propuestos. Esto permitirá obtener información sobre el rendimiento del modelo que será analizada posteriormente para comprobar su correcto funcionamiento.

5.1. Evaluación interna

5.1.1. Impacto de la temperatura

Como se ha indicado en el Capítulo 4, el proceso de elaboración del conjunto de datos implica generar preguntas y respuestas de manera sintética. Estas sirven para el entrenamiento específico simulando ejemplos de interacciones entre el usuario y el modelo.

En esta sección se realizan algunas preguntas que han sido incluidas en el conjunto de datos. El propósito se basa en observar cómo se ve alterada la respuesta, con respecto a la respuesta original que se proporcionó durante el entrenamiento específico, ante un cambio del valor en el parámetro de la temperatura de la función *openai.Completion.create*.

Para llevar esto a cabo, se escogen aleatoriamente ciertas preguntas y se formulan al modelo utilizando diferentes ajustes del parámetro de la temperatura. Específicamente se utilizan los siguientes valores: 0.5, 0.7, 0.9, siendo 0.7 el valor aconsejado. Esto permite explorar si las respuestas se vuelven más precisas y literales a la respuesta original a medida que se reduce la temperatura, o, por el contrario, si se vuelven más creativas a medida que aumenta.

La métrica utilizada para determinar cómo varían las respuestas en función de la temperatura

elegida será la similitud. Para poder determinar la similitud semántica entre las frases generadas es necesario hacer uso de las incrustaciones de palabras definidas en la Sección 2.1.

En este caso se utiliza la librería *Spacy* en Python que permite obtener vectores de las palabras. Para ello se utiliza el modelo de lenguaje “en_core_web_lg”, un modelo preentrenado en inglés y que contiene incrustaciones de hasta 685,000 palabras [42]. SpaCy permite comparar dos objetos y predecir su similitud utilizando la similitud del coseno, comprendida entre 0 y 1. Esta métrica mide el coseno del ángulo formado entre dos vectores estableciendo que a menor valor del ángulo, mayor será la similitud del coseno, es decir, más cercana al valor unidad (véase Ecuación 5.1) [43].

$$\text{Similitud}(\mathbf{A}, \mathbf{B}) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \cdot \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i \cdot B_i}{\sqrt{\sum_{i=1}^n A_i^2} \cdot \sqrt{\sum_{i=1}^n B_i^2}} \quad (5.1)$$

Donde:

- \mathbf{A} y \mathbf{B} son vectores que representan dos frases.
- A_i y B_i representan los elementos de los vectores en la posición i .
- n es la longitud de los vectores.

A continuación, se muestran cinco preguntas (P), una por cada tecnología de imagen médica en la que se especializa el modelo. Se muestran las respectivas respuestas utilizadas durante el entrenamiento específico (C), así como las respuestas generadas con diferentes temperaturas (C-0.5, C-0.7, C-0.9):

Tomografía Computarizada (TC)

- P: “What is the advantage of electron beam tomography over other types of CT?”
- C: “The advantage of electron beam tomography over other types of CT is that sweep speeds can be much faster, allowing for less blurry imaging of moving structures, such as the heart and arteries.”
- C-0.5: “The advantage of electron beam tomography over other types of CT is that it can produce high-resolution images using a large X-ray beam and a shorter scan time.”
- C-0.7: “The advantage of electron beam tomography over other types of CT is that it produces high-quality images much faster than other types of CT.”

- C-0.9: “The advantage of electron beam tomography over other types of CT is that it allows for three-dimensional (3D) imaging.”

Resonancia Magnética (RM)

- P: “What components are typically found in an NMR spectrometer?”
- C: “An NMR spectrometer typically consists of a spinning sample-holder inside a very strong magnet, a radio-frequency emitter, a receiver with a probe (an antenna assembly), optionally gradient coils for diffusion measurements, and electronics to control the system.”
- C-0.5: “The components typically found in an NMR spectrometer are a magnet, a sample holder, a detector, a data processor, and a display.”
- C-0.7: “The components typically found in an NMR spectrometer are a magnetic field generator, a vacuum tube for signal generation and detection, a data processing unit, and a console for operation and control.”
- C-0.9: “The components typically found in an NMR spectrometer are a gradient controller, a pulse programmer, a computer, and a scanner.”

Rayos X (RX)

- P: “What are the benefits of the SDD (Silicon drift detectors)?”
- C: “The benefits of the SDD include high count rates and processing, better resolution than traditional Si(Li) detectors at high count rates, lower dead time, faster analytical capabilities and more precise X-ray maps or particle data collected in seconds, and the ability to be stored and operated at relatively high temperatures, eliminating the need for liquid nitrogen cooling.”
- C-0.5: “The benefits of the SDD (Silicon drift detectors) are that they are less affected by radiation than Ge detectors, and they have a higher probability of triggering on charged particles.”
- C-0.7: “The benefits of the SDD (Silicon drift detectors) are that they have high spatial resolution and can provide information about the momentum of the incident photons.”
- C-0.9: “The benefits of the SDD (Silicon drift detectors) are that they have high efficiency, high speed, and high resolution.”

Ultrasonidos (US)

- P: “What is M-mode?”
- C: “M-mode is an ultrasound analogy to streak video recording in high-speed photography. As moving tissue transitions produce backscattering, this can be used to determine the displacement of specific organ structures, most commonly the heart.”
- C-0.5: “M-mode is a graphic display that uses a horizontal axis to represent the cardiac cycles and a vertical axis to represent the excursion of the heart muscle.”
- C-0.7: “M-mode is a type of echocardiography that creates images in a series of cross-sectional views.”
- C-0.9: “M-mode is a visualization technique used in echocardiography to assess the right and left ventricles of the heart.”

Medicina Nuclear (MN)

- P: “What are the most commonly used intravenous radionuclides?”
- C: “The most commonly used intravenous radionuclides are technetium-99m, iodine-123, iodine-131, thallium-201, gallium-67, fluorine-18 fluorodeoxyglucose, and indium-111 labeled leukocytes.”
- C-0.5: “The most commonly used intravenous radionuclides are technetium99m and iodine 123.”
- C-0.7: “The most commonly used intravenous radionuclides are sodium, potassium, and rubidium.”
- C-0.9: “The most commonly used intravenous radionuclides are technetium99m, iodine 131, and gallium 67.”

Modalidad	Respuestas		
	C-05	C-07	C-09
TC	0.927654	0.968895	0.927848
RM	0.896160	0.936284	0.914568
RX	0.918986	0.929392	0.924701
US	0.818356	0.743492	0.917338
MN	0.680106	0.839507	0.844937

Tabla 5.1: Similitud entre las respuestas generadas con diferentes valores de temperatura y la respuesta original de entrenamiento.

Por otro lado, es conveniente analizar qué sucede para valores extremos de la temperatura. A continuación, se muestra un ejemplo de una pregunta (P) y sus respectivas respuestas con valores límite de la temperatura, como es el caso de 0 y 2 (C-0, C-2):

- P: “What is the advantage of electron beam tomography over other types of CT?”
- C-0: “ The advantage of electron beam tomography over other types of CT is that it can produce volumetric images.”
- C-2: “ The advantage of electron beam tomography over other types of CT is that it producesuchiage over other types of exact colour and Hexachrome work so that fine nuances of Impasto and details of Adopt of types Perspective images can be simulated very well. APEHT delivers a raytrace computer graphic OOM OPF OkIll which sends Opwt your OO PMg scanned beame Cast Eye as an expert computer programme that records draw Ed its worth EOK data. RC0 RKE Ire calls forth rock slides Diograms LEO diagrammatics Dh(hidden co Google UKYP categories DP Disck Overtones Rosc Computer DiagThe RR book five MK Ge RoC help make eChoise render near Ekchical DEVP like Hones Hop GR whether teXINTYT you need AD half moons for Opys a Okpr childl photograph PhOtog enrol what BAM Dix hCHTshite Oh Fokesex patients Poz might shed told ever little QDC V3A to UP0 au reader that sounds PeRmal IXKTIOposter Bsfavour LOL PWelldu yet pRIOhnce on Phosadaysg CILOArse let Piao cock”

5.1.2. Capacidad de generalización

En esta etapa de la evaluación interna se formulan diversas preguntas al chatbot especializado en equipamiento biomédico, pese a que las preguntas realizadas no están relacionadas con el dominio del conocimiento. Con esta prueba se busca evaluar la capacidad del modelo para

responder de manera coherente en un contexto fuera del campo sobre el cual se ha realizado su entrenamiento específico. De este modo se puede comprobar la forma en la que se desenvuelve en otras áreas del conocimiento.

A continuación, se exponen las preguntas realizadas al modelo (P) y las respuestas generadas con valor de 0.7 para la temperatura (C):

- P1: “Hello, how are you?”
- C: “I am fine, thanks.”
- P2: “What is Telegram?”
- C2: “Telegram is a cloud-based instant messaging service.”
- P3: “Name 3 engineering careers”
- C3: “Some engineering careers include materials science, mechanical engineering, and aerospace engineering.”
- P4: “What is Artificial Intelligence?”
- C4: “Artificial Intelligence is a branch of computer science that aims to create machines that can perform tasks that require human intelligence.”
- P5: “What is the Big Bang?”
- C5: “The Big Bang is a theory that explains the early development of the Universe.”
- P6: “1343432”
- C6: “###

###’

5.2. Evaluación externa

5.2.1. Interacción con usuarios

La principal utilidad del chatbot es ofrecer la oportunidad de que diferentes individuos se comuniquen con él aprovechando sus conocimientos en equipos médicos. En esta fase participarán cinco usuarios que previamente han cursado asignaturas relacionadas con el ámbito del equipamiento biomédico, cada uno con la oportunidad de realizar preguntas al chatbot. Los resultados de esta evaluación proporcionarán información valiosa tanto para mejorar las respuestas incorrectas como para apreciar que tipo de cuestiones puede realizar un estudiante de la materia y refrozar el modelo en esos aspectos.

Seguidamente, se muestran las preguntas (P) realizadas por los usuarios así como las respuestas que genera el chatbot (C) para un valor de 0.7 de la temperatura.

- P1: “What is an isotope?”
- C1: “An isotope is a species of the same chemical element that has a different mass number.”
- P2: “What magnitude are the magnets used in magnetic resonance?”
- C2: “Magnets used in magnetic resonance have a magnitude of 0.2 T to 3 T. ”
- P3: “What is a photomultiplier tube?”
- C3: “A photomultiplier tube is a device that uses electromagnetic induction to multiply the current emitted by a single gamma ray interaction in a Geiger tube.”
- P4: “What does an anti-scatter grid do?”
- C4: “An anti-scatter grid does not affect the wavelength distribution, but it reduces the radiation scatter due to interactions with the material of the anti-scatter grid.”
- P5: “What is received dose and administered dose?”
- C5: “Received dose is the dose that the patient actually receives, while the administered dose is the amount of radiation given to the patient.”
- P6: “What are the positive effects of using filters to modify the X-ray radiation spectrum?”

- C6: “The positive effects of using filters to modify the X-ray radiation spectrum include improved contrast and less radiation exposure for the patient.”
- P7: “What is the K-space in magnetic resonance?”
- C7: “The K-space in magnetic resonance is the spatial representation of the raw data.”
- P8: “What are the pros and cons of obtaining magnetic resonance images?”
- C8: “The pros of obtaining magnetic resonance images are that they are accurate and reliable, and they do not expose the patient to radiation. The cons are that they are expensive and time-consuming.”
- P9: “What phenomenon does functional magnetic resonance imaging rely on?”
- C9: “Functional magnetic resonance imaging (fMRI) relies on the blood-oxygen-level-dependent (BOLD) contrast.”
- P10: “What is the inverse square law?”
- C10: “The inverse square law is a law of physics that states that the radiation from a point source is proportional to the inverse square of the distance from the source.”
- P11: “What is the piezoelectric effect?”
- C11: “The piezoelectric effect is the accumulation of charge in a crystal due to mechanical stress.”
- P12: “What is the recommended use of radiocontrast agents in pregnant women?”
- C12: “The recommended use of radiocontrast agents in pregnant women is controversial, but some studies have found no increased risk of fetal loss or congenital anomalies.”
- P13: “What are the benefits of using a cyclotron?”
- C13: “The benefits of using a cyclotron are that it can produce radioactive drugs for diagnosis and therapy and it can produce isotopes for research.”
- P14: “What is ALARA?”
- C14: “ALARA is an acronym for “As Low As Reasonably Achievable”, which emphasizes the importance of radiation safety in X-ray imaging.”

- P15: “Explain the concept of annihilation in PET”
- C15: “In positron emission tomography (PET), annihilation occurs when a positron (the antiparticle of the electron) meets an electron, resulting in the simultaneous emission of a pair of gamma rays with a wavelength of about 511 keV.”

5.2.2. Cuestionario

En esta sección se presentan los resultados de un cuestionario realizado a los usuarios que han interactuado con el chatbot especializado en equipamiento biomédico. Se busca conocer la satisfacción de los usuarios, así como recopilar información valiosa sobre la efectividad, facilidad o probabilidad de recomendación de uso del chatbot.

A continuación, se muestran los principales hallazgos obtenidos a partir del cuestionario realizado:

- **Usabilidad del chatbot:** se pidió a los usuarios calificar la facilidad de uso del chatbot en una escala del 1 al 5, siendo 1 “Muy insatisfecho/a” y 5 “Muy satisfecho/a”. En la Figura 5.1 se observa que los usuarios indicaron una alta satisfacción en relación a la usabilidad del chatbot.

¿Qué tan satisfecho/a estás con la facilidad de uso del chatbot?

5 respuestas

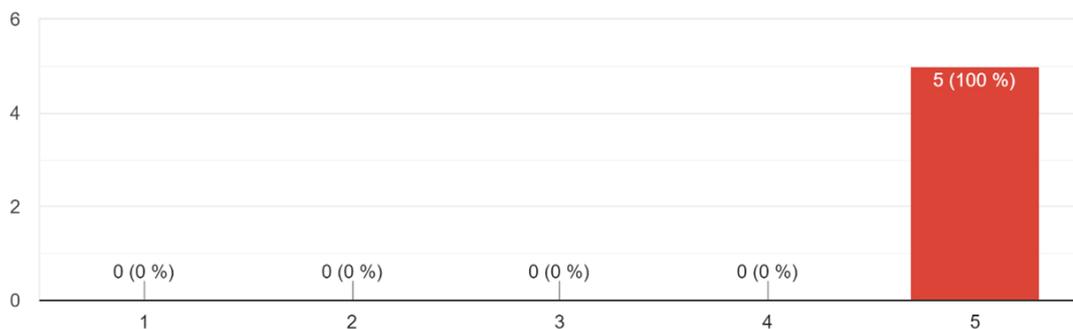


Figura 5.1: Gráfico de la primera pregunta del cuestionario realizado a los usuarios.

- **Utilidad de las respuestas generadas:** en respuesta a la pregunta sobre la utilidad de las respuestas proporcionadas, donde 1 representa “Nada útil” y 5 “Muy útil”, la mayoría de los usuarios consideran que las respuestas fueron útiles (véase Figura 5.2).

¿En qué medida consideras útiles las respuestas proporcionadas por el chatbot?

5 respuestas

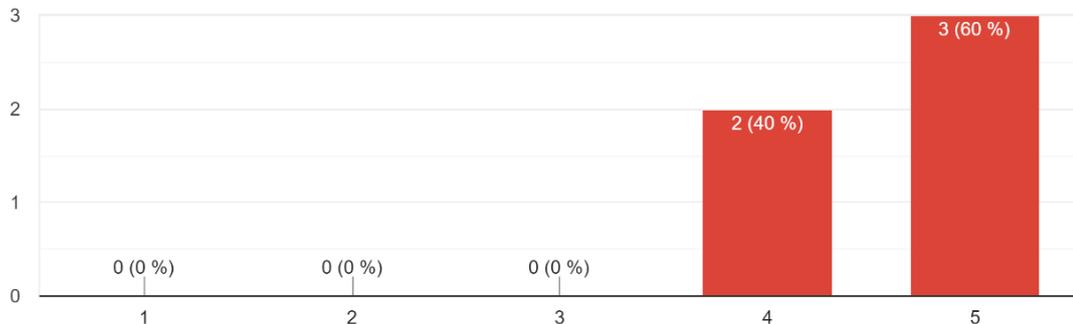


Figura 5.2: Gráfico de la segunda pregunta del cuestionario realizado a los usuarios.

- **Rapidez y eficiencia del chatbot:** en relación a la rapidez y eficiencia del chatbot al responder las preguntas, se pidió a los usuarios que lo calificaran en una escala donde 1 es “Muy lento e ineficiente” y 5 es “Muy rápido y eficiente”. Los usuarios exaltaron su rapidez y eficiencia generando respuestas (véase Figura 5.3).

¿Qué tan rápido y eficiente fue el chatbot al responder tus preguntas sobre equipamiento biomédico?

5 respuestas

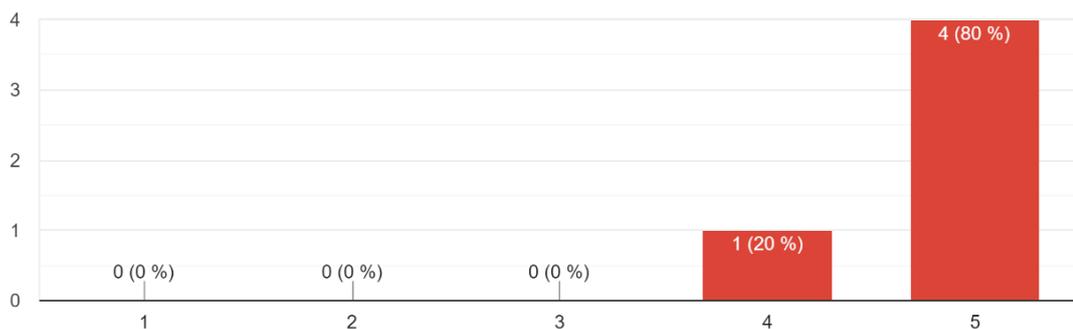


Figura 5.3: Gráfico de la tercera pregunta del cuestionario realizado a los usuarios.

- **Probabilidad de recomendación:** los usuarios evaluaron la probabilidad con la que recomendarían el chatbot a otros estudiantes. Se observa que todos los usuarios recomendarían el chatbot a sus compañeros (véase Figura 5.4).

¿Recomendarías el chatbot a otros alumnos?

5 respuestas

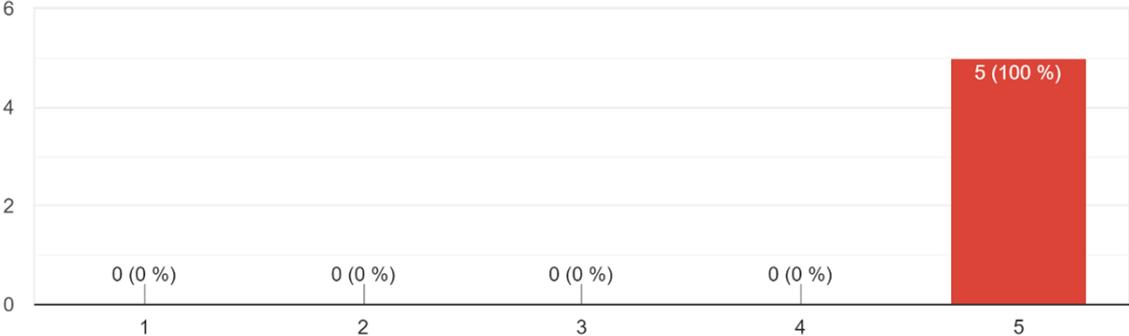


Figura 5.4: Gráfico de la cuarta pregunta del cuestionario realizado a los usuarios.

Capítulo 6

Discusión

En esta sección se analizan los resultados obtenidos durante la etapa de evaluación del chatbot. Así, se pretende determinar el rendimiento del mismo y su capacidad para desempeñar la tarea principal, dar respuesta a cuestiones de equipamiento biomédico. Esto permitirá obtener una visión generalizada del desempeño del chatbot proporcionando información valiosa que se debe tener en cuenta para mejorar y perfeccionar el chatbot en etapas posteriores.

6.1. Análisis de resultados

En cuanto a la evaluación interna, se modificó el valor de la temperatura para generar tres respuestas diferentes a una misma pregunta utilizada durante el entrenamiento específico del modelo. Así se puede comprobar la similitud entre las respuestas obtenidas y la respuesta original con la que el modelo fue entrenado. Se observa que con la temperatura establecida en 0.7 se obtiene una mayor similitud en tres de las cinco preguntas evaluadas. Esto llevaría a pensar que un valor de temperatura de 0.7 proporciona respuestas más coherentes y acordes a la respuesta original. Sin embargo, se debe tener en cuenta que en los otros dos casos la temperatura de 0.9 resultó en respuestas más similares a la respuesta del entrenamiento específico. En cualquier caso, para los tres valores de la temperatura obtenemos una similitud superior al 68% estableciendo que, a pesar de proporcionar mayor o menor libertad de creación, obtenemos respuestas óptimas en cuanto a contenido.

Por otro lado, para valores extremos de la temperatura, como es el caso de 0, el modelo de lenguaje sigue manteniendo su capacidad para proporcionar respuestas correctas. No obstante, si la temperatura toma el valor 2, el modelo generará una respuesta más larga, alcanzando el va-

lor máximo de tokens establecido en la función *openai.Completion.create*. Parte de la respuesta está relacionada con la temática sobre la que se pregunta, sin embargo, en la otra parte de la respuesta el modelo empieza a “alucinar” proporcionando información incoherente. Este evento demuestra la importancia de elegir un valor de temperatura adecuado que permita al chatbot desempeñar su función apropiadamente.

En la evaluación interna también se realizó una evaluación adicional donde se puso a prueba la robustez y capacidad del modelo para enfrentarse a situaciones inesperadas. En esta fase se formularon preguntas más allá de los límites de conocimiento proporcionados durante el entrenamiento en equipamiento biomédico. Esto permitió evidenciar que el chatbot posee capacidad suficiente para responder preguntas que están fuera del dominio de su conocimiento específico. Estos resultados confirman que estos modelos de lenguaje han sido preentrenados con una gran cantidad de datos y tienen capacidad para generar texto ante cualquier entrada. A pesar de esto, en caso de interactuar con el chatbot de manera incoherente, escribiendo tokens sin sentido, el modelo de lenguaje no es capaz de proporcionar una respuesta lógica.

Ante estas interacciones, fuera del dominio de conocimiento o incoherentes, el chatbot podría adoptar la estrategia de reconocer que la pregunta está fuera de su alcance y, en lugar de proporcionar una respuesta correcta o incongruente, dependiendo del caso, debería aceptar que la pregunta está relacionada con una temática para la que no fue entrenado. De esta forma el modelo podría proporcionar respuestas disculpándose por no poder dar una respuesta exacta, dado que no está dentro de su área de conocimiento e invitar al usuario a consultar otra fuente adicional para obtener la información adecuada.

Con relación a la evaluación externa, se pidió que diferentes usuarios interactuaran con el chatbot para comprobar que realizaba sus funciones correctamente y evaluar la experiencia de los usuarios al utilizarlo. Se registraron las respuestas generadas por el modelo de lenguaje, observando que el modelo respondía adecuadamente a las preguntas formuladas. En algunos casos, se puede apreciar la brevedad de las respuestas, lo que restringe la cantidad de información ofrecida, pudiendo no ser suficiente para aclarar completamente el concepto consultado a pesar de obtener una respuesta correcta. En todas las consultas, el máximo de tokens para generar texto se estableció en 250, sin embargo, la respuesta con mayor longitud es de 50 tokens. Esto se puede deber principalmente a las respuestas proporcionadas en el conjunto de entrenamiento, ya que la configuración del conjunto de datos es determinante a la hora de entrenar el modelo. Se observaron las respuestas proporcionadas durante el entrenamiento específico y eran generalmente cortas, condicionando al modelo, que adoptó esa cualidad en el momento de generar texto. Por tanto, podría ser interesante elevar el número de tokens utilizado para generar

las respuestas del conjunto de entrenamiento, lo que incurriría en un mayor gasto tanto en la generación de las respuestas sintéticas como en el proceso de entrenamiento, pero repercutiendo en un mejor rendimiento del modelo de lenguaje.

A esto se añade el *prompt engineering*. Se obtendrán mejores resultados siempre que el *prompt* o pregunta sea formulados de manera cuidadosa. Siempre que se establezcan instrucciones claras el modelo tendrá mayor capacidad para llevar a cabo la tarea conforme el usuario la requiera. Por ejemplo, para el chatbot puede resultar confuso realizar preguntas que impliquen acrónimos, pudiendo asociar ese acrónimo a otra área de conocimiento que no sea la requerida. Para ello, es primordial proporcionar un pequeño contexto adicional que permita identificar la especialidad a la que pertenece la pregunta formulada.

Para complementar esta información, se pidió a los usuarios que completaran un cuestionario que permitió extraer determinadas conclusiones. Los usuarios valoraron positivamente tanto la facilidad de uso, como la rapidez y efectividad del modelo en generar las respuestas. También se valoró satisfactoriamente la utilidad de las respuestas llevando a recomendar el chatbot a otros estudiantes. Sería recomendable brindar la oportunidad de interactuar con el chatbot a un público mayor, en especial, gente que en el momento de interactuar con él estuviera cursando alguna asignatura relacionada con la materia. Así, podría utilizarse como herramienta complementaria en el aprendizaje, evaluando qué conceptos se consultan con mayor frecuencia y poder reforzar esos conocimientos en el modelo de lenguaje para favorecer la satisfacción del estudiantado.

Capítulo 7

Conclusión y líneas futuras

En este capítulo se presentan las conclusiones obtenidas tras la realización de este proyecto y se proponen posibles líneas futuras de trabajo.

7.1. Conclusión

El presente TFG se centra en la tarea de entrenar el modelo Davinci de OpenAI con el propósito de especializarlo en el área del equipamiento biomédico, incluyendo tecnologías como Tomografía Computarizada, Resonancia Magnética, Rayos X, Ultrasonidos y Medicina Nuclear. Para facilitar a los usuarios el acceso y uso del modelo, también se ha implementado el modelo en un bot de Telegram que facilita la comunicación.

En primer lugar, se recopilaron los datos a través de la API de Wikipedia que permitieron confeccionar un conjunto de datos de entrenamiento. La información extraída fue utilizada para elaborar preguntas y respuestas sintéticas que simulan interacciones entre el usuario y el modelo. Esta información fue estructurada según las reglas que establece OpenAI para utilizar los datos durante el proceso de entrenamiento específico.

En segundo lugar, se procedió a entrenar el modelo Davinci a través de peticiones HTTP que establecieron la comunicación con el servidor de OpenAI. Se cargó el fichero de entrenamiento y se especificaron los hiperparámetros pertinentes para la adaptación del modelo existente a los conocimientos del conjunto de entrenamiento.

Una vez completado este proceso, aprovechando la API de Telegram, se creó un bot que permitiera integrar el modelo de lenguaje en su interfaz, resultando en un chatbot. Telegram recibe las consultas realizadas por el usuario que serán enviadas al servidor de OpenAI para que

el modelo de lenguaje previamente entrenado responda en base a los conocimientos adquiridos. Así, se consiguió que la interacción entre el usuario y el modelo fuera sencilla y satisfactoria.

Finalmente, se realizaron pruebas que permitieron evaluar el rendimiento del chatbot. En relación a las pruebas internas, se evaluó la capacidad creativa del modelo, así como su robustez respondiendo cuestiones fuera del dominio de conocimiento. En cuanto a las pruebas externas, participaron usuarios reales realizando preguntas sobre la temática al chatbot. Se registraron las respuestas para comprobar que el modelo de lenguaje proporcionaba información correcta. Se pidió que estos usuarios evaluaran la usabilidad, rapidez y utilidad del chatbot, así como la posibilidad de recomendación de este.

Los resultados obtenidos durante las pruebas de evaluación han sido satisfactorios. Esto indica que el modelo de lenguaje especializado en equipamiento biomédico ha logrado un óptimo desempeño demostrando cierta rapidez, eficiencia y coherencia en la resolución de preguntas. Sin embargo, existen posibles mejoras que permitirán perfeccionar y modificar las capacidades lingüísticas del modelo para impulsarlo a alcanzar un alto rendimiento.

En conclusión, este TFG ha evidenciado la utilidad de los modelos de lenguaje preentrenados, como el modelo Davinci GPT-3 de OpenAI. Realizando un entrenamiento específico sobre estos modelos se pueden modificar sus parámetros ajustándolos a un ámbito de conocimiento concreto. Esto facilita enormemente el camino de la creación de chatbots, ya que evita adquirir los complejos conocimientos que implica construir una red neuronal desde cero y su entrenamiento con cantidades masivas de datos.

7.2. Líneas futuras

A pesar de haber obtenido resultados aceptables, el campo de los modelos de lenguaje está en continuo crecimiento. Una muestra de ello son los nuevos modelos que ha desarrollado OpenAI durante la realización de este TFG.

Con el objetivo de mejorar la experiencia del usuario y garantizar que se satisfagan sus necesidades con respecto a los equipos biomédicos, se pueden considerar las siguientes ideas que permitan un mejor rendimiento del chatbot:

- Incluir el chatbot como un recurso docente más en la asignatura de Equipamiento Biomédico del grado en Ingeniería Biomédica de la Universidad Rey Juan Carlos.
- Realizar un análisis en colaboración con el público objetivo con el fin de mejorar y am-

pliar los conocimientos más demandados.

- Considerar otras tecnologías médicas que puedan ser incluidas en el conjunto de datos de entrenamiento del modelo de lenguaje para enriquecer el conocimiento del modelo en nuevas modalidades de imagen.
- Dotar al modelo de capacidad discriminativa para distinguir entre temas pertenecientes al área de especialización y los que se salen de los límites del conocimiento, limitando así al chatbot a proporcionar información sobre una única temática.
- Capacitación del modelo para operar en español u otros idiomas, ampliando su alcance y utilidad en diferentes públicos.
- Proporcionar habilidades al modelo para crear un contexto basado en las últimas interacciones con el usuario, permitiéndole continuar la conversación generando nuevas respuestas adaptadas a la temática que se plantea.

Bibliografía

- [1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [2] OpenAI. Tokenizer. Consultado el 26 de marzo de 2023. Disponible en: <https://platform.openai.com/tokenizer>.
- [3] Dingcheng Li, Jingyuan Zhang, and Ping Li. Tmsa: a mutual learning model for topic discovery and word embedding. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pages 684–692. SIAM, 2019.
- [4] Wikipedia. Category: Ultrasound. Consultado el 16 de junio de 2023. Disponible en: <https://en.wikipedia.org/wiki/Category:Ultrasound>.
- [5] Wikipedia. Medical ultrasound. Consultado el 16 de junio de 2023. Disponible en: https://en.wikipedia.org/wiki/Medical_ultrasound.
- [6] IBM. Natural Language Processing. Consultado el 18 de marzo de 2023. Disponible en: <https://www.ibm.com/es-es/topics/natural-language-processing>.
- [7] Universidad Autónoma de Madrid Instituto de Ingeniería del Conocimiento. Machine Learning y Deep Learning. Consultado el 18 de marzo de 2023. Disponible en: <https://www.iic.uam.es/inteligencia-artificial/machine-learning-deep-learning/>.
- [8] Tokio School. Redes Neuronales Recurrentes. Consultado el 20 de marzo de 2023. Disponible en: <https://www.tokioschool.com/noticias/redes-neuronales-recurrentes/>.
- [9] GAMCO. Red Neuronal Recurrente. Consultado el 20 de marzo de 2023. Disponible en: <https://gamco.es/glosario/red-neuronal-recurrente/>.

- [10] Atcold. El desvanecimiento de gradiente en redes neuronales. Consultado el 20 de marzo de 2023. Disponible en: <https://atcold.github.io/pytorch-Deep-Learning/es/week02/02-2/>.
- [11] Analytics Vidhya. What is Tokenization in NLP? Consultado el 26 de marzo de 2023. Disponible en: <https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/>.
- [12] DataScientest. Word Embedding, 2022. Consultado el 30 de marzo de 2023. Disponible en: <https://datascientest.com/es/word-embedding>.
- [13] Jay Alammar. The Illustrated Transformer, 2018. Consultado el 10 de abril de 2023. Disponible en: <http://jalammar.github.io/illustrated-transformer/>.
- [14] Aprende Machine Learning. ¿Cómo funcionan los Transformers? en Español, 2022. Consultado el 10 de abril de 2023. Disponible en: <https://www.aprendemachinelearning.com/como-funcionan-los-transformers-espanol-nlp-gpt-bert/>.
- [15] Evdokia Xygi, Andreas D Andriopoulos, and Dimitrios A Koutsomitropoulos. Question answering chatbots for biomedical research using transformers. In *2023 International Conference on Artificial Intelligence in Information and Communication (ICAIIIC)*, pages 025–029. IEEE, 2023.
- [16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [17] Long N Phan, James T Anibal, Hieu Tran, Shaurya Chanana, Erol Bahadroglu, Alec Peltekian, and Grégoire Altan-Bonnet. Scifive: a text-to-text transformer model for biomedical literature. *arXiv preprint arXiv:2106.03598*, 2021.
- [18] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.
- [19] Kai Zhang, Jun Yu, Zhiling Yan, Yixin Liu, Eashan Adhikarla, Sunyang Fu, Xun Chen, Chen Chen, Yuyin Zhou, Xiang Li, et al. Biomedgpt: A unified and generalist biomed-

- cal generative pre-trained transformer for vision, language, and multimodal tasks. *arXiv preprint arXiv:2305.17100*, 2023.
- [20] Wikipedia. OpenAI, 2022. Consultado el 8 de junio de 2023. Disponible en: <https://es.wikipedia.org/wiki/OpenAI>.
- [21] OpenAI. About-OpenAI. Consultado el 8 de junio de 2023. Disponible en: <https://openai.com/about>.
- [22] Félix Palazuelos. Elon musk:“la inteligencia artificial amenaza la existencia de nuestra civilización”. *El País*, 2021.
- [23] Sam Altman. Planning for AGI and beyond, 2023. Consultado el 8 de junio de 2023. Disponible en: <https://openai.com/blog/planning-for-agi-and-beyond>.
- [24] OpenAI. Research Index. Consultado el 8 de junio de 2023. Disponible en: <https://openai.com/research>.
- [25] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [26] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.
- [27] OpenAI. Fine-tuning. Consultado el 14 de junio de 2023. Disponible en: <https://platform.openai.com/docs/guides/fine-tuning>.
- [28] jsonlines.org. JSON Lines. Consultado el 14 de junio de 2023. Disponible en: <https://jsonlines.org/>.
- [29] Python.org. Preguntas frecuentes generales sobre Python. Consultado el 14 de junio de 2023. Disponible en: <https://docs.python.org/es/3.10/faq/general.html>.
- [30] Wikipedia. Historia de Wikipedia. Consultado el 16 de junio de 2023. Disponible en: https://es.wikipedia.org/wiki/Historia_de_Wikipedia.

- [31] Wikipedia. Wikipedia en español. Consultado el 16 de junio de 2023. Disponible en: https://es.wikipedia.org/wiki/Wikipedia_en_espa%C3%B1ol.
- [32] Wikipedia. Wikipedia en inglés. Consultado el 16 de junio de 2023. Disponible en: https://es.wikipedia.org/wiki/Wikipedia_en_ingl%C3%A9s.
- [33] Postman. What is Postman? Consultado el 17 de junio de 2023. Disponible en: <https://www.postman.com/product/what-is-postman/>.
- [34] Estilow3b. Métodos HTTP – POST, GET, PUT, DELETE. Consultado el 17 de junio de 2023. Disponible en: <http://estilow3b.com/metodos-http-post-get-put-delete/>.
- [35] Workana. ¿Qué es la interfaz gráfica de usuario GUI? Consultado el 18 de junio de 2023. Disponible en: <https://i.workana.com/glosario/que-es-la-interfaz-grafica-de-usuario-gui/>.
- [36] Telegram. Preguntas frecuentes. Consultado el 18 de junio de 2023. Disponible en: <https://telegram.org/faq/es>.
- [37] Telegram. Telegram APIs. Consultado el 18 de junio de 2023. Disponible en: <https://core.telegram.org/api>.
- [38] OpenAI. API Reference. Completions. Consultado el 20 de junio de 2023. Disponible en: <https://platform.openai.com/docs/api-reference/completions/create>.
- [39] OpenAI. API Reference. Files. Consultado el 20 de junio de 2023. Disponible en: <https://platform.openai.com/docs/api-reference/files>.
- [40] OpenAI. API Reference. Fine tunes. Consultado el 20 de junio de 2023. Disponible en: <https://platform.openai.com/docs/api-reference/fine-tunes>.
- [41] InfoJobs. Salarios, 2023. Consultado el 5 de julio de 2023. Disponible en: <https://salarios.infojobs.net/ingeniero-biomedico-ingeniera-biomedica/madrid>.
- [42] SpaCy. Linguistic Features · spaCy Usage Documentation. Consultado el 21 de julio de 2023. Disponible en: <https://spacy.io/usage/linguistic-features>.
- [43] ML-Studio. Similitud de coseno: comprensión de las matemáticas y cómo funcionan (con códigos de Python). Consultado el 21 de julio de 2023. Disponible en: <https://mlstudio.jaol.net/similitud-del-coseno/>.