UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería de Telecomunicación

Retrieval-Augmented Generation para la Extracción de Información de Documentos Inteligente

Trabajo Fin de Grado

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

AUTOR/A: Laparra Osca, Daniel

Tutor/a: Naranjo Ornedo, Valeriana

Cotutor/a externo: Marco Nin, Enric

CURSO ACADÉMICO: 2023/2024

Resumen

Retrieval-Augmented Generation (RAG) es una técnica de inteligencia artificial (IA), dentro del campo del procesamiento de lenguaje natural, empleada para la generación de texto combinando técnicas de recuperación de información con la generación de texto a partir de modelos de lenguaje. Su funcionamiento se basa en el uso de documentos relevantes recuperados para proporcionar contexto adicional, algo que permite generar respuestas más precisas y detalladas, lo que resulta especialmente útil en aplicaciones donde se requiere obtener información exacta a la vez que completa.

Este proyecto explora cómo la técnica de RAG puede mejorar significativamente la eficiencia en la extracción de información de documentos. Se describen los sistemas necesarios para implementar la aplicación final y se demuestra cómo la técnica puede impactar positivamente en métodos de trabajo eficientes.

Además, este análisis ofrece una perspectiva sobre cómo la inteligencia artificial está transformando la vida cotidiana y los métodos de trabajo de muchas personas. A medida que la IA se integra más en nuestras actividades diarias, herramientas como RAG demuestran su potencial para hacer nuestras interacciones con grandes cantidades de información más rápidas y precisas.

Resum

Retrieval-Augmented Generation (RAG) és una tècnica d'intel·ligència artificial (IA), dins del camp del processament de llenguatge natural, empleada per a la generació de text combinant tècniques de recuperació d'informació amb la generació de text a partir de models de llenguatge. El seu funcionament es basa en l'ús de documents rellevants recuperats per a proporcionar context addicional, alguna cosa que permet generar respostes més precises i detallades, el que resulta especialment útil en aplicacions on es requerix obtindre informació exacta alhora que completa.

Este projecte explora com la tècnica de RAG pot millorar significativament l'eficiència en l'extracció d'informació de documents. Es descriuen els sistemes necessaris per a implementar l'aplicació final i es demostra com la tècnica pot impactar positivament en mètodes de treball eficients.

A més, esta anàlisi oferix una perspectiva sobre com la intel·ligència artificial està transformant la vida quotidiana i els mètodes de treball de moltes persones. A mesura que la IA s'integra més en les nostres activitats diàries, ferramentes com RAG demostren el seu potencial per a fer les nostres interaccions amb grans quantitats d'informació més ràpides i precises.

Abstract

Retrieval-Augmented Generation (RAG) is an artificial intelligence technique within the field of natural language processing, used for text generation by combining information retrieval techniques with text generation from language models. Its operation is based on the use of relevant retrieved documents to provide additional context, which allows for the generation of more precise and detailed responses, which is particularly useful in applications where accurate and comprehensive information is required.

This project explores how the RAG technique can significantly improve efficiency in document information extraction. The necessary systems to implement the final application are described, and it is demonstrated how the technique can positively impact efficient work methods.

Furthermore, this analysis offers a perspective on how artificial intelligence is transforming everyday life and the work methods of many people. As AI becomes more integrated into our daily activities, tools like RAG demonstrate their potential to make our interactions with large amounts of information faster and more precise.

RESUMEN EJECUTIVO

La memoria del TFG del GTIST debe desarrollar en el texto los siguientes conceptos, debidamente justificados y discutidos, centrados en el ámbito de la IT

CONCEPT (ABET)	**CONCEPTO (traducción)**	**¿Cumple? (S/N)**	**¿Dónde? (páginas)**
1. IDENTIFY:	1. IDENTIFICAR:		
1.1. Problem statement and opportunity	1.1. Planteamiento del problema y oportunidad	S	1-2, 23
1.2. Constraints (standards, codes, needs, requirements & specifications)	1.2. Toma en consideración de los condicionantes (normas técnicas y regulación, necesidades, requisitos y especificaciones)	S	3
1.3. Setting of goals	1.3. Establecimiento de objetivos	S	2-3, 21-23
2. FORMULATE:	2. FORMULAR:		
2.1. Creative solution generation (analysis)	2.1. Generación de soluciones creativas (análisis)	S	24-38
2.2. Evaluation of multiple solutions and decision-making (synthesis)	2.2. Evaluación de múltiples soluciones y toma de decisiones (síntesis)	S	43-44
3. SOLVE:	3. RESOLVER:		
3.1. Fulfilment of goals	3.1. Evaluación del cumplimiento de objetivos	S	45-46
3.2. Overall impact and significance (contributions and practical recommendations)	3.2. Evaluación del impacto global y alcance (contribuciones y recomendaciones prácticas)	S	45-46

A mis padres, a mi hermana, a mi cuñado y a mi abuela:

gracias por ayudarme, cuidarme, brindarme su apoyo día tras día.

Gracias a vosotros soy la persona que soy hoy.

A mis amigos:

gracias por acompañarme en este camino y por sacarme siempre una sonrisa.

A mis tutores:

gracias por guiarme y aconsejarme de la mejor manera.

Gracias a todos de corazón.

Índice general

I Memoria

1. Introducción **1**
- 1.1. Motivación . . . 1
- 1.2. Objetivos . . . 2
- 1.3. Limitaciones . . . 3
- 1.4. Organización de la memoria . . . 4

2. Marco Teórico **5**
- 2.1. Concepto de Inteligencia Artificial . . . 5
  - 2.1.1. Machine Learning . . . 7
  - 2.1.2. Deep Learning . . . 8
- 2.2. Procesamiento de Lenguaje Natural (NLP) . . . 10
  - 2.2.1. Funcionamiento . . . 11
  - 2.2.2. Evaluación de modelos . . . 13
- 2.3. Retrieval-Augmented Generation (RAG) . . . 15
  - 2.3.1. Funcionamiento . . . 16
  - 2.3.2. Aplicaciones . . . 18
  - 2.3.3. Beneficios . . . 18

3. Metodología **21**
- 3.1. Creación de la base de datos . . . 21
- 3.2. Plataformas utilizadas . . . 22
- 3.3. Librerías empleadas . . . 22

4. Proceso de creación de un asistente médico con RAG **23**
- 4.1. Introducción . . . 23
- 4.2. Desarrollo y evolución de la aplicación . . . 24
  - 4.2.1. Fase inicial del proyecto . . . 24
  - 4.2.2. Segunda fase del proyecto . . . 29
  - 4.2.3. Tercera fase del proyecto . . . 32

5. Implementación y Resultados obtenidos **37**
- 5.1. Creación de una interfaz de usuario . . . 37
- 5.2. Métricas empleadas en la evaluación de los sistemas . . . 39
- 5.3. Métricas utilizadas en el análisis de los modelos empleados . . . 40
- 5.4. Evaluación de los resultados obtenidos . . . 40
  - 5.4.1. Análisis con métricas objetivas . . . 40

5.4.2. Aspectos analizados en los modelos empleados 42
5.4.3. Elección del sistema a emplear 43
5.4.4. Análisis subjetivo 43

6. Conclusiones y líneas futuras **45**
6.1. Conclusiones 45
6.2. Líneas futuras 46

Bibliografía **47**

II Anexos
0.1. Rúbrica empleada en la evaluación del modelo 51

Índice de figuras

2.1. Esquema machine learning 7
2.2. Esquema Deep Learning 8
2.3. Esquema Red Neuronal 9
2.4. Jerarquía de los distintos campos expuestos 15
2.5. Diagrama funcionamiento RAG básico 16
2.6. Esquema carga en base de datos vectorial 18

4.1. Estructura documento empleado en fase 1 25
4.2. Diagrama de Búsqueda Semántica 26
4.3. Diagrama de la Generación de la Respuesta 28
4.4. Tabla manual empleado en la segunda fase 29
4.5. Diagrama carga de datos fase 2 30
4.6. Diagrama del sistema de la fase 2 31
4.7. Manual empleado en fases 2 y 3 32
4.8. Diagrama del funcionamiento del overlap 33
4.9. Diagrama sistema fase 3 35

5.1. Interfaz de usuario final 38
5.2. Resultados medios tras la evaluación 44

Índice de tablas

5.1. Resultados obtenidos con métricas objetivas 41
5.2. Resultados del análisis de los modelos 42

1. Rúbrica empleada en la evaluación del modelo 51

Listado de siglas empleadas

BERT Bidirectional Encoder Representations from Transformers.

CIE Clasificación Internacional de Enfermedades.

CNO Clasificación Nacional de Enfermedades.

CSS Cascading Style Sheets.

DL Deep Learning.

FAISS Facebook AI Similarity Search.

GPT Generative Pre-Trained Transformer.

HTML HyperText Markup Language.

IA Inteligencia Artificial.

LLM Large Language Model.

MVC Modelo Vista Controlador.

NLP Natural Language Processing.

OMS Organización Mundial de la Salud.

RAG Retrieval-Augmented Generation.

SNC Sistema Nervioso Central.

Parte I

Memoria

Capítulo 1

Introducción

1.1. Motivación

En los últimos tiempos, la inteligencia artificial (IA) ha emergido como una de las tecnologías más transformadoras de nuestra era, redefiniendo los límites de lo que las máquinas y sistemas automatizados pueden lograr. Desde los primeros programas diseñados en la década de 1950, donde la tarea que ocupaban se centraba principalmente en aspectos específicos como jugar a juegos de mesa o resolver problemas matemáticos complejos. Más adelante, entre los años 1960 y 1980 surgió el enfoque de los sistemas expertos, que consistía en programar conocimiento especializado en una determinada área para que la máquina pudiera tomar decisiones basadas en ese conocimiento. La prueba de estos sistemas se realizaba mediante preguntas realizadas por un usuario al sistema que, una vez recibía una respuesta, era evaluada por el usuario en base a su utilidad y precisión. [1]

Uno de los avances importantes en el contexto de la IA ha sido el desarrollo de algoritmos cada vez más sofisticados y potentes, permitiendo que las máquinas aprendan de forma autónoma a través de análisis de grandes cantidades de datos, lo cual es conocido como *machine learning*. Hoy en día, gracias a esto, los modelos son capaces de reconocer patrones, realizar predicciones y tomar decisiones basadas en información previa. Además, se han logrado avances en ámbitos como el reconocimiento de voz y de imágenes o el Procesamiento de Lenguaje Natural/*Natural Language Processing*(NLP) gracias a la utilización de redes neuronales artificiales y algoritmos de aprendizaje profundo o *deep learning*, algo que permite que los modelos emulen el funcionamiento del cerebro humano y gestionen grandes bases de datos de manera más eficiente.

A lo largo de los años, una gran cantidad de empresas alrededor del mundo han enfrentado y siguen enfrentando el desafío de mejorar la eficiencia en sus procesos y cumplir con los plazos establecidos en la ejecución de proyectos. Es por eso que las herramientas relacionadas con la inteligencia artificial que están surgiendo en la actualidad se están revelando como un apoyo fundamental para una gran cantidad de organizaciones.

Durante mi estancia en la empresa NTT DATA S.L, en mi periodo de prácticas en el departamento de inteligencia artificial, he estado colaborando con compañeros en diversos proyectos que se iban planteando día a día y he podido comprobar de primera mano el enorme impacto que está teniendo la IA en el desarrollo de procesos y resolución de problemas que una empresa afronta cada

semana. He podido apreciar como gracias a la implementación de herramientas de IA generativa en las tareas que desempeñan los trabajadores en su día a día no solo permite que la tarea se cumpla en menos tiempo, sino que también libera a los trabajadores de realizar tareas tediosas, aumentando así su productividad en sus respectivos puestos de trabajo. Por ejemplo, en labores repetitivas y de alto volumen como el análisis y la gestión de grandes cantidades de datos y documentos, donde cada vez es más común ver como se implementan asistentes virtuales que facilitan este tipo de tareas.

En este proyecto se pretende mostrar como, con la ayuda de la IA, se pueden automatizar una gran cantidad de procesos que en otras circunstancias resultarían tediosos y lentos para las personas. En este caso en concreto se va a implementar un asistente virtual que sea capaz de poder realizar diagnósticos previos a la solicitud de una consulta médica. De esta forma el paciente, realizando una explicación acerca de los síntomas que tiene, podría recibir una primera evaluación acerca de lo que le puede estar ocurriendo.

En caso de que un asistente como este fuera desplegado, tendría el potencial de ayudar a reducir la congestión que sufren los centros de salud en España. Realizando evaluaciones previas a una hipotética visita, se optimizaría la gestión de recursos médicos, dando prioridad a casos más urgentes respecto a situaciones que pueden suponer una visita innecesaria.

Todo esto teniendo en cuenta que la intención de la introducción del asistente no es la de sustituir a un médico, ya que la evaluación que pueda hacer un doctor siempre va a ser más correcta que la del asistente, sino ayudar a que a este pueda atender a los pacientes de forma más eficiente, tratando de lograr que los pacientes sean un poco más conscientes de sus situación de cara a tomar la decisión de ir a la consulta médica.

1.2. Objetivos

Tanto el NLP, a la hora de facilitar la comunicación entre las personas y las máquinas, como la técnica de *Retrieval-Augmented Generation* (RAG), que agiliza la extracción de información a partir de los documentos, representan avances significativos en la tecnología que, con el paso del tiempo, se demuestra que están siendo cruciales para facilitar las tareas diarias que realizan numerosas empresas.

En este proyecto se explicará el funcionamiento de los sistemas basados en RAG y la utilidad que pueden tener a la hora de procesar y extraer información relevante a partir de grandes cantidades de datos. El objetivo principal es diseñar un asistente virtual que implemente un sistema que, contando con una base de datos de gran tamaño, sea capaz de poder responder a preguntas que estén relacionadas con el contenido de esa gran cantidad de datos de una forma eficaz.

En este caso, se va a implementar un asistente médico que pueda realizar la labor de responder a consultas que puedan tener las personas sobre su salud de una manera clara y precisa. Para proporcionar un contexto que permita al asistente responder a las preguntas, se cuenta con manuales extensos que cuentan con una gran cantidad de explicaciones acerca de los síntomas que provoca cada tipo de enfermedad.

Como objetivos específicos durante el desarrollo del proyecto se pretenden conseguir los siguientes aspectos:

- Estudiar los diversos sistemas creados para acometer la creación del asistente mencionado.
- Construir y analizar las bases de datos empleadas en las distintas fases del proyecto, así como el procesado de estas.
- Detallar el flujo que sigue el programa internamente desde la recepción de la consulta del usuario hasta la generación de una respuesta a esta.
- Implementar una interfaz de usuario para conseguir que la interacción entre la persona y el sistema RAG sea más cómoda.
- Evaluar el rendimiento del sistema con las métricas pertinentes.
- Seleccionar el sistema que proporcione un mejor rendimiento.

Con todo ello se tratará de mostrar, además del funcionamiento interno de los sistemas basados en *Retrieval-Augmented Generation*, como estos pueden resultar tremendamente útiles para el desarrollo y cumplimiento de una gran cantidad de trabajos cotidianos en diversos sectores.

1.3. Limitaciones

Durante la creación y desarrollo de las bases de datos que iban a ser utilizadas en el sistema, se ha encontrado una considerable limitación a la hora de acceder a datos médicos e historiales de diagnósticos debido a la ley de protección de datos. Esto impide el acceso a la historia clínica del paciente, con la que se debe guardar privacidad en todo momento, algo que restringe totalmente el uso de casos reales para la formación de las bases de datos.

A pesar de ello, se ha encontrado una alternativa viable a esto mediante el uso de manuales sobre la clasificación internacional de enfermedades (CIE), que proporciona ejemplos de diagnósticos en los que se detallan los síntomas que tienen los pacientes cuando sufren algún tipo de enfermedad, todo ello sin mencionar datos sensibles sobre las personas afectadas. Esto ha resultado de gran ayuda para proporcionar un contexto adecuado a los modelos utilizados, de manera que estos sean capaces de generar de respuestas que se ajusten a las consultas del usuario.

1.4. Organización de la memoria

La memoria constará de 5 capítulos:

- **Capítulo 1:**

  Se introduce el contexto del proyecto, el problema a resolver y los objetivos del proyecto.

- **Capítulo 2:**

  Realiza una introducción teórica al marco de la inteligencia artificial, desglosando los campos hasta llegar a la técnica que se tratará en el trabajo.

- **Capítulo 3:**

  En esta parte describen los documentos que conforman la base de datos empleada en el desarrollo del proyecto, así como las plataformas y librerías que se han utilizado.

- **Capítulo 4:**

  Detalla el proceso de creación del sistema desarrollado, exponiendo las distintas fases por las que ha ido pasando el trabajo.

- **Capítulo 5:**

  Se exponen la implementación final del sistema, las diferentes métricas empleadas en la evaluación, así como los resultados obtenidos durante esta.

- **Capítulo 6:**

  Se establecen unas conclusiones tras realizar diferentes análisis del sistema desarrollado, además de proponer nuevas líneas de desarrollo e investigación futuras del proyecto.

Capítulo 2

Marco Teórico

2.1. Concepto de Inteligencia Artificial

La inteligencia artificial es un campo de la ciencia computacional, cuya principal misión es crear máquinas capaces de realizar tarea que típicamente necesitaban de inteligencia y capacidad de razonamiento humanos. Este término se refiere a lograr que las máquinas puedan realizar tareas de forma independiente, como la extracción de características complejas a partir de los datos, el aprendizaje a partir de estos y la adaptación su comportamiento a los diferentes contextos para los que sean requeridas.

Puede parecer, por el poco protagonismo que tenía la IA en la vida de la mayoría de las personas hasta hace un par de años, que se trata de un invento muy reciente. Sin embargo, este concepto lleva siendo investigado y trabajado desde el año 1950, cuando Alan Turing, quien es conocido como uno de los padres de la ciencia de la computación, publicó un artículo seminal sobre este tema llamado 'Computing Machinery and Intelligence', en el que el matemático se preguntaba si las máquinas eran capaces de pensar y en la que además se realizó la llamada Prueba de Turing, que consistía en que un ser humano mantuviera una conversación con un computador y otra persona, pero sin saber quién de los dos conversadores era realmente una máquina, de esta manera el usuario debía evaluar si podía distinguir al humano de la máquina sin ningún tipo de sesgo. En caso de que no se pudiera distinguir, el ordenador habría pasado la prueba. [1]

El desarrollo formal de la IA comenzó con la conferencia de Dartmouth en 1956, organizada por John McCarthy. Este evento es considerado el evento que marcó el comienzo oficial de la inteligencia artificial como campo unificado y que sentó un precedente para las décadas de investigación que vendrían después [1]. En la actualidad se puede apreciar el enorme progreso que ha tenido la IA, se puede ver tanto en la creación de modelos como es GPT creado por OpenAI, que ha supuesto un cambio radical en una gran cantidad de tareas como la generación de texto, la búsqueda y síntesis de información, o la traducción.

Esta aplicación, junto con otras cuantas están convirtiendo la inteligencia artificial en una herramienta prácticamente esencial para la optimización de procesos en muchos ámbitos, como pueden ser:

La Salud: La implementación de la IA está suponiendo grandes cambios en el sector de la salud ya que gracias a esta se ha conseguido mejoras considerables en aspectos como la Biomedicina. La gran precisión y rapidez a la hora de analizar imágenes es de gran ayuda a la hora de realizar diagnósticos en enfermedades como el cáncer, enfermedades cardiovasculares y oculares o enfermedades neurodegenerativas, entre otras.

Un ejemplo de este impacto que ha habido en la biomedicina puede ser el artículo *Towards Generalist Biomedical* AI [2], un proyecto de investigación en cual se presenta MultiMedBench, un modelo de inteligencia artificial capaz de realizar hasta 14 tareas diversas como dar respuesta a preguntas médicas, interpretación de imágenes médicas o generación y resumen de informes relacionados con la radiología, entre otras.

Además, se expone Med-PaLM Multimodal, una prueba de concepto para un sistema biomédico basado en IA que codifica e interpreta datos biomédicos como el lenguaje clínico o imágenes médicas. Como resultado se obtuvo que en una clasificación de 246 radiografías, los médicos seleccionaron los informes realizados por el modelo por encima de otros producidos por radiólogos en un 40,50 % de los casos, algo que sugiere que en un futuro podría resultar útil en este ámbito. [2]

La Industria: En los últimos años es cada vez más común encontrar industrias que aplican la inteligencia artificial en alguno de sus departamentos. Por ejemplo, en la industria automotriz, empresas como Ford cuentan con robots capaces de percibir, razonar y resolver problemas. Por otra parte, la IA también resulta muy importante en la automatización de procesos, optimizando tiempo y recursos de la empresa. Así como en otros aspectos como la mejora en la exactitud y la eficiencia de los sistemas de información. [3]

La Seguridad: Uno de los campos en los que se puede apreciar la influencia de la IA es la seguridad. Aplicaciones como el reconocimiento facial se están viendo potenciadas por algoritmos de aprendizaje profundo, haciendo posible reconocer a personas que se encuentran en movimiento o rodeadas de gente. Esto es especialmente utilizado en ámbitos donde es necesario llevar un control de el aforo o localizar personas. Además de otras herramientas que analizan el comportamiento humano, que son de gran utilidad en lugares donde es necesario garantizar la seguridad de las personas. [4]

El Entretenimiento: Otro de los sectores en los que se puede encontrar el efecto que está teniendo la inteligencia artificial es en las aplicaciones destinadas al consumo de contenido, como pueden ser Netflix o HBO, en las que las recomendaciones que estas aplicaciones le dan a su cliente están basadas en modelos de IA que analizan lo que el propio usuario ha consumido anteriormente. De la misma manera funcionan aplicaciones de reproducción de música como Spotify o Apple Music, que almacenan esa información que el consumidor proporciona para anunciarle en el futuro nuevas canciones o álbumes de los artistas que suele escuchar. [1]

2.1.1. Machine Learning

Uno de los pilares fundamentales de la inteligencia artificial es el *machine learning* (ML). Una división dentro de este campo en la que toman protagonismo la inteligencia artificial y la estadística. Está centrada en el desarrollo de algoritmos que permite a las máquinas extraer conocimiento a partir de grandes cantidades de datos. Este enfoque consigue que modelos sean capaces de realizar tareas de forma autónoma.

Los modelos de ML más exitosos son aquellos en los que se crean procesos en los que se realiza una toma de decisiones automática a partir de la generalización partiendo de ejemplos conocidos. Este marco es conocido como aprendizaje supervisado.

En él, el usuario proporciona al algoritmo una variedad de entradas y de salidas deseadas, el objetivo que tiene el modelo es el de encontrar una forma de producir la salidas deseadas que se le han dado a partir de las entradas con las que cuenta. Mediante este proceso, el algoritmo es capaz de crear una salida para una entrada que no ha visto antes sin necesidad de ayuda humana.

Por otra parte encontramos los modelos de aprendizaje no supervisado. En estos solo muestran datos a la entrada del modelo, sin proporcionar ninguno a la salida de este. En este caso, a pesar de que hay unas cuantas aplicaciones de este método que han resultado exitosas, son más difíciles de comprender y evaluar.

En ambos casos, tanto en el aprendizaje supervisado como en el no supervisado, resulta muy importante tener una buena representación de los datos de entrada, de manera que el modelo pueda entenderlos y procesarlos de una manera adecuada. [5]

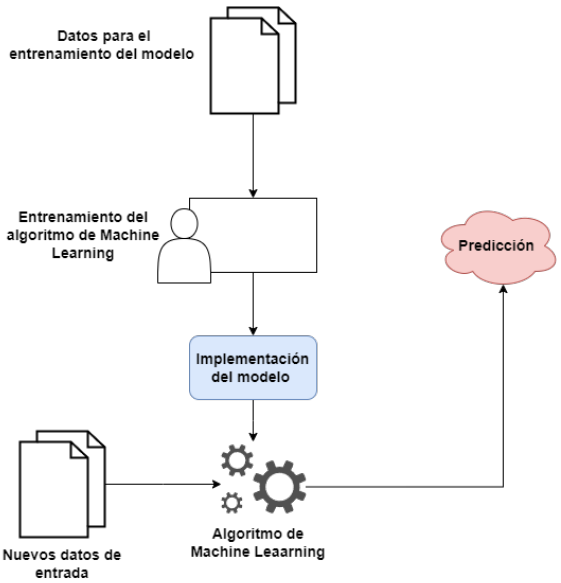


Figura 2.1: Esquema machine learning

En el esquema de la Figura 2.1 se explica el funcionamiento del *machine learning*. En primer lugar, se debe poseer una base de datos lo suficientemente grande a la vez que bien estructurada y de la cual se deben extraer características previamente para que el algoritmo de ML sea capaz de aprender de los datos y generalizar de manera correcta.

La base de datos se suele dividir en 3 partes: la parte de datos destinada al entrenamiento del modelo de ML que estará compuesta de entre un 60-80 % del set de datos total, en esta parte la misión del algoritmo será aprender las características de los datos en las qeu poder basarse para realizar predicciones futuras. Por otro lado, encontramos la parte de datos destinada a validación que será de alrededor de un 10-20 % del total y con la que se evaluará la precisión que tiene el modelo en cada iteración del entrenamiento.

Por último, si se obtienen unos resultados satisfactorios en la fase de validación, el modelo se someterá a la fase de test. Para esta se suelen destinar entre un 10 % y un 20 % de la base de datos. Es la fase en la que se emplearan datos completamente nuevos para el modelo y que permitirán evaluar la precisión que tiene el modelo de una manera imparcial una vez finalizado el entrenamiento.

2.1.2. Deep Learning

El aprendizaje profundo o *deep learning* (DL) es una técnica avanzada dentro del *machine learning* cuyo objetivo es conseguir que los ordenadores sean capaces de extraer conceptos más abstractos o complejos de los datos. El *deep learning* permite alcanzar este objetivo haciendo que los computadores sean capaces de descomponer datos complejos en otros más pequeños y a la inversa, consiguiendo al final del proceso, que los modelos sean capaces de actuar se una forma similar a la que actuar el cerebro de una persona humana [6].

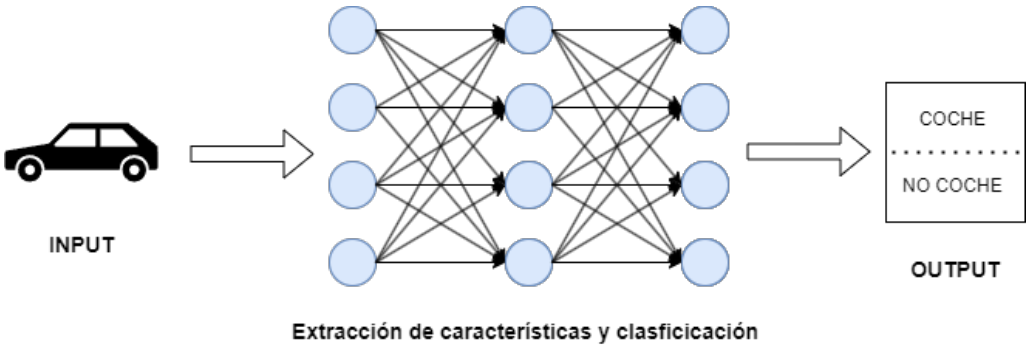


Figura 2.2: Esquema Deep Learning

Este proceso esta basado en las redes neuronales, las cuales están formadas en su interior por neuronas, unas unidades de procesamiento que realizan una serie de multiplicaciones y sumas con los datos que reciben y consiguen extraer características más complejas de las entradas durante su análisis. [7]

Estas neuronas forman las capas de una red neuronal, los datos que son recibidos como entrada van transmitiéndose entre las neuronas de diferentes capas, de manera que la salida de una neurona puede ser la entrada de la siguiente. Las neuronas tienen un peso asociado que influye en la manera en la que aprende la red y en la capacidad de generalizar de esta. Una vez la información ha sido transmitida por todas las capas finalmente llega a la capa de salida, donde se obtendrá un resultado, que será comparado con el esperado. [7]

Una vez se han comparado ambos resultados se obtiene una diferencia entre ambos, que recibe el nombre de error o pérdida, esta diferencia es cuantificada y se utiliza para ajustar el peso que tiene cada neurona en un proceso que recibe el nombre de ”Backpropagation”. Este proceso será repetido muchas épocas de manera que la red vaya siendo cada vez más precisa. [6]

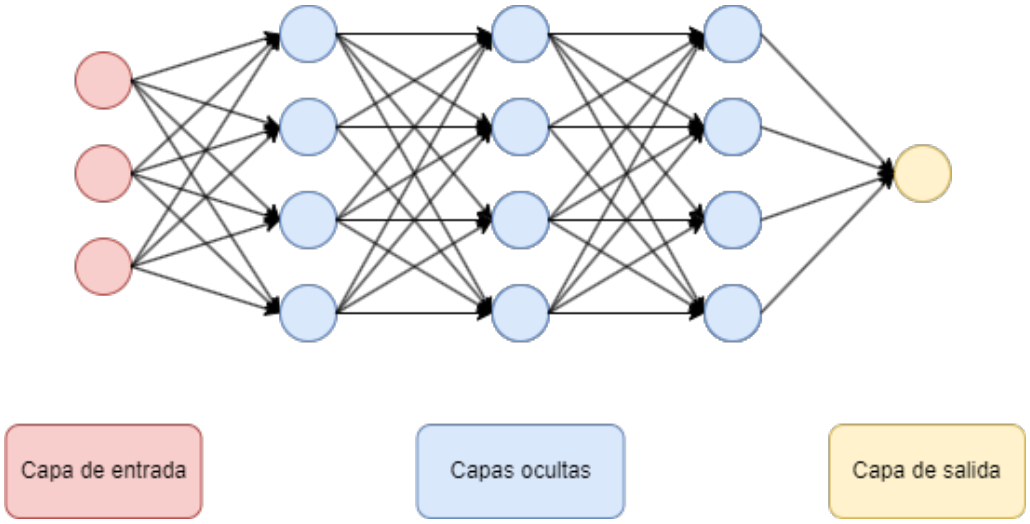


Figura 2.3: Esquema Red Neuronal

En el esquema podemos ver las capas que conforman una red neuronal: la capa de entrada, que es la capa que recibe los datos de entrada. En segundo lugar las capas ocultas de la red, que pueden ser una o varias, es donde ocurre la mayor parte del procesamiento de los datos ya que es donde se encuentran gran parte de las neuronas de la red y donde se ajustarán una mayor cantidad de pesos, por lo que esta capa resulta muy importante en la red. Por último encontramos la capa de salida de la red, que es la que proporciona el resultado de la red y la que dará la información para que se evalúe la precisión del modelo.

Las influencia del *deep learning* en aspectos cotidianos en la vida de las personas puede parecer algo poco notable. Sin embargo, esta rama de la Inteligencia Artificial está teniendo una intervención bastante significativa y positiva en ciertos campos. Algunas de las aplicaciones más comunes del *deep learning* son:

Análisis y reconocimiento de imágenes: el tratamiento de imágenes es uno de los aspectos en los que más se aplica el *deep learning*. Suele estar demandado en campos como los sistemas seguridad en los que se emplea reconocimiento facial o en análisis de imágenes médicas [8]. Ya que que el DL es muy útil en temas como el reconocimiento de personas, objetos, símbolos, números o señales entre otros. [6]

Procesado de Lenguaje Natural: el NLP es otro de los sectores en los que el *deep learning* ha tenido un gran impacto, ya que ha mejorado notablemente la capacidad que tienen los modelos para comprender y generar lenguaje humano. La traducción automática, el reconocimiento de voz y de texto, el análisis de sentimientos o los asistentes virtuales son aplicaciones en las que podemos ver la influencia del aprendizaje profundo. [9]

Automatización de Procesos: también esta siendo muy útil en la procesos en los que es las máquinas se encuentran cada día con tareas como la manipulación de datos, la extracción de información, la detección de ciertos patrones para realizar predicciones, además de que ha permitido que los sistemas automatizados sean capaces de adaptarse a nuevos entornos que se encuentran en constante cambio. [6]

Salud: el DL ha mejorado considerablemente la capacidad de diagnóstico médico que tienen las máquinas, ya que, del mismo modo que en en el reconocimiento de personas. La mejora en el tratamiento de imágenes que ha generado el *deep learning* ha beneficiado también el análisis de imágenes médicas como las ecografías, radiografías o resonancias magnéticas, entre otras. [8]

2.2. Procesamiento de Lenguaje Natural (NLP)

El procesamiento de lenguaje natural es una técnica que ha sido diseñada con el objetivo de utilizar un lenguaje natural para reducir la distancia, en lo que a comunicación se refiere, entre una persona y un computador. Esta rama tiene como objetivo hacer que los ordenadores sean capaces de poder comprender y actuar en función de las oraciones que le sean proporcionadas por parte de las personas. [9]

El NLP facilita el desarrollo de modelos que realicen tareas relacionadas con el lenguaje humano. Esto supone una gran ventaja para los humanos al no tener que aprender un medio de interacción adicional para comunicarse con un ordenador, facilitando así el procesado de datos así como el intercambio de información.

El origen del NLP se remonta a la Segunda Guerra Mundial, concretamente en el Experimento de Georgetown, el cual consistió en la traducción automática de más de sesenta oraciones rusas al inglés, un trabajo que realizó IBM junto con la Universidad de Georgetown. [10]

Otro de los sistemas de procesamiento de lenguaje que resultaron bastante exitosos fue SHRLDU, creado por Terry Winograd en 1972. Este consistía en la implementación de un sistema que trabajaba en un marco de vocabulario distinto. El funcionamiento de este sistema consistía en que el usuario podía mantener una conversación, así dar instrucciones en un lenguaje coloquial, algo

que el programa era capaz de interpretar y esta realizaba la acción que se le había solicitado. [11]

En la actualidad, podemos ver como el impacto de el procesamiento de lenguaje natural potencian aplicaciones tan demandadas como los asistentes de voz. Un ejemplo muy famoso de estos asistentes es Siri, creado por Apple, reconocido a nivel mundial por ser uno de los primeros asistentes exitosos de procesamiento de lenguaje. Este asistente inteligente es capaz de interpretar comandos de voz, captar palabras clave que utilicen los hablantes y, posteriormente, ejecutar las instrucciones que estos le hayan indicado. Siri es capaz de, a partir de las indicaciones dadas por las personas, recoger información de nuestro teléfono o de internet para dar una respuesta de forma rápida y efectiva a nuestras consultas. [12]

Otro de los avances más famosos en el contexto del NLP son los diversos modelos de lenguaje que ha diseñado OpenAI como GPT, que se ha convertido para muchas personas en una herramienta muy útil para la recopilación de información, la consulta de cualquier tipo de información o la generación de cualquier tipo de contenido entre otras, hacen de GPT una creación muy versátil capaz de ayudar a una gran cantidad de personas en la actualidad.

2.2.1. Funcionamiento

La pieza clave del Procesamiento de Lenguaje Natural se encuentra en el procesamiento de textos, que es un paso esencial para dar sentido al lenguaje humano. El proceso de descomponer textos más grande sen partes más pequeñas y manejables hace que sea mucho más sencillo para los computadores el hecho de analizar los datos y poder actuar en función de estos de una manera eficiente. Esto se consigue con los siguientes procedimientos:

Tockenización: esta técnica consiste en dividir un texto más grande en partes más pequeñas que reciben el nombre de tokens, que normalmente suelen ser palabras. Esto resulta muy importante ya que permite que el ordenador trate las palabras como elementos individuales, lo que permite que se analice el texto con una mayor eficacia. [13]

Eliminación de palabras vacías: durante el análisis hay palabras que aparecen de manera frecuente en el texto pero que no aportan información que sea importante de cara a la obtención de un resultado, estás palabras pueden ser ”de”, ”la”, ”que”, ”lo”, entre otras. La eliminación de estás palabras suele ser algo común, ya que ayuda a que el modelo se centre el palabras que sean más importantes de cara a la obtención del resultado. [14]

Lematización: este procedimiento es utilizado en los sistemas NLP para reducir las palabras a sus formas básicas o raíces, a las cuales les aplica un significado particular. Esta técnica es utilizada para acortar la búsqueda de palabras. Con esto se consigue que las distintas formas de una palabra se traten como si fueran una misma, algo que simplifica el análisis del texto al agrupar palabras similares. [15]

Conversión de palabras: se trata de convertir todo el texto a minúsculas antes de que este sea analizado. La conversión de todas las letras a minúscula consigue que todas las palabras sean tra-

tadas de la misma manera, algo que ayuda a normalizar los datos antes de el análisis.

Manejo de signos de puntuación: los caracteres especiales y signos de puntuación suelen eliminarse o tratarse de formas específicas en función de la tarea para la que se requiera el sistema. Estos, dependiendo del contexto, pueden no contener información significativa, algo que hace que a menudo se eliminen para simplificar el análisis del texto, como sucede con las palabras vacías.

Cuando el texto ya se encuentra en un formato manejable, se extraen las características principales. Estas suelen ser palabras clave, patrones o alguna estructura particular, que resultan de gran utilidad para que los modelos puedan a reconstruir y comprender el lenguaje humano.

Otra de las claves para llevar a cabo los sistemas de NLP se encuentra en la codificación del texto, ya que los ordenadores solo son capaces de procesar información numérica, por lo que es necesario codificar los textos para que estos puedan ser procesables.

La codificación consiste en representar el texto en vectores númericos de forma que permita a los ordenadores comprender esta información, este proceso es conocido como método de *embedding* [16].

Una vez los textos son codificados, todas las palabras o estructuras que hay en ellos se convierten en representaciones numéricas, lo que permite que los modelos de NLP puedan trabajar con ellas. Algunas de las técnicas más conocidas que son empleadas para la codificación de texto son las siguientes:

Bolsa de palabras (BoW): esta técnica consiste en crear un diccionario de representaciones de las palabras. Estas representaciones consisten en que a cada palabra la representa el número de veces que ha aparecido esa palabra en el texto. [17]

Una vez se han referenciado todas las palabras con el número de veces que se han visto, se incluyen en una lista en la que BoW destaca las palabras clave que definen el contexto del documento. Esto simplifica las tareas de análisis, que facilita el trabajo de los ordenadores para extraer características de un texto.

Frecuencia de términos (TF-IDF): es otra forma de representar datos de texto. No sólo tiene en cuenta la frecuencia con la que aparece una palabra en el documento, como BoW, además analiza lo única que es esa palabra en todos los documentos de una colección, lo que hace que se asigne una mayor relevancia a las palabras que son específicas de un documento de una colección para destacar su singularidad e importancia. [18]

De esta manera, TF-IDF se centra en las palabras que diferencian un documento de otro a diferencia de BoW, que cuenta cuantas veces aparece cada palabra en un documento sin tener en cuenta su singularidad en cada uno de ellos.

N-gramas: es una forma de analizar la estructura de frases o textos. Desglosan el texto en grupos de palabras y muestran cuáles van juntas con frecuencia. Algo que ayuda a predecir qué palabra puede venir a continuación en una frase o a comprender el significado de un párrafo, lo que hace que los ordenadores comprendan mejor nuestro lenguaje y mejoren la precisión del análisis de

textos. [19]

Modelos de lenguaje: esta técnica es la más importante en el contexto del NLP. Permite que los ordenadores comprendan los matices contextuales del lenguaje. Son marcos informáticos diseñados para comprender, crear y predecir el lenguaje humano. Estos modelos, que se entrenan a partir de una gran cantidad de datos textuales, imitan la generación de textos de tipo humano al captar patrones, relaciones entre palabras y usos contextuales. [20]

Aprovechan los métodos estadísticos y probabilísticos para identificar las estructuras y significados de los textos, con habilidades como traducir idiomas o analizar sentimientos. Con esto se consigue que puedan averiguar qué palabras suelen ir después de otras, como se forman las frases y qué tiene sentido en distintos contextos. Para conseguir esto se utilizan sofisticados marcos matemáticos y patrones estadísticos que les ayudan a entender el lenguaje humano.

Uno de los ejemplo más conocidos de estos modelos son GPT, creado por OpenAI, que ya cuenta con múltiples versiones y que desde su lanzamiento ha sido utilizado en la optimización de multitud de tareas por millones de personas alrededor del mundo. Otro ejemplo muy famoso es BERT, desarrollado por Google para mejorar la comprensión de sus búsquedas y así poder proporcionar unos mejores resultados a los usuarios

2.2.2. Evaluación de modelos

Una vez los modelos han sido implementados, necesitan ser evaluados para asegurarse de que están realizando su tarea de forma correcta y cumplen con los objetivos planteados. Para ello se pueden utilizar métricas de precisión como pueden ser:

Precisión: se define como la proporción de elementos que han sido devueltos por el sistema como positivos y que son verdaderamente positivos. De esta forma se penaliza a los sistemas que devuelven demasiados resultados que en realidad no son positivos. [21]

$$\text{Precisión} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

Recall: que cuantifica la proporción de positivos que se han encontrado en comparación con la totalidad de elementos de este tipo que se deberían haber devuelto. Con esta métrica se pretende premiar la profundidad de análisis con la que el modelo ha analizado los datos. [21]

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

F1-score: consiste en combinar la precisión y el recall en una sola métrica utilizando un promedio armónico ponderado. Este valor oscila entre 0 y 1 y proporciona una visión del rendimiento de un modelo en tareas donde la distribución de clases puede hacer que las métricas mencionadas no reflejen adecuadamente la efectividad. [21] Formalmente el F1-score se calcula a través de la siguiente ecuación:

$$F1Score = (1 + \beta^2) \cdot \frac{\text{Precisión} \cdot \text{Recall}}{\beta^2 \cdot \text{Precisión} + \text{Recall}}$$

A diferencia de la media aritmética, la media armónica es menos susceptible a valores extremos y penaliza de forma más severa las discrepancias entre precisión y recall, por lo que solo se puede obtener un buen F1-Score si se mantiene un equilibrio entre ambas.

Otro aspecto importante en la evaluación de los modelos es el refinamiento, ya que si una vez se ha evaluado el modelo, este no se comporta de la manera que se quiere, se realizará un refinamiento de este.

Una de las formas más comunes de refinar un modelo es ajustar los hiper-parámetros de este, como pueden ser:

La tasa de aprendizaje (η): también conocida como *learning rate* en inglés, es un factor escalar que se utiliza para regular el ajuste de los pesos en los algoritmos que emplean un descenso por gradiente, que es el mecanismo que se suele utilizar en este tipo de modelos y que consiste en la optimización del modelo mediante la minimización del error de predicción de un modelo de aprendizaje automático.

Su función consiste en controlar cuanto se alteran los pesos de la red en respuesta al error calculado durante el *backpropagation* en cada iteración, de esta forma la tasa de aprendizaje controla lo rápido que aprende un modelo, por lo que es importante darle un valor adecuado para que este aprenda las características que hay en los datos de forma correcta y se tenga más precisión en la predicción.

Número de épocas: una época se puede describir como un ciclo completo a través de todo el conjunto de datos de entrenamiento e indica la cantidad de pasadas que el algoritmo de aprendizaje automático ha completado durante ese entrenamiento.

Batch size: es el número de parámetros que entran a un modelo durante el entrenamiento. Suele ser una fracción del conjunto total de los datos y es necesario adecuar su tamaño ya que un tamaño de *batch* muy grande puede utilizar más memoria de la necesaria y, a su vez, un tamaño demasiado pequeño puede provocar que el entrenamiento del modelo sea inestable.

Ground truth: se refiere a las etiquetas con las que se califica a los datos de entrada a un modelo, así como las salidas verdaderas. Es utilizado en el diseño de modelos para aspectos como el entrenamiento o la validación, en los cuales es utilizada como una referencia fiable con la que comparar los resultados obtenidos.

Una vez los modelos han sido creados y el resultado de su evaluación ha sido correcto, estos son implementados en el mundo real mediante aplicaciones como pueden ser los asistentes virtuales. Lo que hace que además de facilitar los trabajos para los que han sido implementados, estos sigan aprendiendo de los datos que reciben como entrada en las aplicaciones en las que han sido puestos en funcionamiento.

2.3. Retrieval-Augmented Generation (RAG)

Dentro del campo del Procesamiento de Lenguaje Natural encontramos, entre otras técnicas, RAG, que es una técnica avanzada en este contexto. Se basa en combinar la información de fuentes de datos con la generación de texto. Combina un modelo de recuperación diseñado para buscar grandes sets de datos con un modelo de lenguaje, que toma esa información y genera una respuesta.

La generación aumentada de recuperación puede mejorar la relevancia de una experiencia de búsqueda, al agregar contexto de fuentes de datos adicionales y complementando la base original del entrenamiento de un *large language model* (LLM), lo que mejora el resultado del modelo de lenguaje sin tener que volver a entrenar el modelo.

RAG es una técnica interesante para tareas como la respuesta a preguntas y la generación de contenido porque permite que los sistemas de inteligencia artificial generativa utilicen fuentes de información externas para producir respuestas más precisas y fieles al contexto. [16]

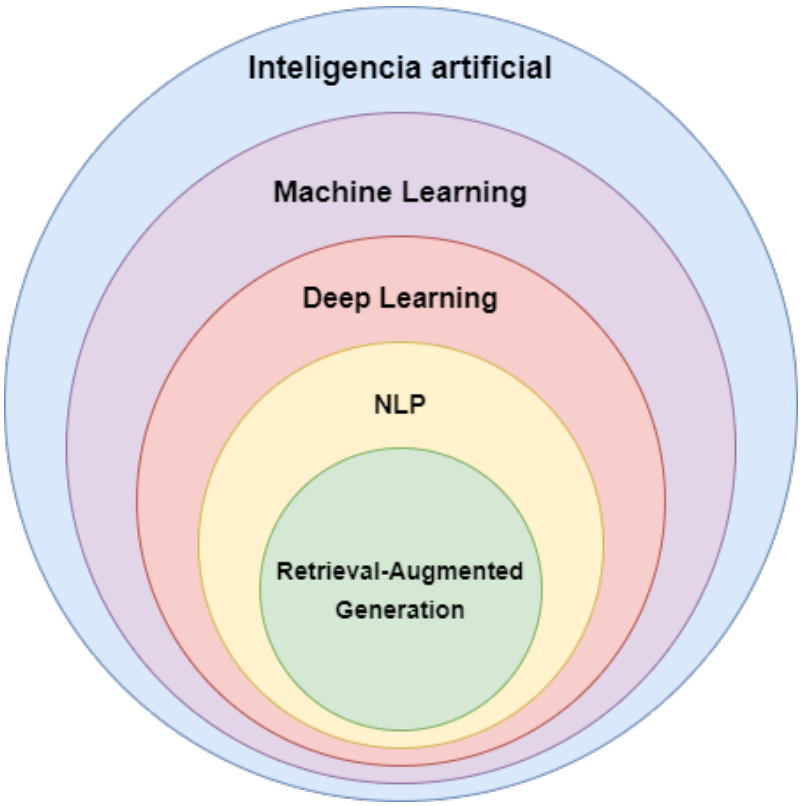


Figura 2.4: Jerarquía de los distintos campos expuestos

2.3.1. Funcionamiento

Retrieval-Augmented Generation utiliza modelos de lenguaje avanzados y técnicas de IA para llevar a cabo el proceso mencionado, que se divide en las siguientes etapas:

Recuperación de información: esta etapa comienza con la consulta por parte del usuario sobre algún aspecto determinado de la base de datos que se desea analizar. A continuación, el modelo realiza una búsqueda en una base de datos indexada o en fuentes externas y recupera la información que puede ser relevante para proporcionar una respuesta al usuario.

La recuperación dependerá de la consulta de entrada, que condicionará la búsqueda y dónde se realiza esta. Para esto se suelen utilizar técnicas que permiten realizar búsquedas de manera eficiente como son la búsqueda semántica o las coincidencias de palabras clave. La información recuperada se convierten en vectores que se almacenan en una base de datos vectorial y que serán ordenados según su relevancia para la búsqueda de entrada. [16]

El modelo de recuperación clasificará la información recuperada según su relevancia para la búsqueda de entrada, analizando las similaridades entre la consulta realizada y la información que hay en cada vector. Los documentos con las puntuaciones más altas se seleccionarán para su posterior procesamiento.

Generación de la respuesta: la información que ha sido recuperada de la base de datos se procesará de forma que ayude a la hora de generar una respuesta, la cual será creada por un modelo de generación basado en técnicas de *deep learning*, como es un LLM, que utiliza la información recuperada para generar las respuestas de texto. [16]

Las respuestas suelen ser más precisas contextualmente, ya que han sido moldeadas por la información de recuperación. Esto resulta especialmente importante en contextos donde los datos que hay en internet son escasos.

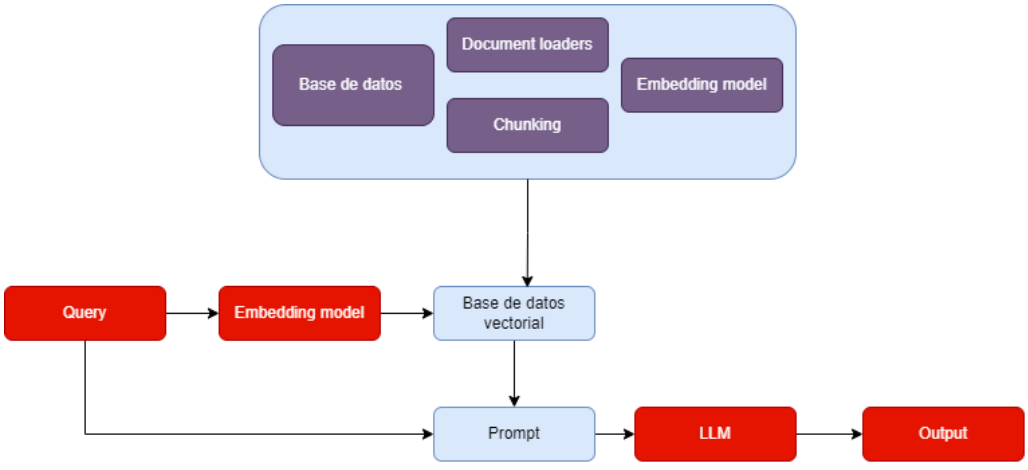


Figura 2.5: Diagrama funcionamiento RAG básico

En la figura 2.5 se presenta un esquema en el que se puede ver el flujo que sigue un sistema RAG. En primer lugar encontramos el bloque de procesamiento de documentos, que cuenta con la base de datos en la que se almacenan los documentos indexados para proporcionar información y contexto al modelo.

Además, este bloque también cuenta con *document loaders*, con los cuales se lleva a cabo el proceso de carga inicial de los documentos que se tienen como fuente de información de manera que estén en un formato adecuado y etiquetados de forma correcta.

Por otro lado, se realiza un procesado de esos datos recogidos en la etapa de *chunking*. En esta se realiza el proceso de división de los grandes documentos en partes más pequeñas (que reciben el nombre de *chunks*) de forma que se puedan manejar de una mejor manera.

En este bloque, además de los módulos ya mencionados, se encuentran los modelos de *embedding*. Su función es transformar el texto que hay en los *chunks* en representaciones numéricas de cada uno de ellos, lo que facilita el procesado de estos.

Por último, una vez se han obtenidos los vectores numéricos que representan a todos los *chunks* en los que se ha dividido la base de datos original. Estos vectores se almacenan en una base de datos vectorial, que permitirá recuperar la información de una forma más rápida cuando se realice una búsqueda de similitudes en el contenido. [16]

Por otra parte, encontramos el bloque *query*, que se refiere a las consultas que realiza el usuario relacionadas con la base de datos indexada, esta será procesada por otro modelo de *embedding* de manera que quede convertida también en un vector numérico.

Para encontrar las similaridades que hay entre lo que el usuario ha dado como entrada y lo que hay en la base de datos del sistema se realizará una búsqueda semántica en la base de datos vectorial, con lo que se obtendrán los *chunks* más relevantes.

A continuación encontramos el *prompt*, que consiste en un conjunto de instrucciones a seguir, que se pasarán, junto con la *query* del usuario y con los *chunks* recuperados, como entrada a un *large language model* (LLM). [16]

Finalmente encontramos el LLM, que tendrá la misión de generar, ciñéndose a las instrucciones recibidas, una repuesta adecuada a la consulta del usuario.

En la figura 2.6 se presenta un esquema que describe el proceso que ocurre en el bloque de procesamiento de los documentos, comentado anteriormente:

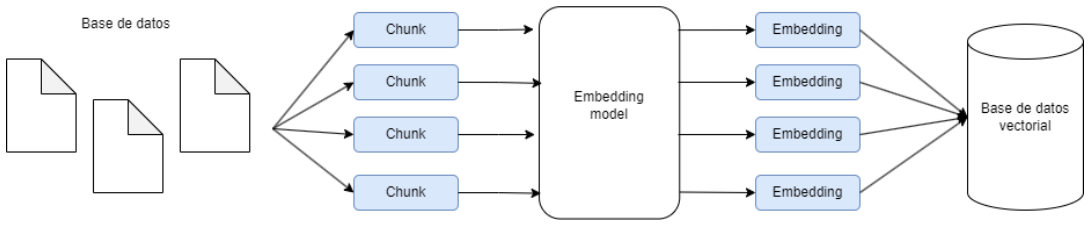


Figura 2.6: Esquema carga en base de datos vectorial

2.3.2. Aplicaciones

Existen varios campos en los que podemos ver la utilidad de *Retrieval-Augmented Generation* [16], como pueden ser:

Sistemas Question-Answer: RAG es capaz de mejorar la calidad de las respuestas en este tipo de sistemas, sobre todo en ámbitos en los que es necesario que se responda con mucha precisión.

Asistentes Virtuales: esta técnica facilita la generación de respuestas en aplicaciones como chatbots, ayudando a que estas sean más informativas y claras.

Síntesis de información: *Retrieval-Augmented Generation* puede ser utilizado para unir información proveniente de diferentes fuentes o de bases de datos muy grandes de manera que sea más fácil que el usuario pueda acceder a información más especifica.

2.3.3. Beneficios

Esta técnica tiene varios beneficios sobre los modelos de lenguaje que funcionan de forma aislada. En primer lugar, podemos apreciar que RAG permite que los modelos sean capaces de acceder a fuentes de datos externas más recientes y actualizadas de forma eficiente, ya que puede actualizar sus referencias más rápidamente.

Esto se debe a que al aportar información renovada a un modelo de RAG, esto no afectará al modelo de una forma global, solamente provocará que la base de datos indexada que había anteriormente quede sustituida o ampliada, algo que solo modificará el funcionamiento de la fase de recuperación de información, que tendrá un conjunto de datos a analizar más grande o distinto. Por tanto, no será necesario hacer un reentrenamiento del modelo para mejorar la precisión o la relevancia en las repuestas del modelo, algo que por otra parte, hará que se asuma un menor coste computacional.

Por otra parte, cuando un LLM se enfrenta a una búsqueda compleja, en ocasiones se puede producir 'alucinaciones', que son situaciones en las que un modelo genera información falsa o imprecisa en relación con la entrada proporcionada a pesar de parecer correctas.

La generación aumentada por recuperación consigue disminuir este problema ya que, al contar con un paso previo de recuperación de la información antes de proporcionar una respuesta al usuario, el modelo no depende solamente de lo que haya en la base de datos que se ha utilizado para entrenarlo, sino que también cuenta con los documentos que le han sido indexados de forma externa, ofreciendo una información añadida que puede ser de ayuda para generar una respuesta más completa y precisa.

Capítulo 3

Metodología

A continuación se desglosarán los componentes metodológicos que han sido empleados para llevar a cabo el diseño de un asistente médico virtual basado en un sistema Question-Answer en el que se ha utilizado RAG. Se explicarán tanto las bases de datos seleccionadas como las plataformas y librerias empleadas para obtener la aplicación final, examinando la función que cumple cada una de ellas en el diseño final del proyecto.

3.1. Creación de la base de datos

El objetivo que se buscaba con la base de datos es que nutriera al asistente de un contexto completo sobre medicina para que este fuera capaz de realizar respuestas completas a las consultas que los usuarios le hagan sobre enfermedades o lesiones que estén sufriendo. Para ello se precisaba una base de datos que contara con bastante información acerca de diagnósticos sobre enfermedades para que, a partir de la descripción del paciente, se pudiera relacionar lo que este comentaba con posibles dolencias.

En este caso las muestras serán Manuales de diagnósticos médicos obtenidos de las bases de datos del Ministerio de Sanidad, Servicios Sociales e Igualdad y del Instituto Nacional de la Seguridad Social. Estos documentos nos aportarán la información necesaria para poder contextualizar al modelo de forma correcta a la hora de analizar los comentarios que haga el usuario. [22] [23]

Estos documentos cuentan con explicaciones que relacionan enfermedades con sus respectivos síntomas, así como multitud de estadísticas acerca del tiempo de recuperación de cada enfermedad, el ratio de personas que la padecen, dependiendo de la edad de la persona, de su sexo y de su grupo ocupacional, para el que se aporta el código de clasificación nacional de ocupaciones (CNO-11).

Además, todo esto se relaciona con el código de clasificación internacional de enfermedades CIE-10. Esta clasificación es un sistema establecido por la organización mundial de la salud (OMS) que clasifica y codifica las enfermedades, así como una amplia variedad de signos, síntomas, hallazgos anormales, circunstancias sociales y causas externas de daños y/o enfermedad. Se compone de miles de códigos que se utilizan de forma global para clasificar las enfermedades. [22] [23]

La CIE se inició a finales del siglo XIX, se realizó una clasificación uniforme de las causas de defunción (CIE-1). Desde entonces la CIE ha evolucionado en respuesta a las cambiantes necesidades de la salud pública y la medicina. Se fue ampliando su campo de estudio incluyendo datos de morbimortalidad, epidemiología y gestión de salud. Con cada nueva edición se incorporaba una estructura más detallada y códigos más específicos para reflejar con precisión las condiciones de salud.

La CIE-10 se divide en 21 capítulos principales, cada uno de los cuales abarca una categoría amplia de condiciones de salud. Se utiliza un sistema de códigos alfanuméricos para describir condiciones de salud de manera específica. Cada código consta de una letra inicial seguida de dígitos, lo que permite una identificación precisa. La letra inicial del código indica el capítulo al que pertenece la condición, mientras que los dígitos subsiguientes reflejan una jerarquía descendente de categorías y subcategorías. Cada afección puede ser asignada a una categoría y recibir un código de hasta seis caracteres de longitud (en formato de X00.00). [22]

3.2. Plataformas utilizadas

Para el desarrollo de la aplicación se ha hecho uso de Microsoft Azure, una plataforma en la nube que cuenta con servicios que permiten a personas y empresas realizar pruebas e implementar aplicaciones a través de una red global que está gestionada por Microsoft. A través de esta se ha hecho uso del servicio Azure OpenAI, el cual permite tener acceso a los modelos de lenguaje que desarrolla la organización OpenAI.

Estos servicios han sido utilizados a través de sus distintas Interfaces de Programación de Aplicaciones, más conocidas por sus siglas en inglés como Application Porgramming Interface (API), importadas desde librerías con el editor de código desarrollado por Microsoft, Visual Studio Code.

3.3. Librerías empleadas

Langchain: esta es una librería muy utilizada para el desarrollo de aplicaciones que utilizan modelos de lenguaje. Esta biblioteca resulta muy útil para crear aplicaciones como chatbots o asistentes virtuales, ya que esta libreria permite la interacción con modelos actuales como GPT-3 o GPT-4, entre otros. Algo que resulta tremendamente útil a desarrolladores que deseen utilizar estos modelos sin necesidad de reescribir o crear grandes cantidades de código.

Azure OpenAI: esta librería suele ser utilizada por las empresas y desarrolladores para poder tratar grandes bases de datos de forma segura y fácil. En este caso, ha sido utilizada para emplear los modelos de OpenAI en el desarrollo final de la aplicación.

Ragas: es una librería creada para evaluar el rendimiento de los sistemas RAG, ya que al tratarse de sistemas de recuperación de información y generación de una respuesta, es necesario tener en cuenta varios puntos del proceso para evaluarlo de una forma correcta. *Ragas* ha sido utilizada para evaluar la precisión, relevancia y coherencia de la respuesta del modelo.

Capítulo 4

Proceso de creación de un asistente médico con RAG

4.1. Introducción

El propósito de este trabajo ha sido crear una aplicación que implemente un asistente virtual capaz de simular el comportamiento que podría tener un médico a la hora de realizar un diagnóstico a un paciente. Se ha buscado mostrar un ejemplo de como la inteligencia artificial es capaz de optimizar procesos en ámbitos como la medicina, optimizando un aspecto importante como es la atención a pacientes.

Con el programa se trata de, a partir de una descripción por parte del paciente, conseguir que el asistente médico sea capaz de poder contarle a la persona que le hace la consulta lo que puede estar sufriendo. De esta manera, las personas recibirían una primera opinión con más conocimiento sobre el tema que les pueda orientar un poco sobre lo que les ocurre. Sin embargo, el objetivo final no es el de sustituir por completo a un médico con el asistente creado, si no crear un complemento a este que pueda facilitar su trabajo.

Con la intención de formar al asistente con un conocimiento que le permita ofrecer a los usuarios unos diagnósticos precisos se ha realizado una búsqueda de bases de datos que el modelo pudiera utilizar como contexto para dar una respuesta coherente a las preguntas que se le hagan.

Este trabajo, además de lo mencionado, tiene intención de mostrar un ejemplo de funcionamiento en un caso real de una importante técnica dentro del campo de la inteligencia artificial como es RAG. Esta técnica, conforme avanza la implicación de la IA en la vida cotidiana de las personas, se está volviendo una herramienta fundamental para facilitar el trabajo de muchas personas.

4.2. Desarrollo y evolución de la aplicación

En este apartado se va a pasar a presentar el proceso de creación del asistente médico. Este proyecto está dividido en 2 partes principales: la elaboración de la arquitectura del algoritmo y la implementación de una interfaz de usuario. En la primera parte se detallará el algoritmo desarrollado para la creación del asistente y en la segunda se explicará el procedimiento seguido para la creación de la interfaz.

En primer lugar, se describirán las fases que ha seguido el proyecto antes de su versión final, los resultados obtenidos con cada una y la mejora que han supuesto con respecto a la versión anterior.

4.2.1. Fase inicial del proyecto

Con la primera fase del proyecto se buscaba realizar, ya en primera instancia, un programa robusto con la capacidad de realizar desde la primera fase las tareas que se buscaban en el proyecto final.

Análisis de la base de datos

Para poder realizar pruebas sin necesidad de ocupar un alto volumen de espacio, se cargó un documento pequeño con datos sobre diagnósticos [24], con su respectivo código CIE-10 para que se almacenara como fuente de contexto para el modelo de lenguaje. La estructura del documento era la siguiente:

TABLA DE DIAGNOSTICOS SEUP – VERSION CIE-10 (MARZO 2017)

CODIGO CIE-9-MC	DESCRIPTOR Y DEFINICION (Se mantiene en la versión CIE-10ES SEUP)	CODIGO CIE-10ES	DESCRIPTOR CIE-10ES (Se aporta para que el usuario encuentre su relación en listado general CIE-10ES)	CODIGO ADICIONAL ACCIDENTES CIE-10ES
682.9	**Absceso:** Colección supurada sin especificar su localización.	L02.91	Absceso cutáneo no especificado	
522.5	**Absceso/ Flemón dental:** Colección supurada o inflamación en encía y/o tejidos circundantes, secundaria a patología dental.	K04.7	Absceso periapical, sin sinus	
475	**Absceso/ Flemón periamigdalino:** Colección supurada o inflamación, de origen infeccioso en los tejidos adyacentes a la amígdala, en general unilateral y como complicación de una amigdalitis aguda.	J36	Absceso peritonsilar	
478.24	**Absceso/ Flemón retrofaríngeo:** Colección supurada o inflamación, de origen infeccioso,localizada en el espacio retrofaríngeo (Rx lateral cuello y/o TAC).	J39.0	Absceso retro y parafaríngeo	
995.50(E967.9)	**Abuso/Maltrato:** Sospecha de agresión sexual y/o física, existan lesiones evidentes o no.	T76.22XA	Abuso sexual infantil,sospecha contacto inicial	
959.9 (E819.9)	**Accidente de tráfico:** Paciente que presenta cualquier tipo de lesiones en relación con un accidente de tráfico, tanto como transeúnte o como ocupante de un vehículo.	T14.90	Accidente de tráfico	V89.2XXA
785.6	**Adenopatía:** Aumento de tamaño de ganglios linfáticos, sin otros signos inflamatorios, sin especificar localización.	R59.9	Adenomegalia, no especificada	
683	**Adenitis/Adenoflemón:** Tumefacción ganglionar aguda con signos inflamatorios, con o sin fluctuación, sin especificar localización.	L04.9	Adenitis aguda	
994.1 (E928.9)	**Ahogamiento:**Paciente que presenta clínica respiratoria, cardiaca o neurológica relacionada con la inmersión y asfixia en medio líquido, generalmente agua	T75.1XXA	Ahogamiento	X58.XXXA
995.3(E928.9)	**Alergia inespecífica:** Síntomas atribuidos a hipersensibilidad o reacción alérgica, sin especificar el agente desencadenante.	T78.40XA	Reacción alérgica contacto inicial	
995.0 (E928.9)	**Anafilaxia:** Dos o más de los siguientes síntomas que ocurren rápidamente tras la exposición del paciente a un alergeno sospechoso: Afectación de piel y/o mucosas; Compromiso respiratorio; Compromiso cardiovascular; Síntomas gastrointestinales persistentes: vómitos de repetición, dolor abdominal cólico.	T78.2XXA	Anafilaxia	X58.XXXA
285.9	**Anemia:** Presencia de valores de hematacrito y hemoglobina por debajo de las cifras normales para la edad del paciente, sin especificar tipo o causa.	D64.9	Anemia, no especificada	
282.60	**Anemia de células falciformes/Drepanocitosis:** Paciente que en su hemograma presenta alteración morfológica de los hematíes que pierden su forma y adoptan un aspecto semilunar	**D57.1**	Anemia de células falciformes/Drepanocitosis	
995.1(E928.9)	**Angioedema:** Edema angioneurótico (edema agudo de labios y párpados con o sin síntomas respiratorios), sin especificar el agente desencadenante.	T78.3XXA	Angioedema contacto inicial	
300.00	**Ansiedad:** Síntomas psíquicos y/o somáticos atribuidos a la presencia de temor o preocupación desligados de hechos o situaciones que los justifiquen.	F41.9	Trastorno de ansiedad	
540.9	**Apendicitis:** Inflamación apendicular sin perforación.	K35.80	Apendicitis aguda	
786.09	**Apnea/Episodio aparentemente letal (EAL):** Episodio brusco que atemoriza al observador y que se caracteriza por alguna combinación de: Apnea (central u ocasionalmente obstructiva); Cambios de color (cianosis o palidez, pero ocasionalmente eritrosis); Cambios marcados del tono muscular (generalmente	R06.00	Disnea no especificada	

Figura 4.1: Estructura documento empleado en fase 1

En la imagen podemos ver un ejemplo de como estaban distribuidas las tablas que contenían la información sobre los diagnósticos. Puesto que los códigos CIE-9 han sido reemplazados por los códigos CIE-10, la columna izquierda no será utilizada para obtener el resultado final.

En primer lugar, nos centramos en la segunda columna de la tabla, la cual nos aporta una descripción de algunas patologías. A continuación, se apunta el código CIE-10 de cada una de ellas y por último se menciona el título con el que se corresponde cada código.

Carga de la base de datos

Una vez se analizó la base de datos y se consideró que era óptima para obtener los resultados deseados, se inicia la carga del documento con los mencionados *document loaders*. Estos son implementados a partir de la librería *langchain*, importando el modulo PyPDFLoader, que nos ayudará a procesar y cargar el archivo .pdf en nuestra base de datos.

Búsqueda de similaridades

Con la base de datos indexada de forma correcta, se importó la biblioteca FAISS dentro de *langchain community*, una extensión de la biblioteca *langchain*. En primer lugar, *Facebook AI Similarity Search* (FAISS) fue de gran ayuda a poder crear un índice en el que almacenaremos los *embeddings* creados a partir de la información indexada desde el documento.

Por otra parte, también cuenta con funciones que nos permiten realizar una búsqueda eficiente en el documento indexado, tomando la solicitud hecha por el usuario (*query*) como punto de partida y basándonos en ella para encontrar los vectores más relevantes dentro de la base de datos vectorial.

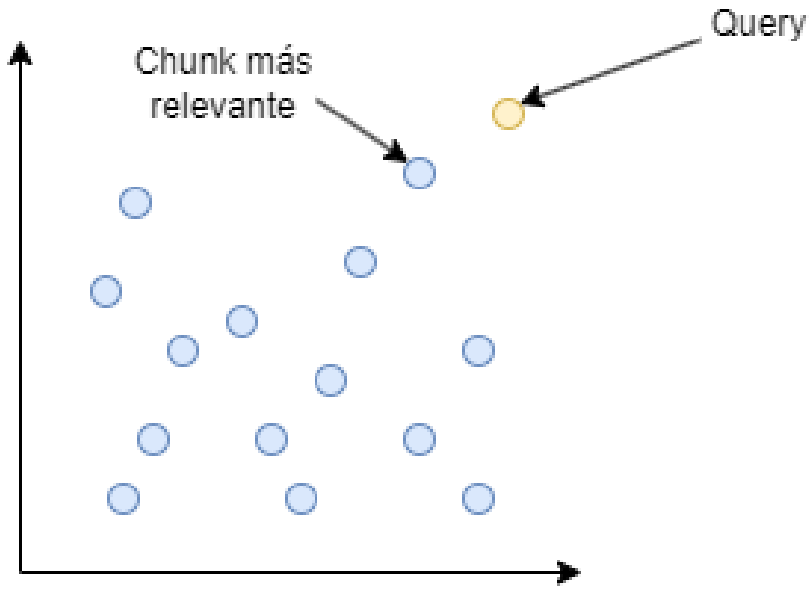


Figura 4.2: Diagrama de Búsqueda Semántica

En la imagen se representa el resultado que se obtiene al realizar la búsqueda semántica. Durante este proceso, para calcular qué documentos o partes de documentos (*chunks*) son más relevantes para la respuesta se presta mucha atención al porcentaje de similaridad que tienen la *query* y el las distintas partes en las que se ha dividido la base de datos.

Después de haber realizado la carga de la base de datos, comenzaron las primeras pruebas en la búsqueda de coincidencias del documento. Para ello se creó una consulta que contuviera información que se encontrase dentro de la fuente de contexto.

Esta sería utilizada para simular un proceso de búsqueda en un caso real, en el que también obtendríamos un porcentaje de coincidencia o *score* entre el contenido de la consulta y el que hay en el documento. El resultado obtenido fue la parte del documento en la que se había encontrado un porcentaje mayor de coincidencias junto con ese *score*.

Generación de la respuesta

Tras verificar que la información que se extraía era correcta y útil para poder generar una respuesta se creó la función *get completion*, esta función nos es imprescindible, pues es la herramienta que se utiliza para poder hacer la llamada a los modelos de OpenAI, en este caso concreto se utilizó el modelo 4-Turbo de GPT.

La función *get completion* se encarga de configurar aspectos como un contador del tiempo que se tarda en recibir la respuesta por parte de OpenAI, algo que nos resulta útil a la hora de poder realizar una evaluación en términos de eficiencia en la generación de una respuesta. Por otra parte, también tiene la misión de enviar la solicitud a la API de OpenAI, además de indicarle a esta que la función del sistema es la de ser un asistente y enviarle el *prompt*, que son las instrucciones que le habremos indicado al modelo para que este ejecute.

El *prompt* constituye un papel muy importante en la creación del sistema, ya que le indica al modelo el cual va a analizar el contexto y a generar una respuesta, cuál es su misión. Se indican instrucciones acerca de cómo debe responder, qué aspectos es importante resaltar y cómo debe estructurar el mensaje de respuesta.

Una de las claves de un buen *prompt* es que sea breve y conciso, de forma que el modelo no se desvíe de la intención principal para la que se le necesita, pero a su vez que sea explicativo, de forma que el modelo pueda entender de forma clara las instrucciones que se le dan y así pueda cumplir con lo que se necesita de él.

A continuación, se presenta el *prompt* empleado en la primera fase del proyecto para la obtención del resultado final:

Prompt

Eres un experto en medicina, con especial experiencia en realizar diagnósticos de pacientes, con un amplio conocimiento acerca de los síntomas de cada enfermedad y cómo se comportan. Un paciente te va a contar los síntomas que está padeciendo últimamente y tú como médico debes contarle que crees que le puede estar pasando según lo que comenta.

Debes contarle también el nivel de gravedad que tiene la enfermedad o enfermedades que hayas valorado que puede estar padeciendo, ya sea mucha o poca.

Por último debes recomendarle que acuda a su médico de confianza, si la gravedad de la enfermedad que te ha comentado el paciente es alta, tu mensaje debe ser con una recomendación más efusiva o insistente, si la gravedad es más bien baja simplemente recomiéndaselo por precaución, para prevenir.

Recuerda no extenderte mucho en tu explicación y utilizar un lenguaje algo coloquial, para que el paciente te pueda comprender.

El objetivo del *prompt* es, en primer lugar, contextualizar al modelo en el aspecto de la medicina, puesto que todas las consultas que va a recibir son médicas, no es necesario tenga conocimiento de otros temas. Es importante también explicarle el tipo de consulta que va a recibir.

Por otra parte, también es necesario indicarle al modelo los detalles importantes que debe transmitirle al usuario una vez analizada su consulta y extraída la información, como son la gravedad de la dolencia o si es más o menos necesario que el usuario visite a su médico de confianza. En cualquier caso, siempre se recomienda que se realice una visita al médico en caso de duda, ya que el doctor en su consulta podrá realizar un diagnóstico mucho más acertado del que pueda realizar el modelo.

Por último se realiza una llamada a la función *get completion*, que recibirá como entrada el *prompt* indicado además de la consulta hecha por el usuario y la parte del texto que guarda más similaridad con la consulta del cliente, que ha sido extraída anteriormente. Posteriormente, la función se encargará de realizar una llamada a un *large language model*). En esta llamada se proporcionarán los datos mencionados y, una vez se haya generado la respuesta a la consulta, se establecerá esta como salida del sistema.

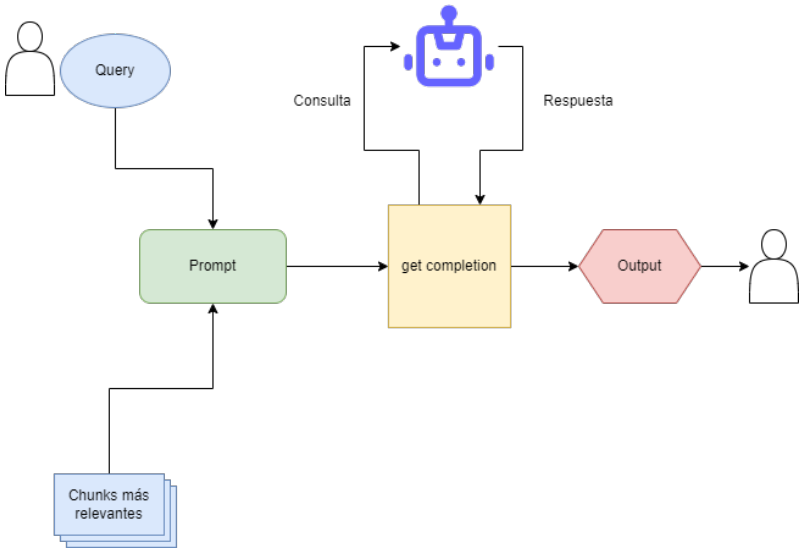


Figura 4.3: Diagrama de la Generación de la Respuesta

4.2.2. Segunda fase del proyecto

Una vez hechas las primeras pruebas en la fase inicial y habiendo comprobado que los resultados obtenidos eran correctos, se inició una segunda fase del proyecto.

Incorporación de fuentes de información más extensas

En esta segunda fase, habiendo visto el correcto funcionamiento del sistema con la base de datos más pequeña, se realizó una carga de los manuales de diagnósticos médicos más extensos. Estos nos iban a aportar verdaderamente un contexto completo que el modelo pudiera utilizar para poder dar una respuesta a consultas de cualquier tipo y no solo de unas pocas como sucedía con la base de datos anterior.

Los nuevos manuales contaban con una enorme cantidad de enfermedades, las cuales estaban distribuidas de acuerdo a los sistemas o aparatos del cuerpo humano que afectaban y por la naturaleza de estas, así como los códigos CIE-10 de cada una de estas enfermedades. Además de disponer de una gran cantidad de estadísticas relacionadas con estas enfermedades y los códigos de cada variación dentro de cada grupo de enfermedad [22] , [23].

CIE-10	DESCRIPCIÓN	GRUPO EDAD	RATIO
J01	Sinusitis aguda	26-35	0,84
J01	Sinusitis aguda	36-45	0,93
J01	Sinusitis aguda	46-55	1,25
J01	Sinusitis aguda	56-65	1,83
J02	Faringitis aguda	16-25	0,70
J02	Faringitis aguda	26-35	0,84
J02	Faringitis aguda	36-45	0,97
J02	Faringitis aguda	46-55	1,25
J02	Faringitis aguda	56-65	1,66
J03	Amigdalitis aguda	16-25	0,89
J03	Amigdalitis aguda	26-35	0,94
J03	Amigdalitis aguda	36-45	1,01
J03	Amigdalitis aguda	46-55	1,29
J03	Amigdalitis aguda	56-65	1,79
J04	Laringitis y traqueítis agudas	16-25	0,66
J04	Laringitis y traqueítis agudas	26-35	0,76
J04	Laringitis y traqueítis agudas	36-45	0,94
J04	Laringitis y traqueítis agudas	46-55	1,20
J04	Laringitis y traqueítis agudas	56-65	1,39
J05	Laringitis aguda obstructiva [crup] y epiglotitis aguda obstructiva	16-25	0,66
J05	Laringitis aguda obstructiva [crup] y epiglotitis aguda obstructiva	26-35	0,76
J05	Laringitis aguda obstructiva [crup] y epiglotitis aguda obstructiva	36-45	0,94
J05	Laringitis aguda obstructiva [crup] y epiglotitis aguda obstructiva	46-55	1,20
J05	Laringitis aguda obstructiva [crup] y epiglotitis aguda obstructiva	56-65	1,39
J06	Infecciones agudas del tracto respiratorio superior de localización múltiple o no especificada	16-25	0,60
J06	Infecciones agudas del tracto respiratorio superior de localización múltiple o no especificada	26-35	0,73
J06	Infecciones agudas del tracto respiratorio superior de localización múltiple o no especificada	36-45	0,90
J06	Infecciones agudas del tracto respiratorio superior de localización múltiple o no especificada	46-55	1,22

Figura 4.4: Tabla manual empleado en la segunda fase

En la tabla de la figura 4.4 se puede ver un ejemplo del formato que seguían las tablas que se emplearon en la creación de la base de datos de la segunda fase, como se ha adelantado en el apartado 3.1, estas tablas contienen estadísticas relacionadas con la edad de los pacientes, su sexo, grupo de ocupación, además del ratio de pacientes que padecen esta enfermedad dentro de cada grupo de los mencionados.

Carga de la base de datos

Puesto que la base de datos de esta segunda fase era mucho más grande que la de la primera fase, se optó por probar con una indexación más compleja y progresiva. Esto se debe a que con el primer documento no iba a haber problemas de indexación si se cargaba todo el documento en el índice de una vez. Pero como el objetivo en este caso era realizar una carga de bases de datos con una extensión mucho mayor, se consideró necesario implementar una manera de indexar los datos más eficiente.

Es por esto que se creó una indexación en la que los datos se fueran subiendo a un índice en lotes más pequeños. En el proceso, los lotes de información que se iban extrayendo de la base de datos se subían a un índice global en el que iban quedando almacenados todos los lotes, guardando pequeñas pausas entre cargas de lotes para evitar sobrecargas. De esta manera se consigue optimizar el proceso de carga de datos.

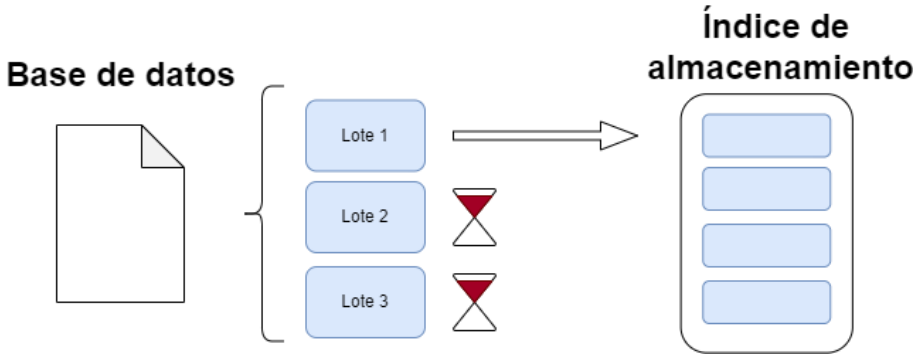


Figura 4.5: Diagrama carga de datos fase 2

Creación de un doble sistema RAG

Para este sistema, la intención era realizar un sistema en el que la consulta del cliente fuera tratada de forma independiente por dos sistemas basados en *Retrieval-Augmented Generation*. Con esto se obtendrían dos respuestas distintas y unificando el resultado de ambas en una sola, que se daría como respuesta final a la solicitud. Esto fue ideado con la intención de conseguir un resultado más completo y acertado que con un sistema de RAG más básico.

En primer lugar, se comenzó realizando una carga de las bases de datos, que serían los dos extensos manuales comentados anteriormente, por separado. A continuación, estos manuales serían indexados cada uno en un índice diferente con el proceso de carga de datos que ha sido descrito.

Una vez se realizaron ambas indexaciones y toda la información quedó almacenada en los índices correspondientes, se creó una única consulta, la cual sería utilizada para realizar una búsqueda de coincidencias en ambas bases de datos de forma independiente, obteniendo los tres fragmentos con más porcentaje de similaridad de cada documento.

Posteriormente, se realiza una doble llamada a la función *get completion*, en la que la *query* será la misma, pero el resultado de la búsqueda será diferente en cada una de las llamadas. En la primera llamada será el resultado de la búsqueda de similaridades en la primera base de datos, mientras que en la segunda llamada será el de la búsqueda en la segunda fuente de información.

Por último, se hará una tercera y última llamada a la función *get completion*, que recibirá como entrada un *prompt* en el que se le habrá especificado la recepción de tres entradas: las respuestas de las dos llamadas anteriores a la función y la consulta inicial del paciente. El objetivo de esta tercera llamada es que, valorando lo que se solicita en la *query*, se analicen ambas respuestas recibidas y se unifiquen para dar una respuesta final lo más completa posible.

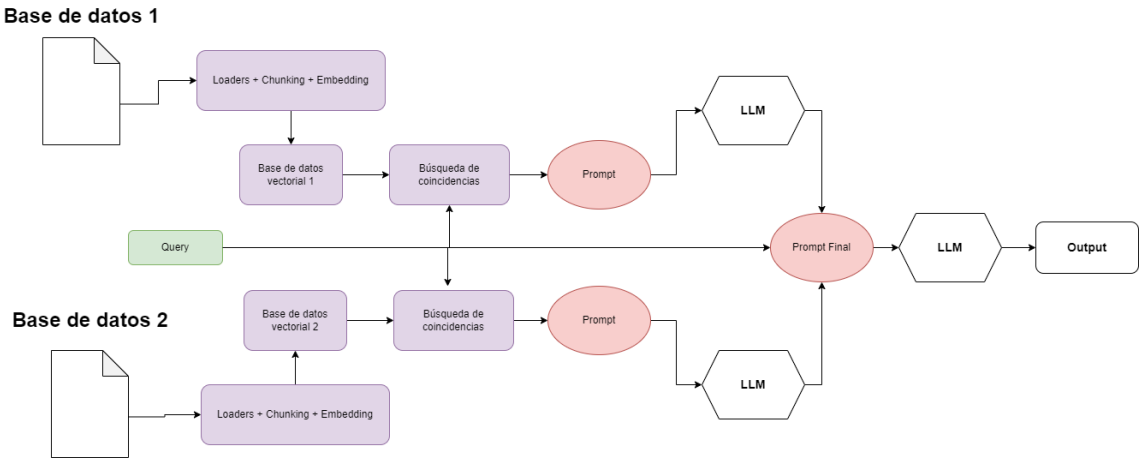


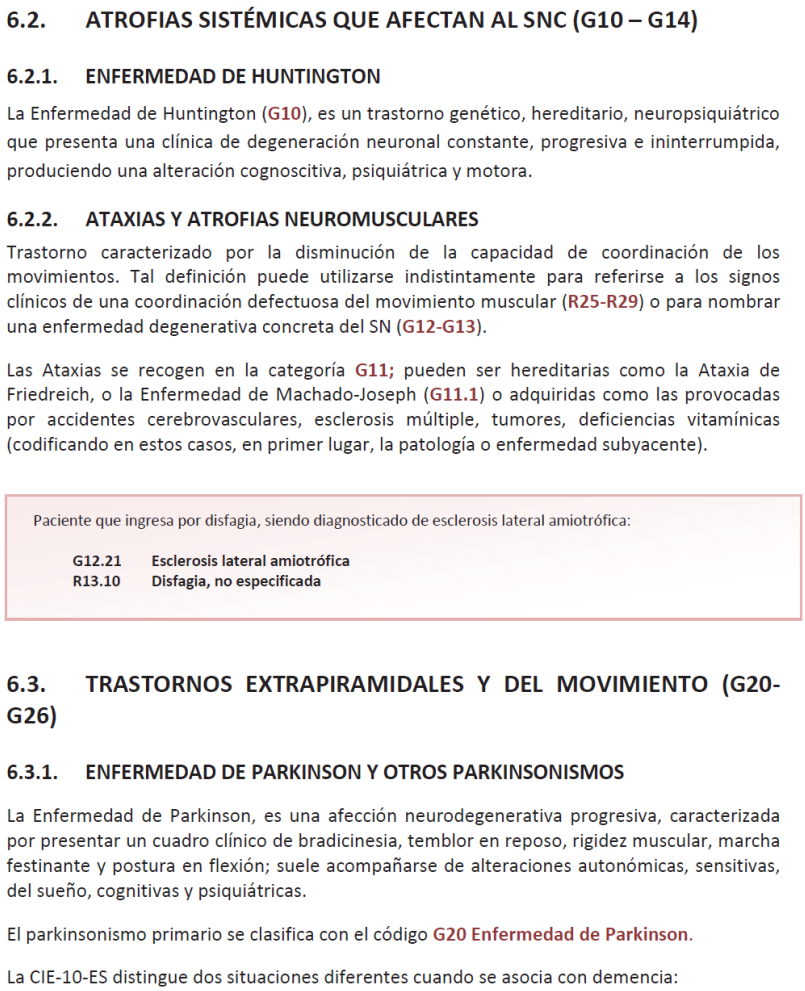
Figura 4.6: Diagrama del sistema de la fase 2

4.2.3. Tercera fase del proyecto

Análisis de la base de datos

En esta tercera y última fase del proyecto se utilizó la misma base de datos que en la anterior, conformada por dos manuales explicativos. El motivo de esto fue que los resultados obtenidos con dichas fuentes de información resultaron bastante satisfactorios en cuanto a la cantidad de contexto que aportaban sobre diagnósticos [22], [23].

Por una parte, la base de datos presentada en la figura 4.3 aporta datos estadísticos sobre la cantidad de personas que padecen una enfermedad dependiendo de la situación en l que se encuentren (edad, sexo, etc.). Sin embargo, la presencia de otra base de datos centrada en la explicación de los síntomas que los pacientes sufren con cada tipo de enfermedad, resulta crucial para conformar un contexto robusto en el que se puedan encontrar coincidencias para cualquier consulta que el usuario pueda tener.



6.2. ATROFIAS SISTÉMICAS QUE AFECTAN AL SNC (G10 – G14)

6.2.1. ENFERMEDAD DE HUNTINGTON

La Enfermedad de Huntington (G10), es un trastorno genético, hereditario, neuropsiquiátrico que presenta una clínica de degeneración neuronal constante, progresiva e ininterrumpida, produciendo una alteración cognoscitiva, psiquiátrica y motora.

6.2.2. ATAXIAS Y ATROFIAS NEUROMUSCULARES

Trastorno caracterizado por la disminución de la capacidad de coordinación de los movimientos. Tal definición puede utilizarse indistintamente para referirse a los signos clínicos de una coordinación defectuosa del movimiento muscular (R25-R29) o para nombrar una enfermedad degenerativa concreta del SN (G12-G13).

Las Ataxias se recogen en la categoría G11; pueden ser hereditarias como la Ataxia de Friedreich, o la Enfermedad de Machado-Joseph (G11.1) o adquiridas como las provocadas por accidentes cerebrovasculares, esclerosis múltiple, tumores, deficiencias vitamínicas (codificando en estos casos, en primer lugar, la patología o enfermedad subyacente).

Paciente que ingresa por disfagia, siendo diagnosticado de esclerosis lateral amiotrófica:

G12.21 Esclerosis lateral amiotrófica
R13.10 Disfagia, no especificada

6.3. TRASTORNOS EXTRAPIRAMIDALES Y DEL MOVIMIENTO (G20-G26)

6.3.1. ENFERMEDAD DE PARKINSON Y OTROS PARKINSONISMOS

La Enfermedad de Parkinson, es una afección neurodegenerativa progresiva, caracterizada por presentar un cuadro clínico de bradicinesia, temblor en reposo, rigidez muscular, marcha festinante y postura en flexión; suele acompañarse de alteraciones autonómicas, sensitivas, del sueño, cognitivas y psiquiátricas.

El parkinsonismo primario se clasifica con el código G20 Enfermedad de Parkinson.

La CIE-10-ES distingue dos situaciones diferentes cuando se asocia con demencia:

Figura 4.7: Manual empleado en fases 2 y 3

En la figura 4.7 se muestra un ejemplo de la estructura del manual empleado para las bases de datos utilizadas en la segunda y tercera fase. A lo largo del documentos se realiza una explicación de numerosas enfermedades distribuidas principalmente por la zona del cuerpo a la que afectan. No obstante, también hay ciertos puntos clasificados por el tipo de enfermedad o trastorno, o bien por las condiciones especiales que afectan a la salud.

En la imagen se habla de atrofias que afectan al Sistema nervioso Central (SNC) además de trastornos que afectan al movimiento de las personas, mencionando en cada apartado el rango de códigos CIE-10 en el que se encuentran este tipo de afecciones. Dentro de cada apartado se realiza una explicación del tipo de síntomas que suelen tener las personas que las padecen. Todo esto dentro del capítulo del manual en el que se encuentran las enfermedades del sistema nervioso.

Carga de la base de datos

La carga de esta fuente de información sigue el mismo funcionamiento que la que se realizó en la segunda fase: con una indexación progresiva y dividiendo las bases de datos en lotes para subirlos evitando sobrecargas.

La diferencia que hay respecto a la anterior fase es que se realizó un tipo de carga más completa que en etapas del proyecto anteriores: se implementó una función de *langchain* que nos permitía editar el tamaño de los *chunks* en los que se dividía el texto. Se analizó el número de caracteres por página que tenían los documentos y se dió un tamaño de *chunks* acorde a los caracteres que había en una página de manera que cada *chunk* se correspondiera con una página del documento.

Por otra parte se agregó un tamaño de solapamiento de texto u *overlap*, con esto se permite especificar la cantidad de caracteres que se solapan entre un *chunk* y otro. Algo que posibilita que en cada chunk, además de la propia información que se almacenaría como contexto con el número de caracteres establecido, se guarde también una cantidad de datos (en este caso entre un 10 % y un 20 %) tanto del *chunk* anterior como del posterior con la intención de preservar un mejor contexto entre *chunks* que sean adyacentes. [25]

Esto nos permite que, a la hora de recuperar los datos indexados, los LLM cuenten con más información para poder proporcionar al usuario una respuesta más completa y adecuada.

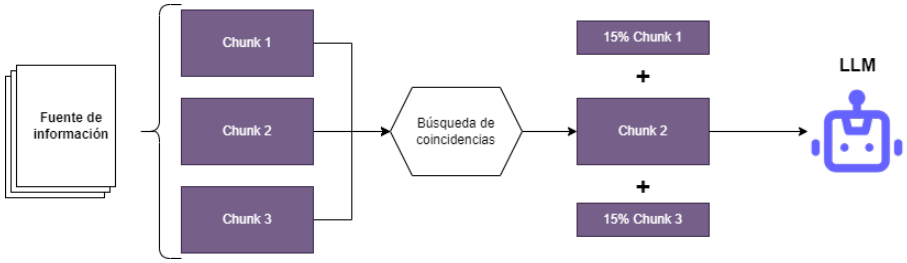


Figura 4.8: Diagrama del funcionamiento del overlap

Diseño de un sistema RAG MultiQuery Retriever

Para esta última fase del proyecto, se desarrolló un sistema *Retrieval-Augmented Generation MultiQuery* con la intención de evaluar su rendimiento y realizar una comparación de los resultados obtenidos frente a los sistemas que fueron diseñados en las dos fases anteriores.

En primer lugar, las bases de datos fueron ambas indexadas en el mismo índice, de manera que toda la información se encontrara en el mismo dataset. El cual sería cargado de la misma manera que en la segunda fase, como se ha comentado anteriormente.

A continuación, el sistema recibiría como entrada una consulta (*query*), la cual sería procesada y añadida a un *prompt*, que sería dado como entrada a un *large language model*. El objetivo de procesar la *query* de primeras es crear un grupo de solicitudes a partir de la inicial, instrucción que fue especificada en el *prompt* mencionado, para evaluar el comportamiento que tenía la búsqueda de similaridades con cada una de ellas. Este proceso fue realizado con el objetivo de realizar una comparación entre el porcentaje de coincidencias que se encontraban con cada una de las consultas.

A partir de la primera solicitud se solicitó al LLM que se creara una salida en formato JSON que constará de tres *querys*: la primera fue la consulta hecha por el usuario, que sería conservada sin ningún cambio. La segunda fue una petición en la que se daría una descripción del mismo problema, pero con una estructura y palabras distintas. Por último, puesto que los sistemas de búsqueda de coincidencias se suelen centrar en las palabras clave que tiene la oración, se creó una tercera consulta. En esta solo constaron las palabras clave de la consulta original, con un formato mucho más resumido. Esto se hizo para evitar que con ciertas consultas en las que el usuario diera datos que no resultaran concluyentes, la información extraída fuera errónea o la relación entre la solicitud y el contenido del *chunk*extraído fuera imprecisa.

Después de realizar varias pruebas con el sistema comentado, se decidió eliminar la segunda consulta mencionada, dejando únicamente la solicitud original del usuario y el resumen de esta como segunda *query*. Esta decisión fue tomada de esta manera debido a que, analizando los datos, se apreció que aún cambiando la estructura de la solicitud, el modelo extraía como resultado el mismo vector con prácticamente el mismo porcentaje de similaridad.

Una vez se han obtenido los resultados de búsqueda con ambas solicitudes, estos serán agrupados en un único *chunk* que será procesado y analizado por un modelo de lenguaje grande (LLM) que dará una respuesta final a la consulta del usuario.

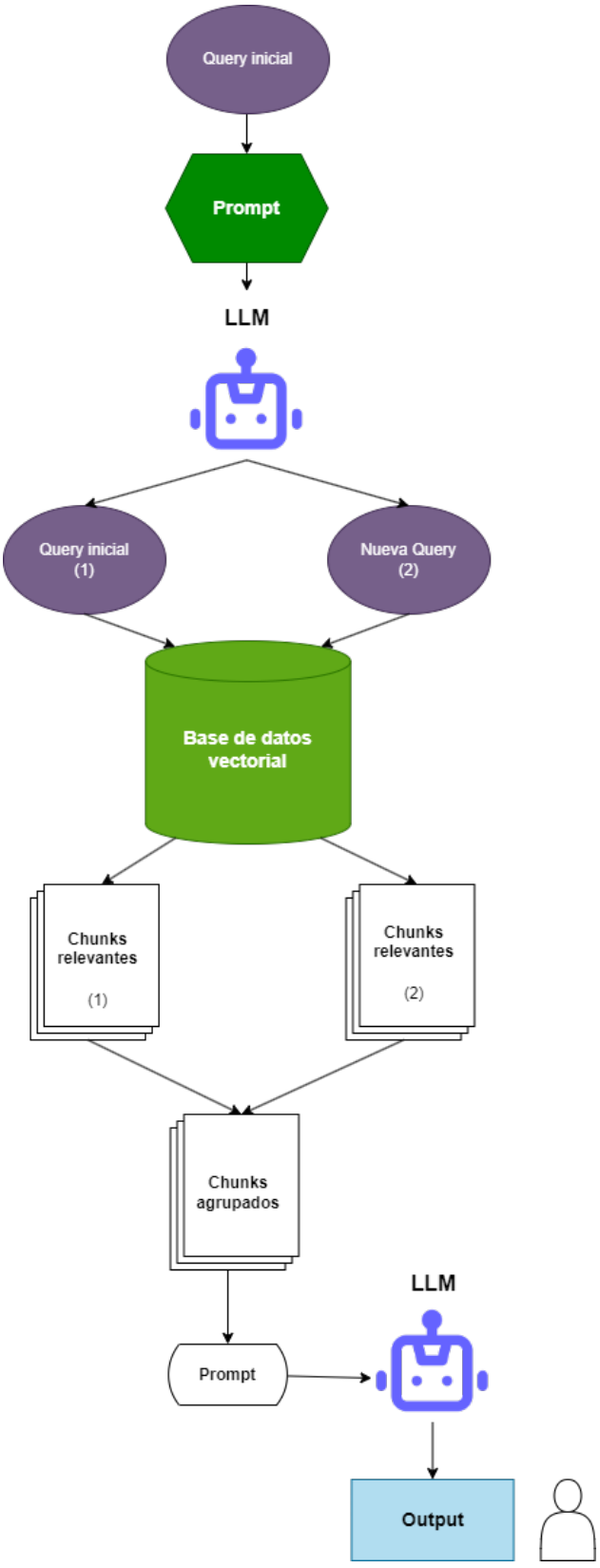


Figura 4.9: Diagrama sistema fase 3

Capítulo 5

Implementación y Resultados obtenidos

5.1. Creación de una interfaz de usuario

Como última parte del proyecto se decidió crear una interfaz para simular una situación real de lo que un usuario podría encontrar si utilizara el asistente médico que se ha implementado.

Para ello se optó por utilizar Flask, un framework en Python creado para facilitar el desarrollo de aplicaciones web con un forma de trabajar Modelo-Vista-Controlador (MVC). Esta manera de trabajo consiste en la separación de la aplicación en: modelo de datos, que es la parte en la que se encuentran los datos y el flujo que sigue el programa. Vista, que es la sección en la que se crea la interfaz que verá el usuario y en la que introducirá sus consulta. Por último, la fase del controlador, que se gestionaran las peticiones que el usuario realiza al asistente y se realiza una llamada al modelo para elaborar una respuesta adecuada a la petición.

Para definir el aspecto que tendría la interfaz de decidieron dos códigos en los siguientes formatos:

Código en formato Cascading Style Sheets (CSS):

Este código fue creado para decidir el aspecto visual que tendría la interfaz. En este, se determinan elementos como la fuente del texto que se presentará, las formas, los colores y la disposición que tendrían los distintos contenedores de entrada y salida, donde se verían tanto la consulta del usuario como la respuesta del asistente. Así como, otros aspectos como la posición del encabezado o el color de la letra.

Código en formato HyperText Markup Language (HTML):

Una vez se define el formato de los elementos que conforman la interfaz con el código CSS, se creó un algoritmo HTML que defina la estructura que tendrá la aplicación. Se especificó el nombre del asistente: *Júlia*, se indica que se trata de un asistente médico, además de definir los campos de entrada de texto como de salida. Con la presencia, entre estos, de un botón que el usuario debe pulsar para que su consulta sea enviada una vez haya terminado con la explicación de los síntomas que tiene y así pueda recibir una respuesta.

Figura 5.1: Interfaz de usuario final

En la figura 5.1 se puede ver el resultado final obtenido con el diseño de la interfaz que cuenta, como se ha mencionado previamente, con dos contenedores en los que el usuario introducirá su consulta y en el que se representará la respuesta a esta. Así como con un botón para enviar la solicitud una vez se haya redactado todo, además del encabezado de la interfaz.

5.2. Métricas empleadas en la evaluación de los sistemas

El hecho evaluar el sistema basado en *Retrieval-Augmented Generation* en primera instancia resultó complejo y costoso debido a que, al involucrar dos fases (recuperación de información y generación de una respuesta), era necesario emplear métricas que evaluaran tanto el recuperador como el generador. Los sistemas RAG suelen encontrar problemas a la hora de su evaluación ya que, en ciertas situaciones como en la que nos encontramos, es difícil poder evaluar ciertos campos de la respuesta de forma objetiva.

Con el objetivo de poder analizar el rendimiento del sistema se utilizó *Ragas*, una librería implementada en Python que fue de gran ayuda como herramienta de evaluación del sistema RAG.

Algunas de las medidas que emplea *Ragas* están basadas en la información que es considerada como verdadera y comprobada, también conocida como *ground truth*. Sin embargo, para realizar una evaluación adecuada, es necesario contar con otros aspectos como una cantidad de preguntas que se le han realizado al sistema, la correspondiente respuesta a estas preguntas, así como el contexto que se ha recuperado de la base de datos a raíz de la pregunta realizada por el usuario.

Partiendo de este punto, las métricas que evalúan los sitemas basados en RAG están divididas en dos grupos:

En primer lugar encontramos las medidas que se encargan de analizar la generación de respuestas del sistema, estas son:

Fidelidad: una métrica que nos resultará muy útil para comprobar si la información que se proporciona en la respuesta se ajusta a la información que se le ha proporcionado en el contexto, realizando una comparación entre los datos proporcionados en la salida y los recuperados desde los documentos.

Relevancia de la respuesta: esta medida consiste en la evaluación lo relevante y adecuada que es la respuesta en relación con la consulta que le ha planteado el usuario, analizando si información proporcionada resulta de utilidad respecto al tema planteado por quien realiza la solicitud.

Por otra parte, encontramos las métricas que se centran en evaluar la recuperación de información del sistema, que son:

Precisión del contexto: se centra en estudiar si los elementos que se han proporcionado en el contexto que ha empleado el modelo para responder son verdaderamente precisos en relación con la referencia verdadera de cara a la generación de una respuesta.

Recuperación del contexto: analiza la capacidad que tiene el sistema para recuperar toda la información de la base de datos que esté relacionada con el tema que ha propuesto el usuario.

Relevancia del contexto: con esta métrica se investiga si la información que se ha proporcionado desde la fuente de conocimiento es realmente relevante en el resultado final que se ha proporcionado.

Además de las medidas implementadas desde la librería *Ragas*, se hizo un estudio de la calidad y los tiempos de respuesta de los sistemas diseñados en la segunda y tercera fase del proyecto. Todo ello con la intención de decidir cual de lo sistemas resultaba más adecuado a la hora de incorporarlo al modelo definitivo de la aplicación.

5.3. Métricas utilizadas en el análisis de los modelos empleados

De forma adicional al análisis con las métricas mencionadas, se realizó una evaluación de los distintos modelos de GPT que se emplearon en las fases de generación de una respuesta con el objetivo de decidir cual de los modelos existentes actuales resulta más adecuado para el sistema RAG.

5.4. Evaluación de los resultados obtenidos

La evaluación del sistema RAG consta de dos partes principales: en primer lugar, se realiza un análisis objetivo de los resultados que se han obtenido con el uso de las métricas mencionadas. Como segunda parte se ha ideado un análisis subjetivo, por parte de un médico experto, de las respuestas que el sistema RAG, de forma que pueda validar la calidad y precisión que estas podrían proporcionarle a una persona que realice una consulta en un caso real. Todo esto con el objetivo de que se pueda disponer finalmente de un análisis completo de la aplicación.

5.4.1. Análisis con métricas objetivas

Para esta primera evaluación se diseñaron quince preguntas sobre distintos campos de la medicina, simulando lo que podría ser una consulta de una persona en un caso real. A continuación, se procesaron estas preguntas, obteniendo a partir de estas, los fragmentos de las bases de datos que más se ajustaban a la solicitud. A este contexto recuperado por el sistema, agregaríamos las respuestas proporcionadas por el *large language model*.

Todo esto fue evaluado por las métricas mencionadas en el punto 5.2, obteniendo los siguientes resultados:

Pregunta	Fidelidad	Relevancia Respuesta	Recuperación Contexto	Precisión Contexto	Relevancia Contexto
Pregunta 1	66.7 %	82.8 %	50 %	61 %	100 %
Pregunta 2	50.0 %	83.0 %	50 %	28 %	20 %
Pregunta 3	88.9 %	85.5 %	30 %	100 %	33.3 %
Pregunta 4	50.0 %	82.1 %	50 %	30 %	35 %
Pregunta 5	60.0 %	84.3 %	100 %	100 %	25.4 %
Media de resultados	63.12 %	83.54 %	56.0 %	63.8 %	36.7 %

Tabla 5.1: Resultados obtenidos con métricas objetivas

En la tabla 5.1 se presentan los resultados obtenidos después de realizar un total de cinco preguntas como etapa de prueba. Como se puede ver en los resultados, la precisión en la mayoría de campos es aparentemente poco satisfactoria.

En métricas como la fidelidad o los campos relacionados con el contexto, se encuentran ciertas preguntas en las que el resultado es bastante satisfactorio, sin embargo no lo es para todas ellas. Después de realizar varias consultas, evaluando los resultados de estas, se ha llegado a la conclusión de que estas métricas se centran de una forma muy estricta en que la respuesta que se haya proporcionado cuente exactamente con todas las afecciones que se enuncian en el *ground truth*. Algo que resulta perjudicial a la hora de evaluar modelos relacionados con el procesamiento del lenguaje, ya que en este, múltiples palabras pueden tener el mismo significado en función de su organización.

Se penaliza de forma muy negativa un resultado cuando algo que se ha mencionado en la verdad absoluta no aparece mencionado de la misma manera en la respuesta, aunque el aspecto que se ha mencionado en la salida del sistema haya sido muy similar pero mencionado con palabras distintas. Esto es algo que se debe tener en cuenta a la hora de valorar los resultados, ya que el método de evaluación que implementa *Ragas* puede llegar a ser demasiado objetivo para esta clase de respuestas.

Medidas como la precisión del contexto pueden verse afectadas por este problema, ya que al tratarse de fragmentos de información que almacenan páginas enteras de un documento, hay más información que la que resulta relevante para la respuesta. Cuando se realiza la comparación, la precisión del contexto disminuye, aunque cuente con esa información relevante que se buscaba.

Es por ello que los resultados obtenidos, a pesar de estar lejos de los que suelen ser habituales en la evaluación de modelos en el ámbito de la inteligencia artificial, no han sido considerados como negativos, pues teniendo en cuenta las limitaciones mencionadas, no son tan desfavorables como podría parecer en una primera observación.

Sin embargo, estos resultados no consiguen ser del todo satisfactorios, pues no se ha considerado que sean del todo concluyentes. Es por ello que se ha ideado una forma alternativa de evaluar los resultados proporcionados por el modelo.

De forma usual, las respuestas proporcionadas por los modelos basados en *Retrieval-Augmented Generation* son evaluadas por personas con un buen conocimiento sobre el tema a evaluar, capaces de determinar la precisión que tiene la respuesta de una forma fiable. Con esto se consigue saber con certeza, no solo que la información que se proporciona sea adecuada a la pregunta recibida, si no que también sea correcta desde el punto de vista de una persona especializada.

5.4.2. Aspectos analizados en los modelos empleados

	GPT-3.5 turbo	GPT-4	GPT-4 turbo	GPT-4o
Tiempo Respuesta	25-35 s	20-30 s	12-20 s	7-15 s
Coste/1M tokens	2,79€	27,94€	9,31€	4,66€

Tabla 5.2: Resultados del análisis de los modelos

En los modelos que se han utilizado a lo largo del diseño del asistente se han valorado un total de tres aspectos, se han analizado los dos puntos expuestos en la tabla 5.2 además de la calidad en obtenida en las respuestas por cada uno de los modelos, en cada uno de los aspectos mencionados los modelos que han supuesto un mayor beneficio son:

Tiempo de respuesta: en cuanto al tiempo de respuesta, se habilitó un campo que proporcionará el tiempo que cada uno de los modelos empleaba en generar las correspondientes respuestas. En este caso, el modelo que mejor rendimiento obtuvo fue el GPT-4o

Coste de los modelos: para este punto, se realizó una búsqueda del coste que tenía cada uno de los modelos por cada millón de tokens que se utilizaban. En este caso, la palabra token se refiere al número de palabras o partes de palabras que han sido procesadas por el modelo.

Por lo tanto, el coste del uso de los modelos depende de la cantidad de texto que estos deban procesar. Después de realizar la búsqueda mencionada se obtuvo que el modelo que menos coste por tokens procesados tiene es el modelo GPT-3.5 turbo.

Calidad en la respuesta: En lo que respecta a la calida de la respuesta, se obtuvo que los modelos que proporcionaban una mensaje más completo y claro al usuario fueron los modelos GPT-4o, GPT-4 y GPT-4 turbo. Sin una gran diferencia en el texto que proporcionaban a la salida.

Una vez sopesados los distintos aspectos mencionados, se optó por el empleo del modelo GPT-4o, pues este era el modelo que mejor balance aportaba en cuanto a tiempo, coste y calidad de respuesta.

5.4.3. Elección del sistema a emplear

Después de analizar los distintos sistemas con las métricas mencionadas, se concluyó con la elección del sistema diseñado en la tercera fase. Para tomar esta decisión se tuvieron en cuenta aspectos como la calidad, el tiempo que tardaba el sistema en proporcionar la respuesta y la cantidad de recursos que demandaba cada uno.

En términos de calidad tanto el segundo como el tercer sistema proporcionaron resultados bastante satisfactorios. Sin embargo, el sistema de la tercera fase proporcionaba un rendimiento considerablemente mejor en cuanto a los recursos que este utilizaba, ya que se empleaban los *large language models* un número menor de veces y, por consiguiente, se demoraba menos en proporcionar una respuesta.

Esto añadido a que las llamadas a LLMs del tercer modelo procesaban una cantidad de datos menor, por lo que suponían un coste económico más bajo, hizo de este una mejor opción para la implementación de la última versión del asistente.

5.4.4. Análisis subjetivo

Debido a los resultados obtenidos con las métricas obtenidas en la tabla 5.1 y ante la poca variedad de métricas para la evaluación de sistemas basados en *Retrieval-Augmented Generation*, se diseñó una evaluación subjetiva por parte de un profesional de la medicina con el objetivo de contar con un análisis del sistema que, añadido al ya realizado, pudiera aportarnos una visión más completa de los resultados que se obtienen a partir del sistema.

Este análisis ha consistido en, junto con la colaboración del profesional, el diseño de una serie de preguntas centradas en distintos campos de la medicina con las se ha tratado de abarcar todos los campos que la componen. Además, las preguntas han sido estructuradas también por su dificultad, planteando preguntas tanto generales como concretas sobre ciertos aspectos de este ámbito hasta conformar un total de quince preguntas para realizar la prueba.

Una vez que las preguntas fueron creadas, fueron introducidas en el modelo para que fueran procesadas y este pudiera proporcionar una respuesta acorde a la consulta recibida. Estas respuestas fueron evaluadas por el profesional mediante una rúbrica (disponible en los anexos de este documento) diseñada específicamente para analizar de una forma completa el resultado, esta contaba con un total de diez aspectos importantes a considerar en los que la evaluación se situaba entre valores de 1 a 5, siendo este último la mejor calificación con la que se puede puntuar cada campo de la rúbrica.

Después de que todas las respuestas fueran evaluadas, se recogieron las rúbricas de cada una de las preguntas, obteniendo los siguientes resultados:

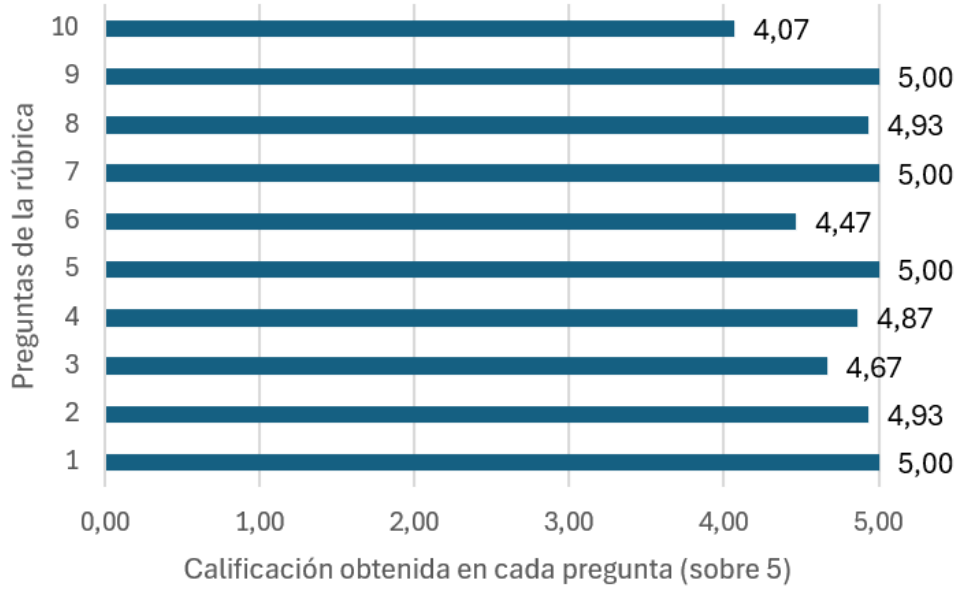


Figura 5.2: Resultados medios tras la evaluación

En la imagen se presentan los resultados obtenidos, en los que se exponen las medias de calificación en cada punto de la evaluación después de que todas las preguntas hayan sido evaluadas por el profesional médico con la rúbrica mencionada. En los gráficos se puede apreciar como los resultados obtenidos son bastante satisfactorios, pues teniendo en cuenta de que la calificación máxima se puede obtener en cada pregunta es de 5, la gran mayoría de los resultados se encuentran muy cercanos a esta puntuación.

Uno de los puntos en los que más podría mejorar la respuesta del modelo es en la mención por parte del asistente de posibles riesgos asociados en las recomendaciones que este ha proporcionado. Sin embargo los resultados no han sido considerados negativos, ya que la calificación media con la que se evalúa este apartado es superior a 4.

Por otro lado, según los resultados obtenidos en lo gráficos, el programa presenta un rendimiento excelente en los siguientes aspectos: proporciona información basada en evidencias médicas actualizadas, utiliza un tono adecuado y respetuoso hacía el paciente, respetando la situación de este. Por otra parte, el comportamiento por parte del asistente en lo que respecta a la recomendación de solicitar atención médica cuando esta sea necesaria ha sido muy acertado. Por último, se ha comprobado que el modelo ha identificado correctamente los síntomas graves comentados los pacientes que necesitan atención médica inmediata.

Con estos resultados obtenidos, después de contar con un análisis por parte de una persona especializada en el campo, se posee una evaluación mucho más completa sobre el rendimiento del asistente, contando con una variedad notable de métricas al final de este proceso.

Capítulo 6

Conclusiones y líneas futuras

6.1. Conclusiones

Primeramente, se han expuesto todos los sistemas que se han ido creando durante el desarrollo del asistente virtual. Comentando los aspectos característicos de cada uno, las evoluciones que han ido realizándose en cada fase con respecto a la anterior, así como el objetivo de cada una de las implementaciones que se han diseñado en las distintas fases.

En segundo lugar, se han presentado las herramientas que se han empleado en el diseño de la aplicación, como son librerías, funciones, así como los módulos creados que nos han ayudado en el proceso de desarrollo del prototipo final. Del mismo modo se han construido y analizado las bases de datos con las que se ha contado para crear un amplio contexto en el que los *large language models* (LLM) pudieran apoyarse para encontrar una información referente a la consulta. Además del método que se ha utilizado para el procesamiento de estas.

Se ha detallado los distintos puntos en los que se ha dividido el sistema basado en RAG, en cada una de las distintas fases, para llevar a cabo la generación de una respuesta adecuada a las necesidades de las solicitudes del usuario.

Se ha conseguido implementar una interfaz de usuario de manera que la interacción entre la persona que quiera realizar una consulta y el sistema diseñado sea mucho más cómoda.

Se ha llevado a cabo una evaluación, tanto objetiva, con las métricas que se han comentado anteriormente, como subjetiva, con las distintas pruebas a las que un médico experto ha sometido al asistente.

Por otra parte, se ha seleccionado el sistema que ha proporcionado un mejor rendimiento teniendo en cuenta el computo global de los resultados obtenidos tras la evaluación de cada uno de ellos.

Para finalizar, se ha conseguido llevar a cabo la elaboración de un asistente médico virtual generado a partir de un sistema basado en *Retrieval-Augmented Generation* que, como se ha comprobado durante la evaluación del modelo y viendo los resultados que se han obtenido, es capaz de responder a cualquier tipo de consulta médica que los usuarios puedan plantearle.

Contando con los puntos comentados, se puede concluir que se ha conseguido cumplir los propósitos planteados inicialmente (apartado 1.2), tanto el objetivo general como los objetivos más específicos.

6.2. Líneas futuras

En lo que respecta a las líneas futuras del proyecto se podrían plantear los siguientes escenarios:

Una posible línea futura de la aplicación creada estaría relacionada con la reciente implementación de la Clasificación Internacional de Enfermedades, 11ª Edición (CIE-11). Esta entró en vigor en el año 2022. Sin embargo este sistema todavía no está implementando en la totalidad de los países del mundo.En paises como España todavía se sigue empleando la CIE-10, por lo que la base de datos con la que cuenta el sistema todavía sería óptima para que el sistema pueda generar respuestas actualizadas a partir de ella. No obstante, en un futuro la CIE-11 será implementada en España, por lo que sería adecuado renovar las bases de datos en ese momento.

El modelo basado en RAG que se ha diseñado sigue la estructura Question-Answer en la que el usuario realiza una pregunta y recibe una respuesta por parte del modelo. Sin embargo, una posible estrategia con la que se podrían realizar nuevas pruebas sería un estructura Chatbot, en la persona pueda mantener una conversación con el modelo. En esta, el sistema podría elaborar preguntas para el usuario y de esta forma, poder obtener más información sobre el estado de este y realizar un diagnóstico más preciso. Esta estructura sería capaz de mantener toda la conversación entre la persona y el modelo, de manera que esta fuera utilizada como contexto adicional para enunciar una respuesta final más adecuada.

Por otra parte, ante la falta de una variedad de métricas objetivas que nos puedan proporcionar una evaluación adecuada sobre el sistema creado, se podría tratar de idear nuevas estrategias para conseguir una evaluación que nos permita obtener resultados más concluyentes que los que se han obtenido con las métricas utilizadas.

Bibliografía

[1] Nuria Oliver. *Inteligencia artificial, naturalmente: un manual de convivencia entre humanos y máquinas para que la tecnología nos beneficie a todos*. Ministerio de Asuntos Económicos y Transformación Digital, 2020.

[2] Tao Tu, Shekoofeh Azizi et al. "Towards Generalist Biomedical AI". En: *arXiv* arXiv:2307.14334 (jul. de 2023). URL: https://arxiv.org/pdf/2307.14334.

[3] Jefferson Aguirre et al. "Aplicación de la Inteligencia Artificial en la Industria Automotriz". En: *Revista Ibérica de Sistemas e Tecnologias de Informação* E42 (feb. de 2021). URL: https://www.proquest.com/docview/2493869275?pq-origsite=gscholar&fromopenview=true&sourcetype=Scholarly%20Journals.

[4] J. Portal Díaz et al. "Aplicación de técnicas de inteligencia artificial para reconocimiento facial en sistemas de seguridad en ambientes de intranet". En: *Mare Ingenii* (oct. de 2022). URL: https://doi.org/10.52948/mare.v4i1.682.

[5] A.C. Müller y S. Guido. *Introduction to Machine Learning with Python: A Guide for Data Scientists*. O'Reilly Media, Incorporated, 2018. ISBN: 9789352134571. URL: https://books.google.es/books?id=jGdXswEACAAJ.

[6] I. Goodfellow, Y. Bengio y A. Courville. *Deep Learning*. Adaptive Computation and Machine Learning series. MIT Press, 2016. ISBN: 9780262035613. URL: https://books.google.es/books?id=Np9SDQAAQBAJ.

[7] M.A. Nielsen. *Neural Networks and Deep Learning*. Determination Press, 2015. URL: https://books.google.es/books?id=STDBswEACAAJ.

[8] Muhammad Imran Razzak, Saeeda Naz y Ahmad Zaib. "Deep learning for medical image processing: Overview, challenges and the future". En: *Classification in BioApps: Automation of decision making* (2018), págs. 323-350.

[9] Elizabeth D Liddy. "Natural language processing". En: (2001). URL: https://surface.syr.edu/cgi/viewcontent.cgi?article=1043&context=istpub.

[10] W. John Hutchins. "The Georgetown-IBM Experiment Demonstrated in January 1954". En: *Machine Translation: From Real Users to Research*. Ed. por Robert E. Frederking y Kathryn B. Taylor. Springer Berlin Heidelberg, 2004.

[11] Terry Winograd. "What does it mean to understand language?" En: *Cognitive science* 4.3 (1980), págs. 209-241.

[12] Rajibul Hasan, Riad Shams y Mizan Rahman. "Consumer trust and perceived risk for voice-controlled artificial intelligence: The case of Siri". En: *Journal of Business Research* (2021). ISSN: 0148-2963. URL: https://www.sciencedirect.com/science/article/pii/S0148296320308419.

[13] Sanghyun Choo y Wonjoon Kim. "A study on the evaluation of tokenizer performance in natural language processing". En: *Applied Artificial Intelligence* (2023). URL: https://doi.org/10.1080/08839514.2023.2175112.

[14] S. Sarica y J. Luo. "Stopwords in technical language processing". En: *PLoS ONE* (2021). URL: https://doi.org/10.1371/journal.pone.0254937.

[15] Divya Khyani et al. "An interpretation of lemmatization and stemming in natural language processing". En: *Journal of University of Shanghai for Science and Technology* 22.10 (2021), págs. 350-357.

[16] Abhinav Kimothi. *Retrieval Augmented Generation - A Simple Introduction*. 2023. URL: https://abhinavkimothi.gumroad.com/l/RAG.

[17] Krishna Juluru et al. "Bag-of-Words Technique in Natural Language Processing: A Primer for Radiologists". En: *RadioGraphics* 41.5 (2021). PMID: 34388050, págs. 1420-1426. DOI: 10.1148/rg.2021210025. URL: https://doi.org/10.1148/rg.2021210025.

[18] Prafulla Bafna, Dhanya Pramod y Anagha Vaidya. "Document clustering: TF-IDF approach". En: *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*. 2016, págs. 61-66. DOI: 10.1109/ICEEOT.2016.7754750.

[19] William Cavnar y John Trenkle. "N-Gram-Based Text Categorization". En: *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval* (mayo de 2001).

[20] W. Pedryez y A.V. Vasilakos. "Linguistic models and linguistic modeling". En: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 29.6 (1999), págs. 745-757. DOI: 10.1109/3477.809029.

[21] Leon Derczynski. "Complementarity, F-score, and NLP Evaluation". En: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. Ed. por Nicoletta Calzolari et al. Portorož, Slovenia: European Language Resources Association (ELRA), mayo de 2016, págs. 261-266. URL: https://aclanthology.org/L16-1040.

[22] MINISTERIO DE SANIDAD, SERVICIOS SOCIALES E IGUALDAD. *Manual de Codificación CIE-10-ES Diagnósticos*. 2016. URL: https://www.sanidad.gob.es/estadEstudios/estadisticas/normalizacion/CIE10/CIE10ES_2016_norm_Manual_codificacion_Diagnosticos.pdf.

[23] Instituto Nacional de la Seguridad Social. *Manual de Tiempos Óptimos de Incapacidad Iemporal 4ª Edición*. 4/01/2018. URL: https://www.seg-social.es/wps/wcm/connect/wss/d24cc76a-e1f4-49b6-b36f-fb8fc00a32a7/Manual+Tiempos+%C3%93ptimos+IT_Castellano_v4.0_+Accesibilidad.pdf?MOD=AJPERES.

[24] Sociedad Española de Urgencias Pediátricas. *tabla de diagnosticos seup - version cie-10 marzo 2017*. 2017. URL: https://seup.org/pdf_public/gt/codificacion_tabla.pdf.

[25] Anoosheh Heidarzadeh y Amir H. Banihashemi. "Overlapped Chunked network coding". En: *2010 IEEE Information Theory Workshop on Information Theory (ITW 2010, Cairo)*. 2010, págs. 1-5. DOI: 10.1109/ITWKSPS.2010.5503153.

Parte II

Anexos

0.1. Rúbrica empleada en la evaluación del modelo

Pregunta	No cumple (1)	Cumple parcialmente (2)	Cumple adecuadamente (3)	Cumple muy bien (4)	Cumple excelentemente (5)
¿La información proporcionada se basa en evidencias médicas y prácticas clínicas actuales?					
¿La respuesta tiene relevancia respecto a los síntomas e inquietudes que ha descrito el paciente?					
¿La respuesta es fácil de entender? ¿Se evita lenguaje médico innecesario?					
¿La respuesta es concisa, tratando de evitar información redundante?					
¿En la respuesta se utiliza un tono apropiado y respetuoso? ¿Se muestra empatía por la situación que vive el usuario?					
¿En la respuesta se ofrece apoyo emocional y técnico? ¿Se trata de tranquilizar al paciente si es necesario?					
¿Se recomienda la búsqueda de atención médica profesional cuando se considera necesario?					
¿Se dan instrucciones claras para el seguimiento de la condición o si se debe volver a contactar al asistente con nueva información?					
¿En la respuesta se identifican de forma correcta los síntomas graves que requieren atención médica inmediata?					
¿Se mencionan posibles riesgos y precauciones asociadas a las recomendaciones que se le han dado al paciente?					

Tabla 1: Rúbrica empleada en la evaluación del modelo