Integrating Retrieval-Augmented Generation Architecture for Clinical Guidance: Building a Chatbot to Assist Rheumatology Physicians

Daniel Camacho Montaño

Machine Learning

Màster en Bioestadística y bioinformática

Nombre del director/a de TF:
Alfredo Madrid García

Nombre del/de la PRA:
Agnès Pérez Millan

1 de abril de 2025

Universitat Oberta de Catalunya

Ficha Del Trabajo Final

Título del trabajo:	Integrating Retrieval-Augmented Generation Architecture for Clinical Guidance: Building a Chatbot to Assist Rheumatology Physicians
Nombre del autor/a:	Daniel Camacho Montaño
Nombre del director/a de TF:	Alfredo Madrid García
Nombre del/de la PRA:	Agnès Pérez Millan
Fecha de entrega:	1 de abril de 2025
Titulación o programa:	Màster en Bioestadística y bioinformática
Área del trabajo final:	Machine Learning
Idioma del trabajo:	Castellano
Palabras clave:	Inteligencia artificial, chatbot, guías clínicas, reumatología, sistema de ayuda a la decisión

Resumen del trabajo

Màxim 250 paraules, amb la finalitat, context d'aplicació, metodologia, resultats i conclusions del treball.

Abstract

A maximum of 250 words, detailing the purpose, context of application, methodology, results and conclusions of the work.

Índice general

Índice de figuras

Índice de cuadros

Siglas

AASLD American Association for the Study of Liver Diseases

ACR American College of Rheumatology

API Application Programming Interface

BERT Bidirectional Encoder Representations from Transformers

CCEG Competencia en Compromiso Ético y Global

EASL European Association for the Study of the Liver

ERC Enfermedades Renales Crónicas

ERM Enfermedades Reumáticas y Musculoesqueléticas

EULAR European Alliance of Associations for Rheumatology

FAISS Facebook AI Similarity Search

GDPR General Data Protection Regulation

GPT Generative Pre-trained Transformer

GPU Graphics Processing Unit

GROBID GeneRation Of BIbliographic Data

HIPAA Health Insurance Portability and Accountability Act

HTML HyperText Markup Language

IA Inteligencia Artificial

JSON JavaScript Object Notation

LlaMA Large Language Model Meta AI

LLM Large Language Model

MAGDA Multi-Agent Guideline-Driven Diagnostic Assistant

ML Machine Learning

MTEB Massive Text Embedding Benchmark

NDCG Normalized Discounted Cumulative Gain

NLM Neural Language Models

NLP Natural Language Processing

ODS Objetivos de Desarrollo Sostenible

PDF Portable Document Format

PLM Pre-trained Language Models

RAG Retrieval-Augmented Generation

RLHF Reinforcement Learning from Human Feedback

SFT Supervised Fine-Tuning

SLM Statistical Language Model

STS Semantic Textual Similarity

TEI Text Encoding Initiative

TXT Text File Format

VHB Virus de la Hepatitis B

VHC Virus de la Hepatitis C

XML Extensible Markup Language

Capítulo 1

Introducción

1.1. Contexto y justificación del trabajo

Las Enfermedades Reumáticas y Musculoesqueléticas (ERM) engloban un amplio espectro de patologías crónicas que afectan principalmente al aparato locomotor, aunque también pueden comprometer otros órganos y sistemas, afectando comúnmente las articulaciones de individuos de todas las edades y géneros, pero también pueden afectar los músculos, órganos internos y otros tejidos. La elevada prevalencia, el dolor y las complicaciones asociadas las sitúan entre las principales causas de deterioro en la calidad de vida a nivel global. La complejidad clínica, sumada a la variabilidad en las manifestaciones y en las respuestas al tratamiento, subraya la necesidad de contar con herramientas avanzadas. que faciliten la toma de decisiones, teniendo en cuenta la última evidencia disponible de las mayores asociaciones, European Alliance of Associations for Rheumatology (EULAR), y American College of Rheumatology (ACR)

En este contexto, la Inteligencia Artificial (IA) y los modelos de Lenguaje de Gran Tamaño (Large Language Model (LLM)) han revolucionado la forma en que se procesa y genera información a partir de datos textuales complejos. En la última década, los modelos de LLM han transformado radicalmente la interacción entre humanos y máquinas, marcando un hito en el desarrollo de la inteligencia artificial. Herramientas comerciales como ChatGPT han demostrado una notable capacidad para comprender y generar texto de manera coherente y versátil, facilitando su adopción en prácticamente todos los sectores. Esta revolución responde a la creciente necesidad de sistemas eficientes capaces de procesar grandes volúmenes de información y proporcionar respuestas rápidas, optimizando el flujo de trabajo y mejorando la toma de decisiones en diversas disciplinas.

El reciente avance de los LLM ha impulsado el desarrollo de soluciones inteligentes para la generación y recuperación de información en múltiples áreas. En el ámbito médico, donde el acceso a información precisa, actualizada y basada en evidencia es crucial, estas herramientas prometen transformar la toma de decisiones clínicas, la educación médica y la investigación. Sin embargo, los modelos fundacionales presentan limitaciones importantes, como la tendencia a generar respuestas incompletas o alucinaciones, lo que puede comprometer la precisión en entornos donde la fiabilidad es fundamental [5]. Además, es importante considerar que los LLM fundacionales adquieren su conocimiento mediante un preentrenamiento computacionalmente costoso que, si bien es eficaz en tareas generales, no permite incorporar información nueva y actualizada sin llevar a cabo un nuevo entrenamiento, lo que hace que su conocimiento

permanezca estático.

Para abordar estas limitaciones, la técnica RAG combina la generación del lenguaje con la recuperación de documentos relevantes, usados como fuentes externas de conocimiento. Esta técnica permite al modelo acceder a una base de conocimiento externa al entrenamiento, como las guías de prácticas clínicas oficiales y verificadas, asegurando que las respuestas generadas sean más precisas y confiables, reduciendo las alucinaciones y proporcionando una respuesta más fundamentada.

Las guías de práctica clínica, a su vez, son documentos normativos elaborados por instituciones de salud y sociedades científicas que contienen las últimas recomendaciones e indicaciones basadas en una evidencia demostrada. Por lo tanto, integrar estas fuentes de información en una arquitectura RAG permite mejorar la calidad de las respuestas en aplicaciones como asistentes médicos virtuales, permitiendo dar apoyo en decisiones clínicas [6, 7].

El uso de RAG en guías médicas permitiría resolver necesidades críticas en el ámbito médico, ya que se reducirían los errores en la generación de texto por alucinaciones al integrar un sistema de recuperación de documentos [2]. También permitiría a los profesionales de la salud e investigadores un acceso rápido y contextualizado a información basada en evidencia y referenciada propiciando una mejora en la toma de decisiones clínicas. Así, el uso de esta tecnología tiene el potencial de facilitar una práctica médica más dinámica y actualizada, en la que los profesionales pueden disponer de recomendaciones respaldadas por evidencia sin depender de su propia capacidad de actualización constante.

1.2. Objetivos del trabajo

El objetivo principal de este trabajo de fin de máster es la creación de un sistema RAG que incluya guías de práctica clínica de reumatología, mejorando el diagnóstico, el tratamiento y facilitando la consulta médica de especialistas reumatólogos para maximizar la probabilidad de diagnóstico y tratamientos correctos. El trabajo se centrará en el desarrollo de un modelo que sirva como base para un asistente virtual accesible tanto para expertos del ámbito sanitario (e.g., médicos de atención primaria, enfermeros...) como para pacientes, capaz de responder preguntas con alta precisión basándose en el contenido de las guías médicas del área de reumatología.

Para alcanzar este objetivo general, se plantean los siguientes objetivos específicos:

1.2.1. Objetivos Primarios

1. **Integración de guías clínicas de reumatología en un modelo de lenguaje basado en una arquitectura RAG**

2. **Evaluación del rendimiento del sistema RAG en comparación con un modelo de lenguaje fundacional (LLM)**: análisis comparativo de la precisión, eficiencia y calidad de las respuestas generadas por el modelo RAG frente a un modelo de lenguaje fundacional.

3. **Desarrollo de una interfaz visual**: implementación de una interfaz gráfica intuitiva que integre el modelo RAG optimizado, proporcionando a profesionales médicos y pacientes una herramienta automatizada para el diagnóstico y la consulta de guías médicas. Esta

interfaz permitirá respuestas rápidas y orientadas al usuario, mejorando el acceso a la información y reduciendo el tiempo de acción.

1.2.2. Objetivos Secundarios

1.1 **Objetivos específicos del objetivo 1**

1.1 **Identificación, recopilación y creación de un conjunto de datos a partir de documentos PDF** sobre patologías reumáticas.

1.2 **Construcción de un dataset especializado para un modelo RAG**: creación de un conjunto de datos basado en guías médicas verificadas, que servirá como base de conocimiento para el sistema de recuperación de información del modelo RAG.

1.3 **Evaluación de técnicas de procesamiento de texto**: análisis y selección de métodos de parseo, segmentación (chunking) y bases de datos vectoriales para el tratamiento de los documentos de las guías clínicas.

2.2 **Objetivos específicos del objetivo 2**

2.1 **Revisión del estado del arte de RAG en el ámbito médico**: revisión y análisis de las últimas implementaciones de modelos RAG aplicados a guías médicas, evaluando sus capacidades y limitaciones en este contexto.

2.2 **Comparación de diferentes arquitecturas RAG**: evaluación y contraste de modelos actuales en términos de métricas de rendimiento, precisión, coherencia y capacidad de adaptación en la generación de texto.

1.3. Impacto en sostenibilidad, ético-social y de diversidad

Esta sección analiza los posibles impactos, tanto positivos como negativos, derivados de esta tesis, considerando las tres dimensiones establecidas por la Competencia en Compromiso Ético y Global (CCEG): sostenibilidad, dimensión socioética y diversidad. El análisis se enmarca dentro de los Objetivos de Desarrollo Sostenible (ODS), garantizando un enfoque basado en la responsabilidad y la ética académica.

1.3.1. Sostenibilidad y medio ambiente

La creciente revolución de la IA ha transformado la industria médica, ofreciendo mejoras significativas en diferentes áreas, como la precisión diagnóstica, la eficiencia y los resultados de los pacientes [8]. Sin embargo, estas tecnologías requieren de un gran consumo de energía, y, consecuentemente, una alta producción de CO2 ambiental, llegando, en ocasiones, superar el consumo ahorrado [8]. De hecho, un estudio reciente concluyó que para el entrenamiento de un solo modelo de IA, se generaron cerca de 285.000kg de CO2, siendo una producción mayor que la de 5 automóviles en toda su vida útil [9].

Además, el funcionamiento de los modelos requieren de agua para enfriar los centros de datos y generar electricidad para proveerlos de energía, aumentando drásticamente el consumo de agua en los últimos años. Se plantea que, para 2027, la demanda de agua relacionada al uso de IA será de 6.600 millones de metros cúbicos.[9]. Durante el entrenamiento de uno de los modelos Generative Pre-trained Transformer (GPT), GPT-3 llegó a consumir 700.000 litros de agua dulce, contribuyendo en gran medida a la huella hídrica relacionada con el uso de IA [10].

Asimismo, la creciente demanda unidades de procesamiento gráfico (Graphics Processing Unit (GPU)) conlleva un impacto ambiental significativo, ya que su producción requiere la extracción de metales como el litio, el cobalto y el níquel [11]. Estos materiales son esenciales para la fabricación de otros componentes electrónicos avanzados, pero su obtención implica procesos de minería intensiva que pueden provocar desforestación, contaminación y pérdida de biodiversidad.

Por otro lado, los modelos RAG presentan una alternativa más sostenible, ya que no requieren un reentrenamiento constante para mantenerse actualizados. Los modelos actuales requieren un entrenamiento costoso y computacionalmente intensivo, lo que dificulta su actualización frecuente. En cambio, RAG permite acceder a informaciónactualizada sin la necesidad de desarrollar y entrenar nuevos modelos específicos de LLM, lo que se traduce en una reducción significativa del impacto ambiental asociado a estos procesos. Al disminuir la demanda de recursos computacionales, como las GPU, también se reduce la necesidad de extraer materiales críticos como el litio, el cobalto y el níquel, mitigando así las graves consecuencias ambientales derivadas de la minería intensiva, como la deforestación, la contaminación y la pérdida de biodiversidad.

1.3.2. Socio-ético

Actualmente, el uso de la IA en el ámbito de la salud se centra en la obtención de resultados, mientras que su aplicación en el diagnóstico y las consultas médicas basadas en guías médicas está en desarrollo. De hecho, el uso de modelos LLM y los asistentes virtuales no especializados presentan una baja precisión, ya que no siempre se basan en documentos verificados. [12]

Sin embargo, la implementación de recuperación en la generación de respuestas permitiría el acceso a información actualizada en tiempo real. Además, al proporcionar la visualización de las fuentes de utilizadas mejorarían la transparencia en la toma de decisiones médicas y reducirían el riesgo de desinformación. [13] y [14]. Se debe tener en cuenta, que debido a la importancia de la precisión en un área tan delicada como la medicina y el diagnóstico, se han formulado modelos RAG específicos para áreas como biomedicina [15], hepatología [16, 6] o cirugía [17], con documentación médica, como guías médicas hepatológicas, pero no en patologías reumatológicas [6].

Finalmente, los asistentes virtuales basados en IA requieren de una infraestructura potente, no siempre disponible en todos los hospitales, especialmente en regiones con menos recursos. En este sentido, aunque un modelo RAG depende de un modelo de lenguaje de gran tamaño (LLM), este puede ser alojado en la nube, mientras que la base de datos vectorial y los sistemas de recuperación de información pueden mantenerse en servidores locales. Esta arquitectura híbrida facilita el despliegue en áreas rurales donde el acceso a médicos especializados es limitado, permitiendo la provisión de diagnósticos precisos y oportunos sin necesidad de infraestructura computacional avanzada en el sitio. De esta manera, contribuiría a reducir la brecha digital

existente, permitiendo que centros de salud con menos recursos también se beneficien de los avances en inteligencia artificial médica [18].

Además, un modelo RAG con memoriapodría facilitar el acceso a la IA por parte de usuarios con distintos niveles de conocimiento, adaptándose a sus necesidades específicas. Esto sería posible mediante el uso de técnicas de prompt engineering, ajustando la formulación de las respuestas en función del contexto y la situación particular del usuario. De esta manera, se democratizaría su uso más allá del ámbito profesional, haciéndolo accesible también a pacientes y cuidadores.

1.3.3. Diversidad y derechos humanos

Se van a usar guías clínicas elaboradas por comités científicos y sociedades científicas, caracterizadas por su rigor y un gran sentido de la diversidad, caracterizadas por su rigor, en las cuales se presupone que el panel de elaboración es diverso. Por lo tanto, el uso de estas guías mediante RAG puede contribuir a reducir el sesgo, asegurando que la información proporcionada esté basada en el consenso de expertos y representando adecuadamente a distintos grupos y contextos clínicos [19].

Este enfoque facilita el acceso equitativo a información médica confiable, mejorando la atención en comunidades marginadas. Además, al adaptar sus recomendaciones en función del usuario, el sistema permite que información médica compleja sea comprensible para cualquier persona [19].

La arquitectura RAG no manejará datos confidenciales de pacientes ni historiales clínicos, limitándose a utilizar información procedente de fuentes validadas y guías clínicas públicas, y su implementación se ajusta plenamente a las normativas vigentes en materia de protección de datos, como el General Data Protection Regulation (GDPR) o la Health Insurance Portability and Accountability Act (HIPAA), garantizando la privacidad y seguridad de la información [20, 21].

1.4. Enfoque y metodología seguida

Los pasos realizados para establecer el modelo RAG fueron los siguientes:

1. **Estudio de la literatura:** Se iniciará con un estudio del estado del arte sobre LLM y los modelos RAG desarrollados en el ámbito médico.

2. **Preparación del dataset:** Se obtendrán las guías médicas en formato Portable Document Format (PDF), se realizará un análisis y descomosición de texto para obtener información (parseo), se fragmentarán (chunking), se realizará una transformación vectorial (embedding) y se creará el repositorio de vectores.

3. **Selección del modelo:** Se estudiarán los diferentes modelos de LLM como generadores de respuesta y los diferentes retrievers disponibles comercialmente. Se evaluará su capacidad de recuperar información a partir del vectorstore, la precisión y rendimiento.

4. **Modelo RAG:** Con los modelos seleccionados, se formulará la arquitectura RAG adecuada para el proceso de los datos. El proceso se enfocará en mejorar las respuestas de LLM fundacionales.

5. **Evaluación del modelo:** Se estudiará el rendimiento del modelo mediante métricas de precisión, comparando las respuestas proporcionadas por el modelo respecto a preguntas de exámenes oficiales y comparándolas con el solucionario oficial.

6. **Interfaz gráfica web:** La arquitectura finalmente diseñada se integrará a una interfaz web (local) que permitirá la interacción usuario-asistente virtual, haciendo uso de herramientas como Streamlit, gradio,...

1.5. Planificación del trabajo

Se han tenido en cuenta las fechas de entrega de las PEC para la distribución de trabajo en 4 bloques. En la figura Figura 1.1 se muestra la distribución del tiempo de las tareas a realizar en cada bloque, descritas a continuación:

1.5.1. Distribución de tareas

Para lograr los objetivos de este trabajo final de máster, se definieron las siguientes tareas

1. **Investigación y revisión de literatura (3 semanas)**

   El objetivo es realizar una profunda revisión de la literatura, enfocada en los modelos LLM, la arquitectura RAG y sus aplicaciones previas en consultas médicas y guías clínicas.

   - Identificar los modelos LLM candidatos y sus aplicaciones previas.
   - Revisar la arquitectura RAG, sus componentes y sus aplicaciones.
   - Resumir los hallazgos de la investigación.

2. **Recopilación de documentos y procesamiento (2 semanas)**

   El objetivo es recopilar guías clínicas en formato PDF y procesarlas para posteriormente crear una base de datos vectorial a partir de la cual se recupere contenido relevante para fundamentar la respuesta del LLM. Se estudiarán los diferentes métodos de parseo disponibles y se realizarán pruebas para determinar el más eficiente parala conversión del formato PDF en texto plano

   Debido a la complejidad de la transformación de imágenes en texto plano, solo se vectorizará el contenido textual, dejando de lado las figuras. Sin embargo, debido a la importancia de la información contenida en las tablas, sí que se incluirá dicha información, convirtiendo a formato tabular para evaluar si esto mejora la precisión del modelo.

   - Data collection: Recopilar guías médicas verificadas
   - Data processing: Identificar, aplicar y evaluar los métodos de parseo, segmentación (*chunking*) y vectorización.

3. **Selección del modelo y análisis preliminar (2 semanas)**

   Se evaluarán diferentes arquitecturas RAG, combinando diversos métodos de recuperación (*retrievers*) y generación (*generators*) para seleccionar la mejor configuración para la arquitectura RAG del trabajo final de máster.

   - Realizar pruebas iniciales para evaluar la idoneidad de cada modelo según los datos disponibles.
   - Comparar los modelos basados en métricas de rendimiento.
   - Documentar el proceso de selección y justificar la elección del modelo final.

4. **Modelo RAG (3 semanas)**

   Se implementará la arquitectura seleccionada para procesar la información del vector store y generar respuestas relevantes basadas en las guías médicas. Se ajustarán los parámetros para tratar de optimizar la relevancia, configurando las estrategias de indexación y búsqueda, así como los parámetros iniciales del modelo.

   - Seleccionar el modelo base para la recuperación de información y la generación de respuestas.
   - Indexar los documentos utilizando técnicas de *embedding*.
   - Entrenar el modelo con datos médicos específicos para especializar el sistema.

5. **Evaluación del modelo y optimización (3 semanas)**

   Se analizarán métricas de rendimiento como la precisión y la capacidad del modelo para adaptar su lenguaje al usuario. Se optimizará el modelo ajustando los componentes del sistema necesarios.

   - Evaluar la precisión y la coherencia del lenguaje generado.
   - Analizar el rendimiento del modelo y aplicar optimizaciones.

6. **Creación de una interfaz web basada en el mejor modelo (2 semanas)**

   Se diseñará una interfaz gráfica web con herramientas de prototipado como Streamlit o Gradio que integre el mejor modelo de arquitectura RAG, permitiendo la interacción del usuario con el sistema.

   - Diseñar una interfaz web clara y fácil de usar.
   - Integrar el modelo dentro de la interfaz.
   - Evaluar la funcionalidad de la interfaz para asegurar su correcto funcionamiento.

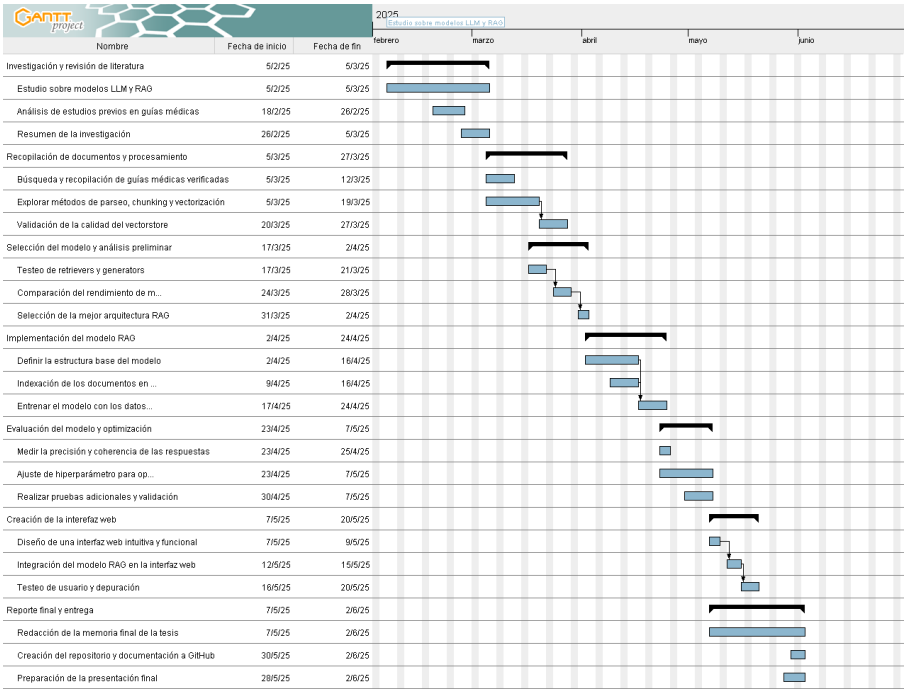
7. **Reporte final (3 semanas)**

   En esta fase se redactará la memoria final del trabajo final de máster, se diseñará y grabará la presentación, y se creará un repositorio en GitHub con el código empleado.

   - Redactar la memoria final.

- Crear el repositorio en GitHub.
- Preparar y grabar la presentación.
- Realizar la entrega final de la tesis.

1.5.2. Diagrama de Gantt



Nombre	Fecha de inicio	Fecha de fin
Investigación y revisión de literatura	5/2/25	5/3/25
Estudio sobre modelos LLM y RAG	5/2/25	5/3/25
Análisis de estudios previos en guías médicas	18/2/25	26/2/25
Resumen de la investigación	26/2/25	5/3/25
Recopilación de documentos y procesamiento	5/3/25	27/3/25
Búsqueda y recopilación de guías médicas verificadas	5/3/25	12/3/25
Explorar métodos de parseo, chunking y vectorización	5/3/25	19/3/25
Validación de la calidad del vectorstore	20/3/25	27/3/25
Selección del modelo y análisis preliminar	17/3/25	2/4/25
Testeo de retrievers y generators	17/3/25	21/3/25
Comparación del rendimiento de m...	24/3/25	28/3/25
Selección de la mejor arquitectura RAG	31/3/25	2/4/25
Implementación del modelo RAG	2/4/25	24/4/25
Definir la estructura base del modelo	2/4/25	16/4/25
Indexación de los documentos en ...	9/4/25	16/4/25
Entrenar el modelo con los datos...	17/4/25	24/4/25
Evaluación del modelo y optimización	23/4/25	7/5/25
Medir la precisión y coherencia de las respuestas	23/4/25	25/4/25
Ajuste de hiperparámetro para op...	23/4/25	7/5/25
Realizar pruebas adicionales y validación	30/4/25	7/5/25
Creación de la interefaz web	7/5/25	20/5/25
Diseño de una interfaz web intuitiva y funcional	7/5/25	9/5/25
Integración del modelo RAG en la interfaz web	12/5/25	15/5/25
Testeo de usuario y depuración	16/5/25	20/5/25
Reporte final y entrega	7/5/25	2/6/25
Redacción de la memoria final de la tesis	7/5/25	2/6/25
Creación del repositorio y documentación a GitHub	30/5/25	2/6/25
Preparación de la presentación final	28/5/25	2/6/25

Figura 1.1: Planificación temporal del trabajo final de máster

1.5.3. Análisis de riesgos

Antes de la elaboración del proyecto, se han contemplado los posibles riesgos que pueden surgir y causar un impacto negativo en la elaboración del trabajo, determinando el nivel de riesgo y las posibles alternativas para mitigarlos.

1. **Problemas en la creación del repositorio**

   Impacto: Alto

   El proceso de parseo es esencial para implementar un modelo RAG, ya que garantiza la extracción y estructuración óptima del texto de los documentos PDF originales. Sin embargo, este paso puede presentar problemas como la presencia de contenido visual (imágenes), formato desorganizado y pérdida de información.

   Para mitigar estos riesgos, se establecen herramientas de parseo alternativas y paralelas para estudiar las diferencias y escoger la herramienta más adecuada. También se han seleccionado guías médicas con la estructura lo más similar y simple posible para evitar diferencias entre documentos que alteren el proceso.

2. **Diversidad entre guías médicas**

   Impacto: Medio

   Las guías médicas no fueron diseñadas específicamente para ser procesadas por modelos de análisis de datos, sino que su propósito es servir como referencia y apoyo para profesionales de la salud. Aunque muchas están reconocidas por la ACR, cada una presenta una estructura particular, lo que dificulta su tratamiento automatizado.

   Por ello, se deben considerar las posibles limitaciones al utilizarlas en un entorno computacional. Como estrategia de mitigación, se priorizan las guías que permitan su integración en software de recuperación de información.

   Además, cabe destacar que algunas guías clínicas pueden estar protegidas por derechos de autor, por lo que su uso en este trabajo se ha realizado exclusivamente con fines académicos, sin ningún propósito comercial.

3. **Dificultades en la vectorización**

   Impacto: Medio

   Los modelos RAG recuperan la información basándose en la proximidad vectorial, por lo que la correcta vectorización de la información impacta directamente en la calidad de la recuperación de documentos. La conversión del contenido de los documentos PDF a formato Text File Format (TXT) es fundamental para una representación adecuada de los datos. Una transformación deficiente puede inducir errores en la interpretación del texto.

   Para optimizar este proceso, se realiza una segmentación eficiente de los documentos en fragmentos de tamaño óptimo (chunks) para preservar el contexto en la recuperación de información. Asimismo, se evalúan diferentes enfoques de vectorización en paralelo para determinar la metodología más eficaz.

4. **Dificultades en el modelo RAG**

   Impacto: Medio

   El modelo RAG enfrenta desafíos de ajuste fino, interpretabilidad y eficiencia en la recuperación de documentos. Estos riesgos pueden clasificarse en dos factores. Primero, al tratarse de una arquitectura relativamente nueva, existen pocos ejemplos de referencia para guiar el desarrollo. Segundo, su demanda computacional requiere de GPU potentes, limitando las opciones de implementación, pero no resultan significativamente perjudiciales. Se debe tener en cuenta que, si la fase de retrieval no es efectiva, el modelo RAG generará respuestas basadas en documentos irrelevantes

   Para abordar estos problemas, se evaluaron distintas plataformas y alternativas a Google Colab, realizando pruebas en diferentes infraestructuras que se ajusten a las necesidades del proyecto. Además, se optimiza el modelo mediante ajustes de los componentes del sistema para mejorar la eficiencia computacional. Para mitigar la demanda computacional, una posible solución es utilizar APIs de modelos en la nube, lo que evita la necesidad de desplegar infraestructuras locales avanzadas, facilitando la implementación en entornos con recursos limitados

5. **Coste económico**

   En algunos LLM como GPT-4 o Claude3 tienen un coste asociado, que corresponde al tipo de modelo usado y el número de tokens requeridos, tanto en el prompt de entrada (input) como en la respuesta del LLM (output), limitando la complejidad del modelo. Se comprobaron alternativas para establecer la más económica, que optimice el coste y mantenga la calidad del resultado, eligiendo modelos con un coste por token menor, como GPT-4o mini.

6. **Evolución constante del ámbito médico**

   Impacto: Medio

   El entorno médico y clínico está en constante evolución, creando y mejorando nuevas guías clínicas consantemente, por lo que es crucial que el sistema propuesto tenga la misma capacidad de evolucionar y acceder a información actualizada en tiempo real.

   Como medida de mitigación, se trabajaron exclusivamente las guías más recientes disponibles hasta la fecha de marzo de 2025, asegurando que el sistema se base en información actualizada. Además, se prioriza la extracción de datos únicamente de fuentes fiables, evitando el uso de información no verificada.

7. **Evaluación del sistema**

   Impacto: Medio

   El modelo de arquitectura RAG proporcionará respuestas generadas por el LLM basándose en las guías médicas. Para evaluar la precisión y relevancia de estas respuestas, se requerirá la validación de especialistas médicos en enfermedades reumáticas y musculoesqueléticas.

   Con el fin de reducir esta dependencia, se propone el uso de un LLM externo capaz de comparar y evaluar automáticamente las diferencias entre las respuestas, facilitando así el proceso de revisión y mejorando la eficiencia del análisis.

8. **Tiempo de redacción de la memoria final**

   Impacto: Bajo

   El tiempo de entrega del TFM puede verse afectado por posibles retrasos en el desarrollo y validación del modelo, comprometiendo la calidad del informe final. Este riesgo podría derivar en una documentación incompleta o una falta de análisis en profundidad.

   Para mitigar este riesgo, se establecen cronogramas con hitos claros, revisiones periódicas del avance y la realización paralela de tareas clave para optimizar los recursos disponibles. Asimismo, se realiza la redacción de la memoria final a lo largo del trabajo, avanzando en paralelo.

1.6. Breve resumen de los productos obtenidos

Al finalizar el Trabajo Final de Máster, se habrá obtenido uno de los primeros modelos RAG basado en guías médicas verificadas que se podrá emplear mediante la plataforma web en formato de asistente virtual.

Se va a obtener:

- **Un conjunto de datos** con los cuales construir la base de datos vectorial
- **Una base vectorial** con los documentos procesados
- **Una interfaz web** que emplee el modelo RAG, basándose en las guías médicas para interactuar con los usuarios de forma amigable teniendo en cuenta la experiencia del usuario.
- **Un repositorio de GitHub** con el código empleado y los obtenibles resultantes, accesible a través del siguiente enlace: **ENLACE DE GITHUB**
- **Una memoria** con la evolución del proyecto.
- **Una presentación virtual** con la tesis final.

1.7. Breu descripció dels altres capítols de la memòria

Breu explicació dels continguts de cada capítol i la seva relació amb el projecte global.

Capítulo 2

Marco teórico

2.1. Introducción a los LLM

Los recientes avances en el campo de la inteligencia artificial (IA) ha permitido el desarrollo de Modelos de Lenguaje de Gran Tamaño (Large Language Models, LLM) como LlaMA, ChatGPT o Claude; algoritmos potentes capaces de sintetizar grandes cantidades de datos y realizando una amplia variedad de tareas en respuesta a comandos humanos.

Los LLM, como GPT-4, funcionan mediante una combinación de Deep Learning, entrenamiento masivo con grandes volúmenes de texto y generación de texto basada en probabilidades. Específicamente, los LLM están basados en redes neuronales profundas, específicamente en la arquitectura Transformers introducida en 2017 [22], [23]. Dependiendo de su enfoque, pueden emplear modelos autorregresivos (como GPT), que generan texto prediciendo la siguiente palabra en función del contexto previo, asignando probabilidades a cada posible término y eligiendo el más probable. También pueden utilizar modelos de autoencoding (como Bidirectional Encoder Representations from Transformers (BERT)), que aprenden representaciones de texto enmascarando palabras dentro de una oración y tratando de predecirlas, lo que permite una mejor comprensión del contexto bidireccional [22].

2.1.1. Historia de los LLM

Los LLM sirven como piedra angular en el procesamiento del lenguaje natural (NLP, Natural Language Processing). De hecho, el lenguaje es la herramienta que permite expresar pensamientos y la comunicación entre dos entidades. Sin embargo, las maquinas carecen de la capacidad intrínseca de comprender y comunicarse en un lenguaje humano, por lo que requieren de algoritmos de IA para disponer de esta capacidad.

El [**?**] abarca una amplia gama de enfoques que incluyen modelos estadísticos, probabilísticos y modelos de lenguaje [1]. Inicialmente, los modelos de procesamiento de lenguaje se basaban en técnicas estadísticas como los modelos de Markov ocultos (HMM) o los Modelos de Lenguaje Estadísticos (Statistical Language Model (SLM)), que calculaban la probabilidad de ocurrencia de palabras o secuencias en un corpus de datos. Con el avance del Machine Learning (ML), surgieron los modelos basados en redes neuronales (como Word2Vec o GloVe), y posteriormente los modelos de LLM, basados en la arquitectura Transformers, que han revolucionado la generación y comprensión del lenguaje natural.

De forma general, los modelos de lenguaje han sufrido diferentes evoluciones que han permitido el desarrollo de los modelos actuales, los LLM. En la Figura 2.1, se puede observar los modelos más relevantes de cada tipo de LM, desde el nacimiento de los más básicos (N-grams) hasta los modelos de última generación basados en la arquitectura Transformer.

Si bien GPT-4 ha sido uno de los avances más significativos en este campo, actualmente existen otros modelos de alto rendimiento, como GPT-4o, Claude 3.5 Sonnet o Gemini 2.5 entre otros. Estos modelos continúan mejorando en eficiencia, capacidad de razonamiento y generación de texto más precisa y contextualizada.

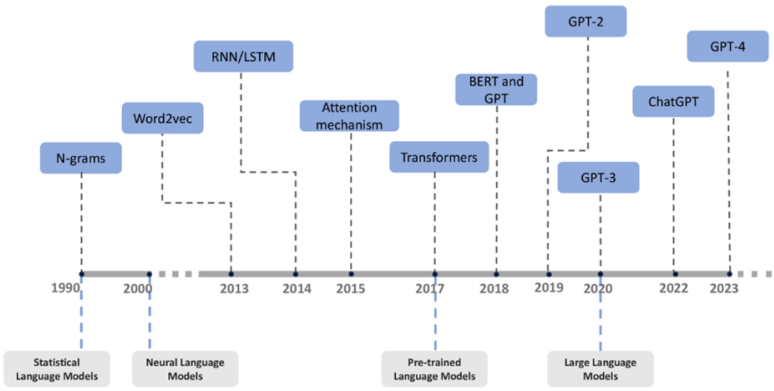


Figura 2.1: Historia y evolución de los distintos modelos de procesamiento de lenguaje natural. Extraída de [1].

A continuación, se presentan las principales categorías de modelso de lenguaje, destacando sus características, ventajas y limitaciones:

- **Statistical Language Models (SLMs)**: Son modelos matemáticos que emplean las propiedades contextualmente relevantes del lenguaje natural desde una perspectiva estadística probabilística. Su esencia radica en determinar la probabilidad de que una oración aparezca dentro de un contexto a partir de las probabilidades condicionales consecutivas de las unidades lingüísticas simples (palabras o letras) [24]. Su mayor limitación radica en que, al tener solo en cuenta las dos primeras palabras precedentes a cada término, se pierde significativamente su precisión y contexto.

- **Neural Language Models (NLM)**: Emplean redes neuronales para predecir las probabilidades de palabras posteriores dentro de secuencias lingüísticas, y son capaces de comprender el concepto de las palabras mediante vectorización y su ubicación. Es decir, convierten las palabras a lenguaje binario, se transforman en vectores con una ubicación fija, de manera que pueden simular las relaciones entre palabras a través de los ángulos entre los vectores. Su mayor limitación es que disponen de un contexto limitado y, pese a generar un texto coherente, no poseen una comprensión real del significado, llegando a generar respuestas sintácticamente correctas pero sin sentido real [25, 26].

- **Pre-trained Language Models (PLM)**: Estos modelos se someten a un entrenamiento inicial con un gran volumen de texto no etiquetado, permitiendo captar estructuras

lingüísticas fundamentales, como la sintaxis o la semántica, para posteriormente, realizar un segundo entrenamiento con conjuntos de datos más pequeños para "afinar" el conocimiento (fine tuning), de manera que se pueden realizar tareas más específicas sin perder calidad. Sin embargo, al depender del pre-entrenamiento, no se puede actualizar dinámicamente con nueva información sin ser reentrenados, además están enfocados en tareas más específicas en las que son entrenadas.

- **Large Language Models (LLM)**: Son los modelos más modernos. Se entrenan con corpus de texto con decenas de miles de millones de parámetros para comprender los comandos humanos. Igual que los PLM, pasan por un pre-entrenamiento inicial, para posteriormente ser alineados con los valores humanos en lugar de especializarse en un dominio específico como los PLMs. Destacan por su adaptabilidad y capacidad de aprovechar el contexto, pero su crecimiento depende del número de parámetros y datos de entrenamiento.

2.1.2. LLM: fundamentos, estado actual y familias

En los últimos años, ha habido diferentes factores que han permitido un rápido avance en los LLM. La disponibilidad de grandes volúmenes de texto de diferentes fuentes ha permitido mejorar su capacidad para generalizar y realizar múltiples tareas. Además, el crecimiento exponencial de la potencia computacional ha permitido entrenar modelos cada vez más grandes, ya que el desarrollo de hardware especializado como GPU ha acelerado su crecimiento (GPT-3 habría tardado 355 años en entrenarse en una sola GPU, pero con 1024Xa100 GPUs solo tardó 34 días) [1]. El desarrollo de la arquitectura Transformer superó enfoques anteriores como redes neuronales recurrentes y convolucionales, optimizando la atención, el preentrenamiento y la eficiencia computacional [23].

Tomando como ejemplo a GPT-3, se pueden observar cuatro grandes principios que siguen los LLM para generar sus respuestas:

- **Entrada y codificación**: Reciben secuencias de palabras (tokens), convirtiendo cada palabra a un vector numérico (one-hot) dentro de un vocabulario de términos.
- **Embedding**: Para mejorar la eficiencia, los vectores one-hot se transforman en representaciones más compactas.
- **Codificación posicional**: Al carecer de estructura secuencial, se agregan codificaciones posicionales usando funciones trigonométricas para capturar el orden de las palabras en una oración.
- **Matriz de entrada**: La combinación de embeddings y codificaciones posicionales generará la matriz de entrada, que se procesará en las capas de transformadores.

Los atributos y comportamientos de los LLM están profundamente vinculados con los procesos del entrenamiento, caracterizado por tres etapas principales: preentrenamiento, ajuste fino supervisado (Supervised Fine-Tuning (SFT)) y aprendizaje de refuerzo por feedback humano (Reinforcement Learning from Human Feedback (RLHF)) [27]. Estas fases permiten que los modelos no solo adquieren una comprensión básica del lenguaje, sino que también mejoren su capacidad para generar respuestas precisas y alineadas con los criterios humanos de calidad.

- **Pre-entrenamiento**: Permite adquirir conocimientos y habilidades fundamentales, entrenando mediante la predicción autorregresiva de tokens dentro de secuencias de texto. De esta manera, desarrollan una comprensión profunda de la sintaxis y habilidades de razonamiento, estableciendo una base sólida para su posterior ajuste y refinamiento [28].
- **SFT**: Entrena el modelo con un conjunto de datos anotado compuesto por pares de instrucciones y respuestas, mejorando la capacidad de los modelos para seguir instrucciones específicas, ofreciendo respuestas más precisas y útiles. Se ha confirmado la eficacia de este paso para un rendimiento adecuado en tareas no vistas previamente, mostrando capacidad de generalización [29, 30].
- **RLHF**: Se utiliza la evaluación de respuestas por parte de humanos para optimizar el comportamiento del modelo, para maximizar la recompensa determinada, usando algoritmos de aprendizaje por refuerzo. Esto permite que el modelo genere respuestas más alineadas con las expectativas humanas, mejorando la calidad de las respuestas y reduciendo la generación de contenido sesgado [31].

2.1.3. Limitaciones de los LLM

Los modelos de LLM han demostrado ventajas significativas frente a otros modelos de lenguaje. Sin embargo, a pesar de su eficiencia en aplicaciones generales, los modelos disponibles comercialmente presentan limitaciones en el ámbito clínico y médico, llegando a generar, en ocasiones, "alucinaciones", es decir, respuestas con referencias incompletas o incluso completamente inventadas. Esta falta de precisión compromete la fiabilidad de los resultados, lo que, en un contexto de diagnóstico médico, puede tener consecuencias graves. [6].

A pesar de su capacidad de generar texto coherente y realizar tareas complejas, los LLM tienen deficiencias estructurarles, como la falta de comprensión real, ya que no disponen de un razonamiento ni conocimiento como un humano, sino que identifican patrones en datos, por lo que pueden llegar a generar información inconsistente [32, 33].

Sin embargo, se deben reconocer las desventajas que acompañan estos modelos.

- **Sesgo**: Las decisiones pueden presentar sesgos en contra de poblaciones o grupos más marginados, ya que durante el entrenamiento con datos extensos y no estructurados, se pueden absorber representaciones erróneas y comportamientos excluyentes. Esta problemática se mitiga eliminando los datos claramente sesgados del conjunto de entrenamiento, sin embargo esta solución no es completamente efectiva, ya que puede afectar la capacidad del modelo para generalizar correctamente y reducir su efectividad en ciertas tareas [34]. El sesgo no solo proviene de los datos de entrenamiento, sino también de su funcionamiento interno. Por ejemplo, el modelo puede generar el siguiente token basándose en sus propias predicciones previas, lo que puede dar lugar a errores acumulativos (sesgo de exposición), o cuando prioriza el conocimiento aprendido durante el entrenamiento sobre la información proporcionada en la entrada (sesgo de conocimiento paramétrico) [4].
- **Seguridad y alucinaciones**: Los LLM tienen diversas aplicaciones en industrias como el ámbito médico, legal o de finanzas, por lo que la veracidad de la información es especialmente relevante. En ocasiones, pueden generar resultados que difieren del contexto proporcionado o del conocimiento factual, denominados *alucinaciones*. [4]

- **Privacidad**: Los modelos de lenguaje a gran escala pueden implicar riesgos para la privacidad, especialmente si se entrenan con datos sensibles o sin un adecuado control de filtrado. La exposición de información personal o confidencial puede representar un problema crítico en su implementación [35]. Para mitigar estos riesgos, se implementan guardrails, que incluyen técnicas como la anonimización de datos, la detección y eliminación de información sensible en las respuestas del modelo, el uso de mecanismos de acceso restringido y auditorías periódicas del sistema. Estas estrategias buscan garantizar que los modelos cumplan normas de privacidad y eviten la divulgación de datos confidenciales [36, 37].

De hecho, las alucinaciones en los LLM fundacionales como ChatGPT o Gemini pueden parecer mostrar capacidades de razonamiento muy elevadas, por lo que pueden pasar desapercibidas y dar lugar a resultados falsos o sin fundamento. Esto puede generar desconfianza en su aplicación en ámbitos críticos, como el legal o el médico, donde la precisión y la fiabilidad de la información son fundamentales [38].

2.2. Sistemas RAG en LLM

Tanto los LLM como IA han demostrado un gran potencial para mejorar la eficiencia de la interacción de los médicos y los pacientes con los sistemas de atención médica [39]. Aún así, debido a que toda la información procede de los datos proporcionados por el entrenamiento, esta información no se actualiza regularmente, pudiendo inducir a errores.

Para abordar esta problemática, se ha propuesto un sistema avanzado de recuperación de información que garantiza que las respuestas generadas se basen en un conocimiento verificable y previamente establecido. Este enfoque, conocido como Retrieval-Augmented Generation (RAG), combina un modelo generativo basado en LLM con un mecanismo de recuperación de información, integrando el conocimiento del modelo con información externa relevante para mejorar la precisión y fiabilidad de las respuestas.

La generación aumentada por recuperación permite la incorporación de datos personalizados en los LLM, especializando su contenido y reduciendo las alucinaciones [40], sin necesidad de volver a entrenar el modelo de nuevo.

Por lo tanto, la combinación de LLM con RAG representa una evolución significativa en la generación de texto, ya que permite que los modelos de lenguaje utilicen información externa relevante durante la generación, mejorando la precisión y relevancia de las respuestas sin depender exclusivamente del conocimiento interno del modelo.

El uso de modelos RAG en combinación con LLM ha experimentado un aumento reciente, demostrando que, con un correcto pre-procesamiento de los datos, pueden alcanzar una precisión cerca del 99.0 % [41].

2.2.1. Arquitectura de RAG. Workflow

La arquitectura Retrieval-Augmented Generation combina la generación de un texto mediante un LLM con la recuperación de información externa para mejorar la calidad y la precisión de las respuestas generadas. Esta arquitectura se basa en dos etapas principales: la búsqueda y recuperación de información relevante en una base de datos (retriever) y posteriormente

se emplea la información recuperada para generar una respuesta más precisa y contextualizada, accediendo a información que el LLM no disponía durante su entrenamiento. De este modo, el enfoque RAG permite al modelo LLM una mayor versatilidad y eficiencia, especialmente en tareas que requieren de conocimiento específico y actualizado sin necesidad de un re-entrenamiento. [16]

En general, una arquitectura RAG inicia con la consulta de entrada, la cual pasa al módulo recuperador que accederá al vectorstore para buscar los k documentos más relevantes. Posteriormente, se procesarán junto con la consulta al módulo generador como contexto para que produzca una salida condicionada. La Figura 2.2 muestra el workflow genérico de las arquitecturas RAG.

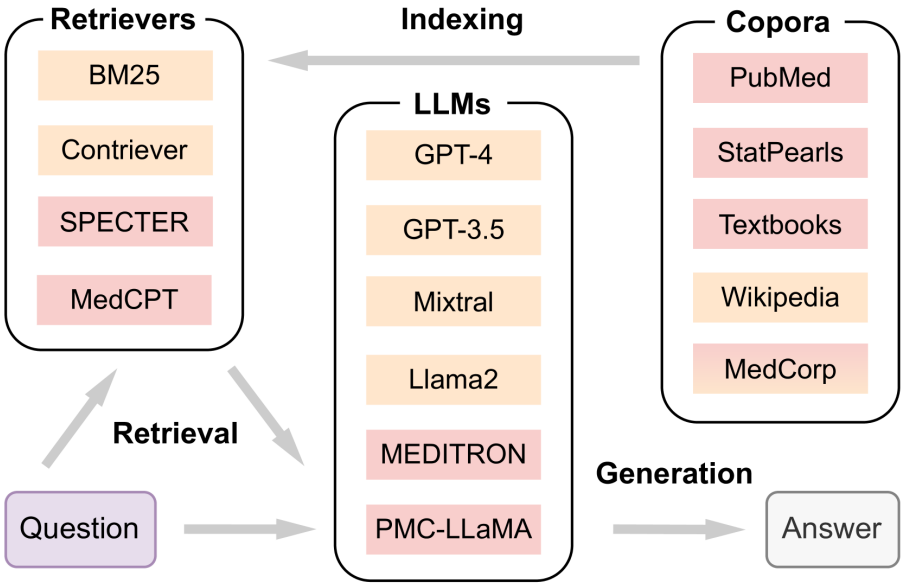


Figura 2.2: Esquema de una arquitectura RAG. Extraído de [2].

Framework

Durante el desarrollo de sistemas de RAG, el uso de frameworks especializados facilita la integración entre los modelos de lenguaje y la bases de datos vectoriales, optimizando la eficiencia y precisión en la recuperación y generación de respuestas. Actualmente existen múltiples herramientas especializadas, destacando LangChain, LlamaIndex y Haystack, cada una con sus características. Estos frameworks proporcionan una infraestructura que permite la implementación de técnicas avanzadas cómo la búsqueda semántica, re-ranking y generación de texto en base a un contexto, imprescindibles en un modelo de RAG.

La elección de uno de estos frameworks dependerá de sus características como la escalabilidad, compatibilidad con los LLM y la flexibilidad con la personalización del workflow diseñado.

- **LangChain** [42]: Está diseñado para facilitar la integración de LLM con fuentes de datos externas, bases de datos vectoriales y herramientas adicionales. Permite personalizar cada

componente del pipeline, además de herramientas nativas como la memoria conversacional, haciéndola una gran opción en modelos de RAG. Su estructura básica se centra en los retrievers (módulo de recuperación basada en embeddings), chains (flujos de trabajo), memoria y agentes (que ejecutan workflows dinámicos con toma de decisiones).

- **LlamaIndex** [43]: Está especializado en la indexación y recuperación eficiente de información desde documentos estructurados (y semiestructurados), optimizando la integración con modelos de LLM para generar respuestas basadas en información. Es compatible con múltiples LLM y fuentes de datos. Su arquitectura se basa en cargadores de documentos (Document Loaders), Indexadores (estructurando la información de bases vectoriales) y Query Engine (el motor de búsqueda semántica).

- **Haystack** [44]: Es un framework de código abierto que permite crear aplicaciones LLM y canales generativos con recuperación mejorada y sistema de búsqueda en grandes colecciones de documentos. Su estructura modular permite combinar diferentes motores de búsqueda, con pipelines híbridos e integración con bases de datos vectoriales. Su arquitectura se basa en Document Stores (almacenajes basados en SQL-NoSQL), Retrievers (módulo de recuperación), Readers (Modelo de NLP para la extracción) y el Generator (Modelo de LLM para la generación de respuesta).

Pre-procesados: parseo, chunking y vectorstore

Para el eficiente procesamiento de datos en la recuperación de la información, las arquitecturas RAG se fundamentan en dos componentes clave: la recuperación y la generación. Para que el modelo pueda acceder a los documentos más relevantes, se debe crear un repositorio de los mismos, requiriendo un preprocesamiento adecuado. Esto implica convertir la información desde un formato no estructurado, como los archivos en formato PDF de las guías clínicas a un formato estructurado y accesible para su análisis computacional. Este proceso implicará la extracción de texto de los documentos, su normalización y limpieza, así como la transformación de la información textual en vectores numéricos mediante técnicas de embeddings. Finalmente, estos embeddings se almacenarán en una base de datos vectorial (VectorStore), que permitirá gestionar grandes volúmenes de información de forma eficiente, facilitando la búsqueda, comparación y análisis semántico, optimizando la recuperación de documentos relevantes y mejorando el rendimiento general del sistema.

Parseo: Extracción del texto

El primer paso en el proceso de preprocesamiento de documentos es el análisis de los datos o parseo, que implica extraer y organizar la información desde fuentes no estructuradas (como los archivos PDF de las guías clínicas reumatológicas) para convertir los datos en un formato adecuado para el procesamiento computacional de la información. Este proceso es crítico, ya que debido a la complejidad del formato PDF y su carencia de una estructura semántica clara, lo hace más difícil de manejar en comparación con otros formatos como Extensible Markup Language (XML) o HyperText Markup Language (HTML), que tienen una jerarquía definida en los datos.

Existen múltiples herramientas que permiten extraer el contenido de documentos en formatos diferentes, como XML o Markdown. El formato PDF en específico es complejo de extraer, ya que carece de una estructura semántica clara como otros formatos como XML o HTML, que definen los datos con una estructura jerárquica personalizada.

Los métodos de parseo se pueden clasificar en dos enfoques principales:

El primer enfoque sigue un conjunto de reglas predefinidas para extraer la información de los documentos, basándose en patrones o estructuras bien definidas dentro del documento. De esta manera, emplea expresiones regulares o delimitadores ya presentes en los documentos, siendo altamente preciso en documentos con formatos uniformes y bien estructurados, como XML. Este enfoque es rápido y eficiente, ya que no requiere de entrenamiento de modelos de ML, sin embargo, son dependientes de etiquetas de estructura bien definidas, lo que los hacen menos flexible ante variaciones de formato. Un ejemplo de este enfoque es PDFminer, que extrae texto de archivos PDF.

El segundo enfoque, emplea modelos de Natural Language Processing (NLP) y ML para interpretar y estructurar el contenido del documento sin seguir las reglas rígidamente. Esto permite manejar documentos con estructuras complejas, como los artículos científicos (con más de una columna, gráficos y tablas), de manera que extrae no solo el texto, sino que emplea técnicas avanzadas para extraer las tablas y las referencias del texto. Al emplear métodos más avanzados, regularmente son más costosos computacionalmente, como GeneRation Of BIbliographic Data (GROBID), que estructura artículos científicos en un formato XML y Text Encoding Initiative (TEI).

Actualmente existen múltiples herramientas comerciales disponibles, siendo capaces de extraer la información de los documentos PDF en diferentes formatos [45]. Las herramientas más populares son las siguientes:

- **PyMuPDF**: Se trata de una librería de Python ligera y rápida que permite la extracción de texto y estructuración de documentos. Permite navegar y extraer elementos específicos de las páginas del documento a través de las diferentes clases y métodos que ofrece el paquete. Los formatos de salida pueden ser desde texto plano sin estructura (TXT) hasta HTML o XML con información jerárquica.

- **Pdfplumber**: Es una librería de Python diseñada para la extracción avanzada de datos desde archivos PDF. Se especializa en la extracción de texto, tablas e imágenes enfocado en la precisión y la estructuración. Es especialmente práctico en documentos complejos, con múltiples columnas, tablas embebidas o textos con formatos irregulares.

- **GROBID**: Es una herramienta de código abierto diseñada para la extracción y estructuración automática de información en documentos científicos. A diferencia de los modelos anteriores, emplea técnicas de NLP y ML para transformar artículos académicos en formatos estructurados como XML y TEI. Además del texto, también permite la extracción de metadatos, como los títulos, referencias o palabras clave.

- **LlamaParse**: Emplea una combinación de IA, visión computacional y técnicas de NLP para extraer información de documentos complejos. Se destaca por su capacidad de manejar documentos con objetos embebidos como tablas, proporcionando una representación en formato Markdown.

Pese a que hay herramientas significativamente eficaces en el parseo, todas presentas ciertas limitaciones que imposibilitan una única selección con eficacia absoluta. Si, además, añadimos un conjunto de documentos que no tienen la misma estructura en todos los documentos, la tarea de extracción puede hacerse aún más compleja con solo una herramienta de parseo. Por lo tanto, para optimizar dicha tarea, se debe realizar una combinación de dos herramientas que se complementen en sus características, como GROBID, que permite identificar entre columnas, pero dificulta la extracción de tablas, y LlamaParse, que permite la extracción de tablas pero tiene dificultades con la diferenciación de columnas.

Chunking, la fragmentación de documentos

El siguiente paso en el preprocesamiento de los datos es la fragmentación (chunking) que consiste en dividir el texto extraído en fragmentos (chunks) más pequeños, que pueden ser desde párrafos completos a oraciones o unidades lingüísticas más simples. Esta fragmentación permite una recuperación más precisa y específica, ya que el modelo se puede enfocaren unidades de texto más manejables y relevantes, ya que al gestionar fragmentos más pequeños, se optimiza el modelo en términos de memoria, y permite una mejor paralelización de las tareas de procesamiento [46].

Se ha demostrado que los sistemas RAG que emplean LLM suelen generar respuestas inexactas al recuperar todo un documento, ya que se incorpora información innecesaria que introduce ruido en el sistema [3]. Esto se debe a que la información innecesaria y excesiva distorsiona el resultado de dichos modelos, reduciendo la confiabilidad del sistema, sobre todo en tareas críticas como el asesoramiento clínico. En la Figura 2.3 se puede observar el rendimiento de dos modelos, detallando el efecto del chunking, demostrando la mejora del renidmiento de la recuperación y de la generación de la respuesta gracias a la fragmentación.

De hecho, los modelos RAG basados en fragmentos (chunks) como ChunkRAG han demostrado una eficiencia mucho más elevada respecto a los modelos basados en recuperación de documentos, sobre todo en documentos largos y complejos [3, 47]. Los modelos basados en chunks realizan una jerarquización de los fragmentos, de manera que puede considerar aquellos más relevantes para recuperar, dando lugar a un incremento de la precisión general del modelo.

Además de la jerarquización de los fragmentos, el sistema también será capaz de comparar fragmentos entre ellos por similitud, filtrando los resultados para evitar información repetida o redundante que dificultan la capacidad del modelo para generar repuestas coherentes y únicas. Existen múltiples enfoques que analizan la similitud entre fragmentos, como el cálculo de similitud de coseno, que calcula el ángulo entre dos vectores distintos de cero en un espacio de producto interno, cuantificando la similitud entre dos vectores independientemente de su magnitud. [48]

También existen modelos que, para optimizar la recuperación de fragmentos, emplean una búsqueda basada en caché. Este enfoque permite reducir el cálculo redundante en un 51 % en cargas de trabajo de producción reales, mejorando así la latencia y el rendimiento en tareas de inferencia [49]. El almacenamiento de caché de claves y valores permite almacenar fragmentos de información previamente procesada para su reutilización, reduciendo la redundancia computacional. Sin embargo, este sistema enfrenta limitaciones en cuanto a la sobrecarga computacional en toma de decisiones en tiempo real, además de la baja flexibilidad ante diferentes modelos de lenguaje. Esto refleja la importancia del almacenamiento de memoria, pero también implica

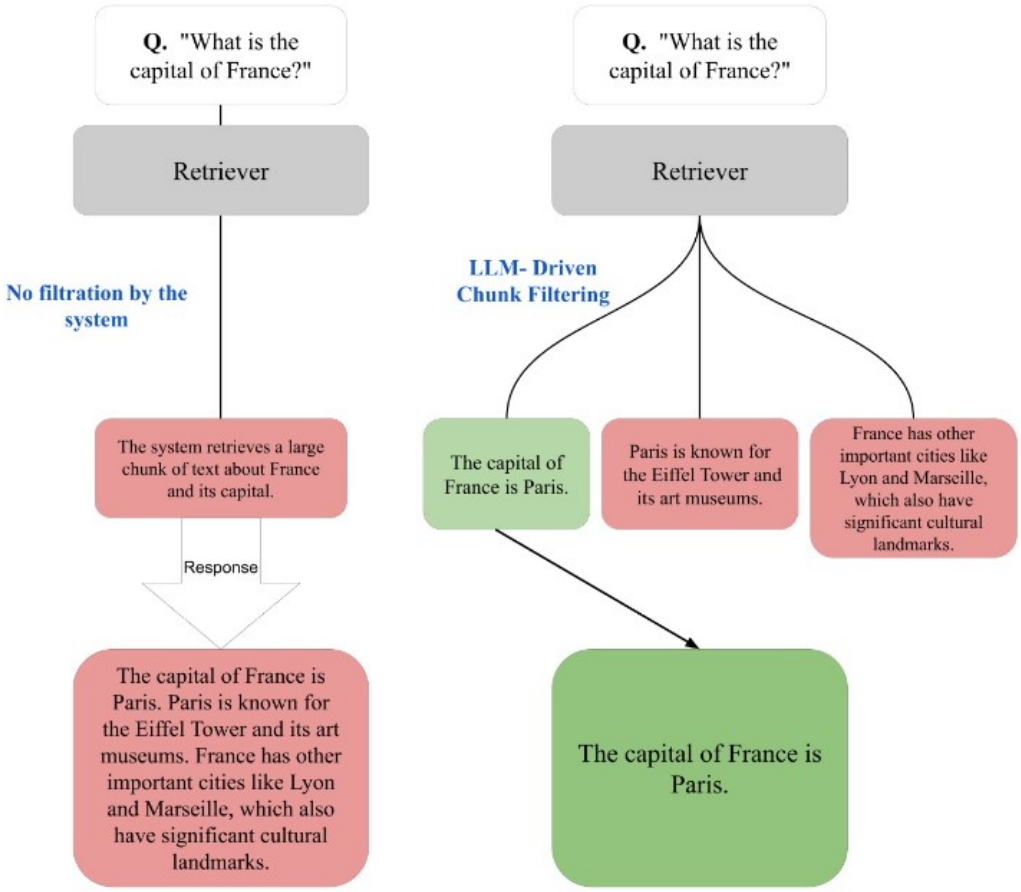


Figura 2.3: Comparación del efecto del chunking. Extraído de [3]

una limitación en el almacenamiento de caché.

Se debe tener en cuenta que optimizar el tamaño de los fragmentos es esencial para la precisión y la recuperación eficiente, sin embargo, determinar el tamaño adecuado depende de la estructura de los datos y la densidad de información e incluso, en la misma base se pueden encontrar diferentes grados de granularidad [50]. De hecho, otros factores como la misma pregunta pueden afectar al tamaño del chunk óptima, ya que en función de la granularidad de la query, el tamaño del chunk puede variar (si se solicita una información más amplia, se prefiere una granularidad más gruesa). En conclusión, se debe encontrar un equilibrio en la granularidad, ya que una granularidad gruesa proporciona más información con menor precisión, mientras que la fina ofrece información completa a costa de la eficiencia.

- **Basado en tokens**: Este enfoque divide el texto en fragmentos de tamaño especificado según los recuentos de tokens sin procesar. Permite definir una ventana de solapamiento (*overlap*) para respetar el contexto entre fragmentos [50]. Aunque es eficiente, puede cortar frases o ideas en puntos arbitrarios, afectando la coherencia y el contexto del framento. En LangChain, se denomina `TokenTextSplitter`.

- **Basado en oraciones**: Este método se centra en dividir el texto en unidades más natu-

rales, como oraciones completas, intentando mantener unidas las oraciones y los párrafos. De esta manera, se preserva mejor la coherencia y el contexto, mejorando la eficacia en la recuperación de información. Sin embargo, puede resultar menos eficiente si el texto no tiene una estructura clara o uniforme. En LangChain, se llama `SpacyTextSplitter`.

- **Recursivo**: La fragmentación recursiva divide el texto de manera iterativa hasta obtener el tamaño adecuado, intentando mantener el mayor tiempo posible párrafos completos para conservar la relación semántica. Es un enfoque bastante flexible, ya que permite establecer limitadores de tamaño y puede ajustarse a diferentes estructuras de texto, pero su efectividad depende de la estructura de los datos procesados. En LangChain, se denomina `RecursiveCharacterTextSplitter`.

En modelos RAG, la precisión de la recuperación depende en gran medida de los fragmentos de información. Por ello, es importante no solo el tamaño del chunk, si no que también es relevante que no se pierda el contexto dentro del mismo fragmento, por lo que **RecursiveCharacterTextSplitter** destaca entre el resto de técnicas.

Vectorización: embedding

Una incrustación (embedding) es una representación numérica densa de datos, generalmente en forma de un vector, en un espacio de alta dimensión. Su objetivo es transformar la información, como palabras, oraciones o conceptos en vectores que capturen las relaciones y características semánticas del contenido original [51]. . En el contexto del NLP, los embeddings convierten frases en vectores de números que permiten que el modelo entienda la similitud, el contexto y las relaciones entre los términos de manera más eficiente.

Los inicios de los modelos de embedding se remontan a los enfoques basados en frecuencia que permitían calcular la relevancia de términos en documentos, pero sin capturar las relaciones semánticas entre las palabras del contexto. Posteriormente, se introdujo la idea de representar palabras como vectores en un espacio continuo, de manera que términos con significado similar mantuvieran la similitud en las representaciones mediante distancia. La revolución llegó cuando la arquitectura Transformer permitió crear modelos como BERT y GPT, que permitieron generar embeddings dinámicos, adaptándose al contexto de cada palabra en una oración, permitiendo su incorporación en tareas de recuperación de información y en el desarrollo de arquitecturas RAG.

Una de las desventajas de los LLM es que al incorporar más contexto de entrada se genera un cuello de botella en la generación de la inferencia, ya que la ralentiza la generación además de requerir más memoria. Para abordar estas limitaciones, se han desarrollado técnicas de compresión de contexto, reduciendo la cantidad de información que el modelo necesita procesar, siendo posible emplear dos técnicas: la filtración de palabras irrelevantes del texto antes de pasarlo al modelo de LLM, y usando embeddings, que convierten el contexto en una representación más compacta, pero manteniendo gran parte del significado.

Durante el proceso, la información contextual se transformó en vectores incrustrados y almacenada en una base de datos vectorial, lo que permite representar y recuperar el contexto de manera eficiente. Del mismo modo, cuando el usuario ingresa la pregunta al modelo, se realiza un embedding de la misma pregunta empleando el mismo modelo, realizando comparaciones entre los vectores de la base de datos y obteniendo aquellos con menor distancia. Esta distancia,

codificada como contexto, implica que aquellos que se obtengan con menor distancia serán, a su vez, los más similares a nivel contexto.

Los documentos recuperados son utilizados como contexto para generar una respuesta, permitiendo al modelo generar respuestas más precisas e informativas al basar sus resultados en datos reales.

Los enfoques de compresión basados en embeddings se centran en comprimir el contexto en incrustraciones (embeddings) que el modelo decodificador del LLM puede interpretar directamente. Esta técnica permite comprimir la información contextual segmentándola en fragmentos empleando herramientas de clustering. autoencoders o mean poolintg para reducir dimensionalidad, que posteriormente se agregan en incrustaciones de resumen mediante un modelo iterativo hasta alcanzar el tamaño de incrustación objetivo.

De este modo, palabras estructuralmente diferentes como “perro” y “cachorro” serían palabras cercanas por su semántica.

Modelos como OpenAI Embedding convierten fragmentos de texto en vectores densos que pueden ser empleados para buscar y recuperar información relevante en bases de datos vectoriales. Esto permite mejorar la recuperación de documentos relevantes, de manera que el modelo recibe el contexto más relevante para generar respuestas informadas.

Los modelos de embeddings, pese a su efectividad presentan viarias limitaciones [52]:

- **Dependencia de modelos de compresión grandes:** Estos métodos se basan en modelos de compresión grandes para lograr una efectividad, incrementando los costos computacionales y dificultando su implementación en entornos con recursos limitados.
- **Baja efectividad en la generación de respuestas:** Actualmente no aprovechan completamente el potencial de los LLM, ya que optimizan solo ciertos componentes del sistema, dejando el decodificador del modelo sin modificaciones.
- **Tasa de compresión fija:** Los métodos actuales no ofrecen diferentes tasas de compresión con respecto a la longitud del contexto de entrada, impidiendo optimizar el equilibrio entre velocidad de inferencia y la calidad de generación.
- **Limitación a un solo documento:** Los métodos efectivos actuales admiten en su mayoría la generación de respuestas a partir de un único documento, restringiendo la utilidad en tareas de análisis e integración de información de múltiples fuentes, como la generación de respuestas complejas basadas en diversos documentos.

Actualmente, existen múltiples modelos de embedding para realizar el proceso de embedding. La plataforma HuggingFace (`https://huggingface.co/spaces/mteb/leaderboard`) proporciona una tabla de clasificación en Massive Text Embedding Benchmark (MTEB) Leaderboard, donde se comparan diferentes modelos de embedding en tareas del MTEB.

Esta clasificación ayuda a evaluar el rendimiento de los diferentes modelos en diferentes escenarios, permitiendo seleccionar el modelo más adecuado en función de las necesidades. La tabla proporciona las características y métricas evaluadas en diferentes contextos. En el contexto de RAG y la recuperación de información, las métricas más destacables serian:

- **Retrieval (Recuperación)**: Evalúa qué tan bien el embedding recupera información relevante.

- **Semantic Textual Similarity (STS)**: Mide qué tan bien el embedding entiende relaciones entre textos similares.
- **Re-ranking**: Ayuda a mejorar los documentos recuperados en RAG, priorizando ciertos documentos.
- **Bitext Mining**: Capacidad de identificar pares de textos equivalentes en distintos idiomas (para RAG multilingüe).

Se extrajeron las 4 mejores opciones en base a estos parámetros, teniendo en cuenta su compatibilidad con un buen rendimiento en RAG con LangChain. De este modo, se plantearon estas opciones:

- **Alibaba NLP (gte-Qwen2-7B-instruct)**: Tiene una gran recuperación y STS, además de que es open-source (gratuito). Soporta preguntas complejas en RAG, funcionando bien con retrieval y generación de contexto relevante. No tiene integración directa con LangChain.
- **Intfloat (e5-mistral-7b-instruct)**: Tiene un buen equilibrio entre Retrieval y re-ranking, y está optimizado para tareas de NLP, pero tiene un consumo mayor que otros modelos, además de una integración menos directa con LangChain.
- **Gemini (gemini-embedding-exp-03-07)**: Se destaca tanto en re-ranking, retrieval y STS, optimizado para comprensión semántica profunda y tareas de recuperación de información. Tiene acceso restringido y falta de integración optimizada para LangChain.
- **OpenAI (text-embedding-3-large)**: Tiene una buena optimización para retrieval, STS y re-ranking. Es totalmente compatible con LangChain e incluso con el vectorstore de ChromaDB. Tiene una eficiencia y costo superior a otros modelos, sin requerir de una infraestructura propia.

Entre las opciones disponibles, OpenAI destaca entre ellas al disponer de integración nativa con LangChain y ChromaDB, simplificando su implementación. Además ofrece un rendimiento óptimo en los parámetros de retrieval, STS y re-ranking, garantizando una recuperación de información eficiente. También evita la necesidad de infraestructura propia para la inferencia, optimizando tanto en costos como eficiencia. Finalmente, su escalabilidad y persistencia asegura que el sistema, en caso de ser necesario, pueda crecer con el tiempo.

El resto de alternativas, pese a tener ventajas en ciertos aspectos, presentan también limitaciones clave en integración, costos o flexibilidad.

Base de datos vectoriales: vectorstores

Para almacenar y consultar embeddings, se emplean bases de datos vectoriales (VectroStores), optimizadas para realizar búsquedas rápidas y eficientes, permitiendo recuperar los fragmentos de texto más relevantes en función de una consulta específica. Esto facilita una recuperación precisa y relevante de la información, mejorando la eficiencia en la generación de respuestas basadas en el contexto adecuado. Estas bases especializadas permiten almacenar

datos de alta dimensión que no han podido ser caracterizados por sistemas de gestión de bases de datos tradicionales. Estos vectores pueden tener una cantidad variable de dimensiones, dependiendo de la granularidad de los datos, y representan matemáticamente las características o atributos de los fragmentos de texto [53].

Estas bases de datos especializadas presentan ciertas ventajas respecto a las bases de datos tradicionales:

- **Búsqueda y recuperación por similitud**: Pueden encontrar los datos más relevantes según distancia en el espacio vectorial, siendo especialmente práctica en aplicaciones de NLP, a diferencia de las bases de datos tradicionales, que no pueden captar el significado contextual o semántico en el texto.
- **Soporte para datos complejos y no estructurados**: Permiten almacenar y buscar datos con alta complejidad y granularidad, sin necesidad de estar estructurados como en las bases de datos tradicionales.
- **Escalabilidad y rendimiento**: Pueden manejar análisis y procesamiento de datos a gran escala y en tiempo real, esencial para aplicaciones de IA. Pueden distribuir la carga de trabajo y reducir la latencia en consultas a gran escala mediante técnicas avanzadas de indexación y paralelización.

Actualmente, los tres modelos de VectorStore más populares son ChromaDB, MongoDB (Atlas Vector Search) y Facebook AI Similarity Search (FAISS), cada una con sus características y limitaciones [54, 55].

- **MongoDB** soporta búsquedas vectoriales dentro de una base de datos documental, y permite combinar consultas estructuradas y vectoriales [56].
- **FAISS** es una librería optimizada para búsqueda rápida en colecciones de vectores, soportando una indexación eficiente y búsqueda aproximada (lo que reduce la latencia en consultas grandes) [57].
- **ChromaDB** es una base de datos optimizada para búsquedas por similitud, permitiendo almacenar y consultar metadatos adicionales. Tiene una integración nativa con el framework de LangChain, y se puede emplear en entornos locales y escalables en la nube [58].

Para determinar la base de datos vectorial adecuada, se debe tener en cuenta ciertos aspectos clave: la integración con el framework utilizado (LangChain), la posibilidad de almacenar metadatos y la facilidad para implementar el modelo y escalarlo.

Respecto a la integración con LangChain, MongoDB no tiene una integración nativa con el framework (a diferencia de ChromaDB), así como FAISS, que pese a ser eficiente en búsquedas vectoriales, tampoco tiene un soporte nativo.

En cuestión de soporte para el almacenamiento y metadatos, MongoDB puede almacenar los metadatos en documentos JavaScript Object Notation (JSON) dentro de una base de datos documental, pero requiere de indexación específica para la búsqueda por similitud (algo que

ChromaDB hace de forma automática). FAISS, por otro lado, no puede almacenar los metadatos dentro de la base de datos, requiriendo un almacenaje externo.

En términos de facilidad de implementación, MongoDB requiere el uso de Atlas Vector Search, una funcionalidad de pago, requiriendo de una implementación compleja de comprender. FAISS, al ser una librería optimizada en búsquedas, no es una base de datos persistente, por lo que para mantener los datos a largo plazo requiere de un almacenamiento externo.

Los vectores incrustados, en un modelo RAG, se recuperarán mediante similitud con la pregunta del usuario. De este modo, uno de los factores clave en la selección de la base de datos vectorial será su optimización para búsquedas por similitud. Chroma, como se ha mencionado anteriormente, está diseñado específicamente para este propósito. MongoDB, pese a soportar estas búsquedas, su rendimiento es inferior ya que está diseñado como base de datos documental. FAISS, aunque es altamente eficiente en búsquedas vectoriales y está optimizado para grandes volúmenes de datos, carece de almacenamiento persistente.

Finalmente, debido a que las guías clínicas que se emplearán en este modelo pueden aumentar con el tiempo (con nuevas publicaciones, por ejemplo), el modelo seleccionado debe tener una escalabilidad y flexibilidad adecuada. MongoDB tiene una alta escalabilidad, pero depende del uso de Atlas Vector Search (de pago), limitando su uso en grandes volúmenes de datos. FAISS, pese a ser eficiente en búsquedas a gran escala, no está diseñado para ser una solución completa. ChromaDB, en cambio, se puede ejecutar tanto localmente como en la nube, con opciones de persistencia que permite escalar en función de las necesidades del modelo.

(¿tabla resumen?)

RAG

Recuperación de información: retrieval

En esta fase, los retrievers buscarán la información más relevate dentro de la base de datos proporcionada en el vectorstore, que contendrá los fragmentos de información y sus embeddings asociados. De este modo, la query realizada por el usuario sirve como input para el modelo de recuperación y buscará la información más relevante.

Cabe destacar que la consulta (query) se somete a un proceso similar al de la indexación de los documentos, vectorizandose usando el mismo modelo de embedding, por lo que el sistema puede calcular una similitud semántica entre la consulta y la información indexada. De este modo, al comparar los embeddings de la consulta con los almacenados, se determinan los fragmentos más relevantes tanto en contexto, significado y similitud de palabras.

La información recuperada será la base para los posteriores procesos, por lo que se deben tener en cuenta las limitaciones que afecten a la calidad de la respuesta generada, por lo que debemos asegurar que el LLM generador de la respuesta se integre eficientemente con el módulo recuperador.

Durante la configuración del módulo de recuperación, el parámetro k define la cantidad de documentos que se recuperarán en el proceso, y su ajuste tiene un impacto crucial en la calidad de las respuestas generadas. Recuperar demasiados documentos puede introducir información irrelevante, confundiendo al modelo, reduciendo la precisión y aumentando de forma innecesaria la carga computacional. Por el contrario, si se recuperan muy pocos documentos, el modelo podría no contar con el contexto suficiente, lo que aumenta el riesgo de generar respuestas inexactas o alucinadas. Además, una recuperación limitada podría reflejar información

sesgada, sin representar adecuadamente la diversidad del conocimiento disponible, dando lugar a respuestas parciales o incompletas.

Augmented Retrieval

Enel contexto de RAG, el proceso de Augmented Retrieal juega un papel crucial en la mejora de la calidad de las respuestas generadas. En esta fase, una vez se han recuperado los fragmentos más relevantes del conjunto de datos, se emplearán estos fragmentos para proporcionar contexto adicional a la consulta realizada.

El contexto recuperado contiene información relevante y específica relacionada con la pregunta del usuario. Este contexto se introduce como parte del prompt en un modelo de LLM. Este prompt se construye con la consulta del usuario, combinándola con los fragmentos recuperados. El LLM generará una respuesta que no solo se basará en la consulta, sino que también considera el contexto proporcionado para intentar producir una respuesta más precisa, completa y relevante.

Este enfoque permite que el Modelo de Lenguaje aproveche el conocimiento externo, superando las limitaciones del modelo de generación de respuestas a partir de conocimiento proporcionado durante el entrenamiento del modelo. La clave de la Recuperación Aumentada, es que permite adaptar el contexto recuperado para que el modelo genere respuestas más informadas y específicas a preguntas complejas, mejorando la calidad de las respuestas en comparación con respuestas generadas sin contexto.

La ingeniería de prompt juega un papel crucial en la optimización de la recuperación aumentada. Al crear prompts de manera coherente con el contexto y con fidelidad al contenido recuperado se puede incrementar la precisión numérica y relevancia de las respuestas generadas, mejorando hasta un 5

Además, la ingeniería de prompt permite mitigar debilidades en los enfoques de recuperación, como la capacidad de la búsqueda vectorial de equilibrar la semántica y la exactitud, ya que ambos son aspectos fundamentales para generar respuestas relevante y completas.

Generador de respuestas: generator

Finalmente, el LLM utilizará la información aumentada para generar una respuesta coherente y precisa. El módulo generador será el encargado de producir respuestas a partir de los documentos recuperados, sintetizando el conocimiento relevante y combinándolo con la capacidad generativa del modelo.

Dado que la generación se basa en documentos verificados, el riesgo de alucinaciones se reduce significativamente. Sin embargo, el módulo generador sigue enfrentando desafíos inherentes a la calidad de la información recuperada. La integración de fuentes diversas puede dar lugar a inconsistencias o conflictos de conocimiento, lo que requiere mecanismos adicionales para evaluar la fiabilidad y coherencia de los datos antes de generar una respuesta.

2.2.2. Tipos de RAG en la actualidad

Los LLM están diseñados principalmente para procesar los datos en formato de texto, pero hay modelos similares que trabajan con otros tipos de datos, como audio o video. Los LLM más conocidos comercialmente, como GPT-4 de OpenAI, Large Language Model Meta AI (LlaMA)

de Meta AI o Claude de Anthropic son modelos que pueden realizar tareas y generar un texto coherente, trabajando solo con entradas y salidas textuales.

Del mismo modo, los modelos de RAG se basaron inicialmente en estudios de texto, pero han evolucionado para abarcar modalidades como audio, video incluso multimodal, permitiendo aplicaciones como el reconocimiento de voz o análisis de video [59].

Por ejemplo, los modelos de audio extienden los principios al procesamiento del habla, permitiendo la transcricpión o los asisntentes de voz inteligentes. Los datos se representan mediante embeddings de modelos preentrenados y se emplean tanto para la recuperación de la información coo la generación de respuestas contextuales. A inicios de 2025, [60] evaluó audios de enseñanza simulada con un modelo RAG local, obteniendo puntuaciones relativamente altas en precisión (4.13 sobre 5.00) y lenguaje (4.37 sobre 5.00). Este modelo presentó limitaciones técnicas derivadas del tipo de dato, ya que se detectaron faltas de precisión en el reconocimiento multilingüe.

Por otro lado, también existen modelos RAG que basan su recuperación en datos extraídos de imagenes, combinando transformadores de visión con la generación aumentada para mejorar la generación de respuestas a partir de imágenes, como se comprobé en el estudio de [61], en el que se emplearon Vision Transformers y variantes para procesar imágenes médicas y extraer información visual relevante, empleando GPT-2 como decodificador. Este estudio superó los modelos recurrentes en la generación de informes médicos, demostrando el potencial de los sistemas RAG en la mejora de eficiencia y precisión en el diagnóstico.

Unificando ambos tipos de datos, se puede asumir que la aplicación de RAG en video presenta un desafío significativo debido a la naturaleza multimodal del contenido audiovisual, necesitando de recuperación eficiente de fragmentos de vídeo, pero también generando respuestas basadas en los metadats de los mismos. El estudio de [62] permite identificar las limitaciones más relevantes de un sistema RAG multimodal, siendo la más relevante que no existen métricas estandarizadas que midan la capacidad de los LLM para seleccionar segmentos de video adecuados a partir de descripciones textuales.

El estudio de [6] demostró que la calidad de los datos disponibles del sistema, mayor rendimiento muestra. Concretamente, el estudio de [6] realizó una transformación progresiva de los datos, desde una simple limpieza inicial de los datos en formato textual, a la conversión de imágenes a texto y tablas en formato tabular, obteniendo una precisión significativamente superior cuánto mayor calidad mostraba la conversión,

Se puede observar, entonces, que los modelos RAG están evolucionando para convertirse en herramientas multimodales capaces de manejar todo tipo de datos. Aunque existen desafíos técnicos y metodológicos, los avances en este campo prometen mejorar significativamente sectores como el diagnóstico médico.

2.2.3. Retos y límites de RAG, direcciones de futuro

La capacidad para combinar la recuperación de información y la generación de texto ha permitido avances significativos en tareas complejas, como la generación de informes a partir de imágenes, la creación de resumenes o la respuesta de preguntas. Sin embargo, pese a su creciente evolución y aplicabilidad, los sistemas RAG enfrentan varios retos y limitaciones técnicas, quedando potencial por explotar.

En los sistemas RAG, las respuestas incompletas o erróneas pueden deberse a diversos fac-

tores dentro de su arquitectura, y no únicamente a la falta de contenido, lo que puede resultar en respuestas sin referencias sólidas que las respalden. En la Figura 2.4 se observan los posibles fallos en la recueperación de documentos. Uno de los fallos más comunes ocurre cuando el sistema no logra identificar información relevante en los documentos recuperados, a pesar de que esta esté presente, debido a errores en la clasificación que impiden destacar dicha información como significativa. Otro posible problema surge cuando, aun habiendo detectado la documentación adecuada, el modelo de lenguaje (LLM) no la incorpora correctamente en la respuesta, afectando negativamente la calidad de la información proporcionada. En situaciones donde se manejan numerosos documentos con contenido similar, la presencia de ruido en la información recuperada puede llevar a respuestas imprecisas o confusas. Asimismo, cuando la pregunta es demasiado general, el modelo puede generar respuestas que no abordan adecuadamente la necesidad de la consulta. Finalmente, también es posible que se generen respuestas incompletas que, aunque no sean incorrectas, omitan detalles relevantes presentes en el contexto, reduciendo así la utilidad de la información ofrecida [34].

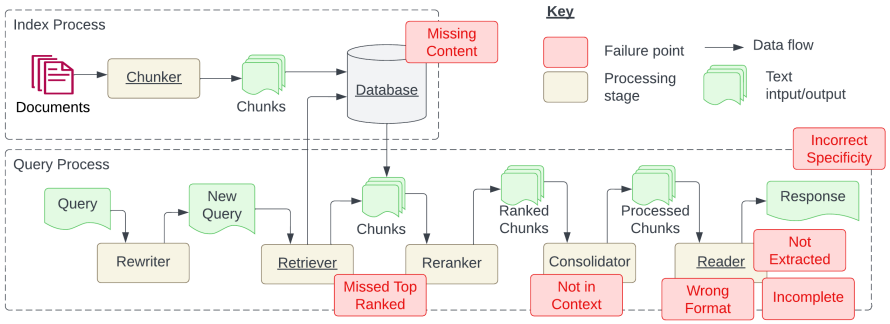


Figura 2.4: Workflow de arquitectura RAG. Extraído de [4]

A partir de estos errores, se derivan una serie de desafíos técnicos que impactan directamente en el rendimiento y la fiabilidad de los sistemas RAG:

- **Dependencia de la calidad de los documentos recuperados**: La precisión de los modelos RAG se basa, principalmente, en la calidad de los documentos recuperados, ya que en caso de incluir datos inexactos o incorrectos se puede producir una recuperación sesgada o .ªlucinaciones", en que las respuestas generadas por el modelo se basan en referencias incompletas o completamente inventadas pero tomadas como reales. Esto compromete la fiabilidad de los resultados, influenciando a su aplicación práctica en ámbitos en que la precisión es fundamental [16]. Del mismo modo, los modelos pueden "heredar"sesgos presentes en los documentos recuperados, siendo uno de los desafíos más críticos, ya que pese a que se han propuesto diferentes técnicas, como la recuperación justa, la completa eliminación del sesgo sigue siendo una tarea complicada.

- **Alta demanda computacional**: Uno de los principales retos de los modelos RAG radica en su alta demanda computacional, ya que requieren de dos módulos, la recuperación de información y la generación del texto. Solamente la latencia en la recuperación de documentos relevantes puede ser altamente costosa, afectando a la eficiencia del sistema si se manejan grandes cantidades de datos. Además, la creación de un repositorio de

los vectores, los embeddings y la generación de respuesta impactan directamente e nla capacidad del modelo RAG y la posibilidad de aplicarlo al mundo real.

- **Seguridad del sistema**: Las arquitecturas RAG presentan ciertas vulnerabilidades de seguridad, como los ataques de recuperación y los ataques generativos con el objetivo de manipular la información recuperada o la respuesta generada [63]. Aun así, estos ataques requieren de condiciones específicas que pueden evitarse, como el acceso único a bases de datos locales [64].

Capítulo 3

Estado del arte

3.1. Introducción

Las guías clínicas son herramientas fundamentales en la práctica médica, proporcionando recomendaciones basadas en evidencia para el diagnóstico y tratamiento de diversas patologías. Con el auge de la Inteligencia Artificial y el Procesamiento del Lenguaje Natural, han surgido nuevas metodologías para mejorar el acceso y recuperación de información de estas guías.

De este modo, surgieron modelos de RAG que han demostrado ser eficaces en la extracción y organización de conocimiento clínico, facilitando su uso en aplicaciones médicas y teniendo potencial de reducir el tiempo en la toma de decisiones por parte de los especialistas médicos.

3.2. Estudios y aplicaciones recientes

Un estudio reciente de febrero de 2025 realizado por Vivani et al. [65] integró bases de datos científicas, como PubMed, en un modelo de recuperación de información basado en RAG. En este enfoque, se empleó un módulo de recuperación en que se extrajeron fragmentos relevantes de PMC empleando representaciones TF-IDF y el modelo de recuperación BM25. Estos fragmentos se segmentarion y luego se transformaron en embeddings mediante el modelo BioBERT, calculando la similitud del coseno para determinar su relevancia. A continuación, los fragmentos recuperados, junto a la pregunta del usuario, se pasaron a un modelo de LLM para generar respuestas precisas, siguiendo instrucciones específicas para limitar la generación a la información previamente proporcionada por los fragmentos recuperados, mejorando la precisión fáctica a través de técnicas de prompt engineering. El estudio comparó varias configuraciones del modelo, mostrando que los modelos que incluyen BioBERT como modelo de embedding junto con modelos generativos, como GPT o LLaMA, lograron un rendimiento significativamente superior, obteniendo valores de Normalized Discounted Cumulative Gain (NDCG) notablemente más altos que las configuraciones sin el modelo generativo, destacando por su capacidad para recuperar documentos relevantes en función de la posición en la lista de resultados [66].

En el ámbito cardiovascular, Adejumo et al. [67] implementaron una arquitectura RAG con el objetivo de optimizar la evaluación del riesgo de ictus y guiar el tratamiento anticoagulante en pacientes con fibrilación auricular. Esta arquitectura, que combina la recuperación de información con el modelo de LLM Llama3.1, fue aplicada con un conjunto de 1000 notas clínicas.

Los resultados se validaron mediante la revisión manual de 200 notas por parte de médicos expertos. Al comparar los resultados del modelo RAG con los datos estructurados, se observaron diferencias significativas en la capacidad de identificar los factores de riesgo. Por ejemplo, RAG fue capaz de identificar hipertensión en un 82.4 % frente al 26.2 % de los datos estructurados. Del mismo modo, otros factores clave fueron identificados con mayor precisión utilizando el modelo RAG. Este enfoque, al proporcionar una identificación más precisa de factores de riesg, tuvo un impacto directo en la toma de decisiones, particularmente en el cálculo de la puntuación CHADS-VASc, que es crucial para evaluar el riesgo de accidente cerebrovascular en pacientes con fibrilación auricular.

En los servicios de urgencia y hospitales rurales, donde los profesionales clínicos pueden no contar con radiólogos cualificados para un análisis rápido de imágenes médicas, se presenta un desafío significativo para brindar una atención médica adecuada. Con el fin de mejorar la precisión diagnóstica, Bani-Harouni et al [68] presentaron un enfoque denominado Multi-Agent Guideline-Driven Diagnostic Assistant (MAGDA), integrando modelos de visión y lenguaje con guías clínicas de zero-shot para la clasificación de enfermedades raras a partir de imágenes médicas. Este enfoque se basa en la generación dinámica de modelos de visión-lenguaje sin necesidad de realizar ajustes finos (fine-tuning), facilitando su aplicación a enfermedades raras sin entrenamiento previo específico. Este sistema multiagente opera técnicas de razonamiento en cadena de pensamiento a través de tres componentes clave: cribado, diagnóstico y refinamiento. De este modo, se proporciona una justificación paso a paso hasta tomar la decisión diagnóstica, haciendo que el proceso sea transparente, crucial en el entorno clínico.

En medicina preoperatoria, Ke et al., 2025 [69] desarrollaron un modelo RAG basado en 35 guías clínicas preoperatorias como base de conocimiento para mejorar la precisión de las respuestas generadas por LLMs. Para ello, se emplearon herramientas como LangChain y LlamaIndex en el preprocesamiento de los documentos, fragmentándolos en chunks y almacenándolos en Pinecone como base de datos vectorial, utilizando la similitud coseno para la recuperación de información. El rendimiento se evaluó en 14 escenarios clínicos mediante la comparación de respuestas generadas por diferentes LLMs, llegando a lograr un aumento en la precisión al 91.4 % con GPT-4.0, superando el desempeño de modelos sin RAG. Además, el tiempo de respuesta se redujo de 10 minutos a 15-20 segundos, manteniendo una precisión comparable a la respuesta humana, demostrando el potencial de RAG para mejorar la eficiencia y confiabilidad de los sistemas de apoyo en el entorno médico.

Recientemente, el estudio de Ge et al. [16] desarrolló el modelo de LLM LiVersa, y evaluó la capacidad de los LLM en la interpretación de casos clínicos específicos, respondiendo preguntas sobre carcinoma hepatocelular y Virus de la Hepatitis B (VHB). Para entrenar y optimizar LiVersa se utilizaron 30 documentos de la American Association for the Study of Liver Diseases (AASLD) incorporados mediante Azure OpenAI Cognitive Search, incluyendo guías clínicas, revisiones y estudios hepatológicos. El desempeño se evaluó comparando las respuestas por el modelo y las respuestas proporcionadas por médicos especializados, analizando tanto la exactitud de respuestas sí/no como respuestas detalladas. Mientras que las respuestas de tipo sí/no se respondieron correctamente, las preguntas detalladas mostraron 3 errores de 10 preguntas en las justificaciones, caracterizadas por la falta de documentación y por sesgos contextuales en la información proporcionada. Este estudio mostró las posibles limitaciones en las arquitecturas RAG, identificando la posibilidad de sesgo en la selección de documentos o la dependencia de la calidad de los documentos para proporcionar respuestas fundamentadas.

En el estudio de Kresevic et al. [70], se analizó el rendimiento de un modelo de LLM, específicamente GPT-4 Turbo, en la recuperación y evaluación de recomendaciones clínicas sobre el Virus de la Hepatitis C (VHC) proporcionadas por la European Association for the Study of the Liver (EASL). En este estudio, se analizaron diferentes configuraciones experimentales, variando el reformateo y la estructura de los documentos y el uso de few-shot learning. Se plantearon 5 posibles configuraciones, con diferentes niveles de procesamiento de los documentos PDF, y se demostró que, a mayor nivel de procesamiento, mayor calidad mostraba la información extraída dentro de la documentación. De este modo, se observó que los procesos que incrementaban la precisión del modelo fueron la extracción de texto en formato TXT, requiriendo de una limpieza del contenido del mismo. Respecto a las imágenes y figuras, se extrajo la información contextual, mientras que en las tablas se transformaron en listas. Tras estas modificaciones, se redujeron drásticamente las alucinaciones, con un descenso de 57 a solamente 1 caso de alucinación, mientras que en las métricas cuantitativas de la similitud entre las respuestas generadas por el modelo de LLM y las respuestas proporcionadas por expertos humanos demostraron una clara mejora después de todo el preprocesado.

En el ámbito de la nefrología, Miao et al. [71] presentó un modelo de RAG enfocado en el manejo de la Enfermedades Renales Crónicas (ERC), empleando GPT-4 para recuperar información especializada en guías KDIGO 2023. Este sistema permitió respuestas más precisas, informadas con la evidencia clínica y adaptadas a la progresión ed la ERC y su tratamiento farmacológico. El estudio demostró una mejora significativa en la precisión y especificidad en las respuestas en comparación al modelo sin RAG, obteniendo respuestas detalladas y alineadas con las últimas directrices de las guías clínicas, incluyendo tratamientos avanzados, a diferencia del modelo sin RAG, que solo proporcionaba respuestas más generales. Aún así, también se demostró la dependencia a la información contextual en la generación de las respuestas, requiriendo de integrar más guías clínicas especificas para subtipos de la ERC para abordar de forma más precisa situaciones específicas. En nefrología, donde los avances son constantes, la actualización automática de información resulta clave para evitar información obsoleta, por lo que RAG permite mejorar la precisión al extraer directamente fragmentos relevantes de artículos y guías clínicas, reduciendo el riesgo a alucinaciones.

Capítulo 4

Materiales y métodos

En esta sección se detallan los procedimientos y herramientas utilizados para desarrollar el modelo de RAG aplicado a guías clínicas. Se describen los pasos empleados en la adquisición y preprocesamiento de los documentos, las técnicas de fragmentación y la conversión de los archivos en formatos estructurados. Además, se especifican los algoritmos y modelos empleados para la recuperación de la información, junto a las métricas utilizadas para evaluar su rendimiento.

El proceso metodológico se ha diseñado con el objetivo de garantizar la eficiencia y precisión en la extracción y recuperación de contenidos relevantes dentro de las guías clínicas reumatológicas.

4.1. Materiales

En esta sección se presentan los datos y herramientas empleadas en el desarrollo del modelo. Se describe el conjunto de guías clínicas utilizadas, especificando su formato, origen y características, así como las herramientas de software y librerías utilizadas para el procesamiento de los documentos y la implementación del modelo.

4.1.1. Conjuntos de datos

Para la realización de este trabajo, se ha utilizado un conjunto de datos de documentos de guías clínicas en reumatología obtenidos de la ACR sobre patologías reumatológicas. Estos documentos engloban 17 patologías reumatológicas, como artritis reumatoide o gota, presentando una gran variedad de recomendaciones repartidas en 23 documentos. Se encuentran en documentos PDF, cada una con una configuración diferente (diferente número de columnas, diferentes formatos de estructuración...).

Cada patología cuenta con un documento de recomendaciones clínicas detalladas, respaldadas por evidencia científica, junto con tablas y figuras que permiten entenderlas, en diferentes grados de complejidad.

Dado a que estas guías no siguen un formato homogéneo, requerirán de un proceso de preprocesamiento inicial previo a la incorporación en el modelo RAG, implicando una extracción, una limpieza y normalización de la información. Se puede observar en tabla siguiente la información sobre las guías clínicas empleadas en el trabajo.

****AQUI VA LA TABLA QUE NO SE COMO PONER****

4.1.2. Herramientas y tecnologías

El desarrollo de este trabajo ha requerido de diversas herramientas y tecnologías para la implementación, procesamiento y almacenamiento de datos.

Respecto al lenguaje de programación y bibliotecas, se ha empleado el lenguaje de programación **Python** debido a su versatilidad y amplio abanico de bibliotecas especializadas, como pandas para la manipulación de datos o langchain para la integración del modelo con la arquitectura RAG.

Se ha empleado el modelo de lenguaje [**GPT-4o mini**] de OpenAI como base para la recuperación aumentada de información. Se ha seleccionado este modelo por su capacidad para procesar grandes volúmenes de texto y generar respuestas coherentes en contexto de recuperación de contexto clínico, considerando también su bajo valor económico.

Para almacenar y gestionar la información extraída de los documentos se ha empleado **ChromaDB**, elegida por su eficiencia demostrada en la gestión de bases de datos vectoriales, facilitando la búsqueda semántica y la integración con modelos de lenguaje.

Para el diseño de arquitecturas RAG existen varios frameworks especializados. Entre LangChain, LlamaIndex y Haystack, todos tienen características distintas tanto en la integración de modelos de LLM, la estrategia de recuperación que siguen y el grado de modularidad. Para este trabajo, se ha seleccionado LangChain por su flexibilidad, además de la integración nativa con ChromaDB como base vectorial y OpenAI GPT como LLM y su capacidad para gestionar interacciones complejas en sistemas de recuperación de información.

Los otros dos frameworks, pese a ser buenas opciones, fueron descartadas por diferentes motivos. LlamaIndex fue descartado ya que se centra en la integración de múltiples componentes en un workflow modular, pero su enfoque en la optimización de la indexación no es el adecuado para un modelo de RAG. Respecto a Haystack, se descartó porque su enfoque en pipelines híbridos y NLP tradicionales no es necesario para el modelo RAG, basado en embeddings y LLMs, donde LangChain permite una integración más directa.

4.2. Métodos

En esta sección se detallan los procedimientos de cada etapa del desarrollo del modelo, describiendo el flujo de trabajo aplicado en las etapas de preprocesamiento y la posterior implementación del modelo de recuperación de información y generación de respuesta. Asimismo, se explicarán los criterios de evaluación empleados para analizar el rendimiento del sistema.

4.2.1. Preprocesamiento de los documentos

Tal y como se ha mencionado anteriormente, las guías clínicas de ACR no mantienen una estructura uniforme, sino que contienen elementos complejos como tablas y figuras. Por ello, el preprocesamiento de los documentos es una fase crítica en la implementación del sistema RAG, ya que garantiza una homogeneización, estructuración y segmentación de la información contenida en las guías clínicas. Principalmente, se debe convertir el formato original de los documentos (PDF) en formatos más estructurados como XML o MarkDown.

Para la extracción del contenido textual de los PDF's, se emplearon herramientas especializadas como GROBID y LlamaParse, que permitieron obtener una representación estructurada,

separando secciones, títulos, párrafos e incluso tablas.

Parte de la complejidad de las guías clínicas corresponde a la información no contextual de los documentos, como las referencias del texto o los encabezados de la página. Esta información, pese a proporcionar información sobre la procedencia del texto, no proporciona un contexto sobra la información de las recomendaciones de las guías clínicas. Para mantener la mayor claridad posible dentro del texto de los documentos, se realizaron limpiezas diferentes en cada documento considerando las capacidades de la técnica de parseo.

La herramienta LlamaParse, se ejecutó en un entorno en la nube mediante una clave API, en la que se obtuvo un documento Markdown como resultado del parseo. Esta herramienta es eficaz diferenciando y obteniendo las tablas en un formato Markdown, diferenciando entre las celdas tanto en filas como columnas. En este documento, la limpieza se basó en una eliminación de todo contenido diferente a las tablas. El formato Markdown, pese a poseer cierta estructura, no se pudo "limpiar" eficazmente mediante código, por lo que se realizó una limpieza manual, asegurando la correcta estructura de las tablas, y estableciendo una diferenciación mediante secciones entre ellas.

La herramienta de GROBID, pese a tener una alta capacidad en la diferenciación de columnas, tiene una carencia significativa en la extracción y estructuración de la información contenida en las tablas, sin poder diferenciar la información de celdas colindantes. Por ello, en los documentos XML/TEI obtenidos a partir de esta herramienta, se eliminaron todas las etiquetas asociadas a las tablas y figuras, así como la información que contenían. En los documentos XML/TEI, al tener una estructura más clara gracias a las diferentes etiquetas, se realizó una limpieza mediante código en que se realizaron las siguientes acciones:

- Eliminación de la sección de referencias bibliográficas.
- Limpieza de las referencias ubicadas en el texto (ubicadas gracias a la estructura de referenciado Vancouver)
- Eliminación de la sección de autores y reconocimientos.
- Eliminación de las tablas y figuras gracias a las etiquetas ¡table¿y ¡figure¿.

Las figuras, pese a la posibilidad de ser formateadas en una estructura tabular, no mantienen todo el contexto de la figura en ninguno de los dos formatos. Debido a que la información de las figuras es, en gran medida, la información contenida en el texto y las tablas, se determinó su eliminación en ambos formatos.

Con los documentos formateados y limpios, se empleó una fragmentación de los documentos mediante RecursiveCharacterTextSplitter. Esta herramienta de chunking permite la división del texto, empezando por secciones más grandes como párrafos, y va dividiendo en secciones más pequeñas como oraciones, y palabras, manteniendo la coherencia y evitando corar frases de manera abrupta. El método Recrusive encuentra un equilibrio en el tamaño del chunk, manteniendo el significado dentro de cada chunk sin hacerlo de un tamaño excesivo o insuficiente. Además, ajusta los chunks para que se adapten a los líomites de tokens sin perder coherencia, que será especialmente útil al emplear embeddings de texto (que tienen un límite de tokens dependiendo del modelo de embedding).

Antes del chunking, para evitar la pérdida de información, se extrajo la información de ambos documentos y se combinaron obteniendo un documento en formato TXT por cada documento PDF original.

Un parámetro clave para mantener el contexto durante el chunking es la superposición de los fragmentos (overlap) que permitirá al modelo encadenar los fragmentos. Con la información combinada, se especificaron los parámetros de fragmentación, con un tamaño de 1024 caracteres y un overlap de 150.

4.2.2. Representación y almacenamiento de la información

Con la información fragmentada, se procedió a la representación de la información contextual de cada fragmento en un formato numérico, convirtiendo los chunks a un formato vectorial de altas dimensiones, de manera que se mantuviera gran parte de la información contextual.

En esta fase, se empleó el modelo de embedding de OpenAI text-embedding-ada-002, caracterizado por su buena relación costo-calidad, destacando entre el resto de opciones consideradas anteriormente. Este modelo analiza la semántica del texto y asigna un vector que captura su significado de manera distribuida en el espacio de características. Dado que los embeddins son representaciones numéricas de alta dimensión, requieren de un sistema de almacenamiento especializado que permita realizar búsquedas por similitud. Debido a su integración nativa tanto con el modelo de embedding como con el framework, se empleó el vectorstore de Chroma (ChromaDB), una base de datos vectorial optimizada para la indexación y recuperación de embedding en entornos de recuperación aumentada de información (RAG).

ChromaDB se configura como un modo de almacenamiento persistente, de manera que los embeddings se pueden reutilizar sin necesidad de recalcularlos en cada consulta (se guardan en un directorio). Cada vector generado se vinculó con los metadatos correspondientes, incluyendo el nombre original del documento, los identificadores DOI y Pubmed y el título del documento.

Durante el almacenamiento, se optimizan los parámetros de búsqueda para garantizar que los futuros tiempos de respuestas sean óptimos y rápidos. ChromaDB permite, gracias a su configuración, realizar consultas en tiempo real basadas en métricas de distancia (como coseno o producto escalar), optimizando la precisión en la recuperación de fragmentos relevantes durante el proceso de recuperación.

4.2.3. Recuperación de información

4.2.4. Generación de respuestas

4.2.5. Evaluación de sistema

La evaluación de una arquitectura RAG es bastante compleja, ya que interfieren múltiples componentes independientes, cada uno con métricas y limitaciones propias. Por ello, se establecerán parámetros de evaluación de las métricas.

Interacción Retriever-Generator

El rendimiento del sistema depende, en mayor medida, de la calidad de la recuperación y de la generación del LLM, ya que si el retriever devuelve documentos irrelevantes, el LLM

generará respuestas incorrectas o incompletas, mientras que si la generación del LLM no sigue el contexto proporcionado por el retriever, la recuperación puede ser inútil.

Variabilidad de la respuesta

Estudio de las alucinaciones

Efecto del re-ranker

Capítulo 5

Valoració econòmica

Consideración previa:

El uso de Application Programming Interface (API)s de LLM como OpenAI requieren del uso de tokens, por lo que tendrán un coste asociado. El coste depende del modelo de LLM empleado en el modelo. Se barajaron las opciones de GPT-3.5 turbo y GPT-4o mini, buscando la optimización del

Capítulo 6

Referencias

Bibliografía

[1] Zichong Wang, Zhibo Chu, Thang Viet Doan, Shiwen Ni, Min Yang, and Wenbin Zhang. History, development, and principles of large language models-an introductory survey, 2024.

[2] Guangzhi Xiong, Qiao Jin, Zhiyong Lu, and Aidong Zhang. Benchmarking retrieval-augmented generation for medicine, 2024.

[3] Ishneet Sukhvinder Singh, Ritvik Aggarwal, Ibrahim Allahverdiyev, Muhammad Taha, Aslihan Akalin, Kevin Zhu, and Sean O'Brien. Chunkrag: Novel llm-chunk filtering method for rag systems, 2024.

[4] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, March 2023.

[5] Brandon T. Garcia, Lauren Westerfield, Priya Yelemali, Nikhita Gogate, E. Andres Rivera-Munoz, Haowei Du, Moez Dawood, Angad Jolly, James R. Lupski, and Jennifer E. Posey. Improving automated deep phenotyping through large language models using retrieval augmented generation. *medRxiv*, Dec 2024. Preprint.

[6] Simone Kresevic, Mauro Giuffre, Milos Ajcevic, Agostino Accardo, Lory S. Croce, and Dennis L. Shung. Optimization of hepatological clinical guidelines interpretation by large language models: a retrieval augmented generation-based framework. *NPJ Digital Medicine*, 7(1):102, April 23 2024.

[7] Mullai Murugan, Bo Yuan, Eric Venner, Christie M Ballantyne, Katherine M Robinson, James C Coons, Liwen Wang, Philip E Empey, and Richard A Gibbs. Empowering personalized pharmacogenomics with generative ai solutions. *Journal of the American Medical Informatics Association*, 31(6):1356–1366, May 20 2024.

[8] Paul Delanoe, Dieudonne Tchuente, and Guillaume Colin. Method and evaluations of the effective gain of artificial intelligence models for reducing co2 emissions. *Journal of Environmental Management*, 331:117261, 2023.

[9] Tomoko Yokoi. Genai: Ventaja competitiva versus coste medioambiental, Dec 2024. Accessed: 2025-02-21.

[10] Pengfei Li, Jianyi Yang, Mohammad A. Islam, and Shaolei Ren. Making ai less "thirsty": Uncovering and addressing the secret water footprint of ai models, 2025.

[11] Programa de las Naciones Unidas para el Medio Ambiente (PNUMA). La ia plantea problemas ambientales. esto es lo que el mundo puede hacer al respecto. *Programa de las Naciones Unidas para el Medio Ambiente*, September 21 2024. Reportaje Medio ambiente bajo revisión.

[12] Jad Abi-Rafeh, Nader Henry, Hong Hao Xu, Brian Bassiri-Tehrani, Adel Arezki, Roy Kazan, Mirko S Gilardino, and Foad Nahai. Utility and comparative performance of current artificial intelligence large language models as postoperative medical support chatbots in aesthetic surgery. *Aesthetic Surgery Journal*, 44(8):889–896, 02 2024.

[13] Jessica Morley, Caio C.V. Machado, Christopher Burr, Josh Cowls, Indra Joshi, Mariarosaria Taddeo, and Luciano Floridi. The ethics of ai in health care: A mapping review, 2020.

[14] Dennis Küster and Tanja Schultz. Künstliche intelligenz und ethik im gesundheitswesen - spagat oder symbiose? *Bundesgesundheitsblatt, Gesundheitsforschung, Gesundheitsschutz*, 66(2):176–183, Feb 2023.

[15] Jiwoong Sohn, Yein Park, Chanwoong Yoon, Sihyeon Park, Hyeon Hwang, Mujeen Sung, Hyunjae Kim, and Jaewoo Kang. Rationale-guided retrieval augmented generation for medical question answering, 2024.

[16] Jin Ge, Steve Sun, Joseph Owens, Victor Galvez, Oksana Gologorskaya, Jennifer C. Lai, Mark J. Pletcher, and Ki Lai. Development of a liver disease-specific large language model chat interface using retrieval-augmented generation. *Hepatology*, 80(5):1158–1168, November 2024.

[17] Chin Siang Ong, Nicholas T. Obey, Yanan Zheng, Arman Cohan, and Eric B. Schneider. Surgeryllm: a retrieval-augmented generation large language model framework for surgical decision support and workflow enhancement. *npj Digital Medicine*, 7(1):364, 2024.

[18] Chao Zhang, Hanxin Zhang, Atif Khan, Ted Kim, Olasubomi Omoleye, Oluwamayomikun Abiona, Amy Lehman, Christopher O. Olopade, Olufunmilayo I. Olopade, Pedro Lopes, and Andrey Rzhetsky. Lightweight mobile automated assistant-to-physician for global lower-resource areas, 2021.

[19] Charlene H. Chu, Rune Nyrup, Kathleen Leslie, Jiamin Shi, Andria Bianchi, Alexandra Lyn, Molly McNicholl, Shehroz Khan, Samira Rahimi, and Amanda Grenier. Digital ageism: Challenges and opportunities in artificial intelligence for older adults. *The Gerontologist*, 62(7):947–955, August 2022.

[20] Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 (general data protection regulation), 2016.

[21] Health insurance portability and accountability act of 1996 (hipaa), 1996.

[22] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya

Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[23] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[24] Fali Wang, Zhiwei Zhang, Xianren Zhang, Zongyu Wu, Tzuhao Mo, Qiuhao Lu, Wanjing Wang, Rui Li, Junjie Xu, Xianfeng Tang, Qi He, Yao Ma, Ming Huang, and Suhang Wang. A comprehensive survey of small language models in the era of large language models: Techniques, enhancements, applications, collaboration with llms, and trustworthiness, 2024.

[25] Y. Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. volume 3, pages 932–938, 01 2000.

[26] T. Mikolov, Martin Karafiát, Lukas Burget, J. Cernocký, and S. Khudanpur. Recurrent neural network based language model. *Proceedings of Interspeech*, 2, 01 2010.

[27] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55, January 2025.

[28] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers, 2023.

[29] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022.

[30] Srinivasan Iyer, Xi Victoria Lin, Ramakanth Pasunuru, Todor Mihaylov, Daniel Simig, Ping Yu, Kurt Shuster, Tianlu Wang, Qing Liu, Punit Singh Koura, Xian Li, Brian O'Horo, Gabriel Pereyra, Jeff Wang, Christopher Dewan, Asli Celikyilmaz, Luke Zettlemoyer, and Ves Stoyanov. Opt-iml: Scaling language model instruction meta learning through the lens of generalization, 2023.

[31] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022.

[32] Yash Saxena, Sarthak Chopra, and Arunendra Mani Tripathi. Evaluating consistency and reasoning capabilities of large language models, 2024.

[33] Andrew Shin and Kunitake Kaneko. Large language models lack understanding of character composition of words, 2024.

[34] Scott Barnett, Stefanus Kurniawan, Srikanth Thudumu, Zach Brannelly, and Mohamed Abdelrazek. Seven failure points when engineering a retrieval augmented generation system, 2024.

[35] Tilmann Bruckhaus. Rag does not work for enterprises, 2024.

[36] Yi Dong, Ronghui Mu, Gaojie Jin, Yi Qi, Jinwei Hu, Xingyu Zhao, Jie Meng, Wenjie Ruan, and Xiaowei Huang. Building guardrails for large language models, 2024.

[37] Jinwei Hu, Yi Dong, and Xiaowei Huang. Trust-oriented adaptive guardrails for large language models, 2025.

[38] S. Farquhar, J. Kossen, L. Kuhn, et al. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630:625–630, 2024.

[39] Jad Abi-Rafeh, Hong Hao Xu, Roy Kazan, Ruth Tevlin, and Heather Furnas. Large language models and artificial intelligence: A primer for plastic surgeons on the demonstrated and potential applications, promises, and limitations of chatgpt. *Aesthetic Surgery Journal*, 44(3):329–343, Feb 15 2024.

[40] Jin Ge, Steve Sun, Joseph Owens, Victor Galvez, Oksana Gologorskaya, Jennifer C Lai, Mark J Pletcher, and Ki Lai. Development of a liver disease-specific large language model chat interface using retrieval augmented generation. *medRxiv*, Nov 2023. Preprint.

[41] Y. Wang, S. Leutner, M. Ingrisch, C. Klein, L. C. Hinske, and K. Danhauser. Optimizing data extraction: Harnessing rag and llms for german medical documents, August 2024.

[42] LangChain. Langchain documentation, 2025. Último acceso: 20 de marzo de 2025.

[43] LlamaIndex. Llamaindex documentation, 2025. Último acceso: 20 de marzo de 2025.

[44] deepset.ai. Haystack documentation, 2025. Último acceso: 20 de marzo de 2025.

[45] Arnau Perez and Xavier Vizcaino. Advanced ingestion process powered by llm parsing for rag system, 2024.

[46] Antonio Jimeno Yepes, Yao You, Jan Milczek, Sebastian Laverde, and Renyu Li. Financial report chunking for effective retrieval augmented generation, 2024.

[47] In Fei y Li Wenjie y Navigli Roberto Zong, Chengqing y Xia, editor, *Hallazgos de la Asociación de Lingüística Computacional: ACL-IJCNLP 2021*.

[48] Lun-Chi Chen, Mayuresh Sunil Pardeshi, Yi-Xiang Liao, and Kai-Chih Pai. Application of retrieval-augmented generation for interactive industrial knowledge management via a large language model. *Computer Standards Interfaces*, 94:103995, 2025.

[49] Shubham Agarwal, Sai Sundaresan, Subrata Mitra, Debabrata Mahapatra, Archit Gupta, Rounak Sharma, Nirmal Joshua Kapu, Tong Yu, and Shiv Saini. Cache-craft: Managing chunk-caches for efficient retrieval-augmented generation, 2025.

[50] Zijie Zhong, Hanwen Liu, Xiaoya Cui, Xiaofan Zhang, and Zengchang Qin. Mix-of-granularity: Optimize the chunking granularity for retrieval-augmented generation, 2025.

[51] Phu-Vinh Nguyen, Minh-Nam Tran, Long Nguyen, and Dien Dinh. Advancing vietnamese information retrieval with learning objective and benchmark, 2025.

[52] David Rau, Shuai Wang, Hervé Déjean, and Stéphane Clinchant. Context embeddings for efficient answer generation in rag, 2024.

[53] Yikun Han, Chunjiang Liu, and Pengfei Wang. A comprehensive survey on vector database: Storage and retrieval technique, challenge, 2023.

[54] Emir Öztürk and Altan Mesut. Performance analysis of chroma, qdrant, and faiss databases. *UNITECH – Selected Papers*, 2024.

[55] MongoDB Inc. What are vector databases?, 2024. Accessed: March 23, 2025.

[56] MongoDB. MongoDB Atlas Vector Search, 2025. Último acceso: 23 de marzo de 2025.

[57] Facebook AI Research. FAISS: A Library for Efficient Similarity Search, 2025. Último acceso: 23 de marzo de 2025.

[58] ChromaDB. ChromaDB Documentation: Introduction, 2025. Último acceso: 23 de marzo de 2025.

[59] Shailja Gupta, Rajesh Ranjan, and Surya Narayan Singh. A comprehensive survey of retrieval-augmented generation (rag): Evolution, current landscape and future directions, 2024.

[60] Ke Fang, Ci Tang, and Jing Wang. Evaluating simulated teaching audio for teacher trainees using rag and local llms. *Scientific Reports*, 15(1):3633, January 29 2025.

[61] S. Raminedi, S. Shridevi, and D. Won. Multi-modal transformer architecture for medical image analysis and automated report generation. *Scientific Reports*, 14(1):19281, August 2024.

[62] Yannis Tevissen, Khalil Guetari, and Frédéric Petitpont. Towards retrieval augmented generation over large video libraries, 2024.

[63] Cheng Su, Jinbo Wen, Jiawen Kang, Yonghua Wang, Yuanjia Su, Hudan Pan, Zishao Zhong, and M. Shamim Hossain. Hybrid rag-empowered multi-modal llm for secure data management in internet of medical things: A diffusion-based contract approach, 2024.

[64] Jiaqi Xue, Mengxin Zheng, Yebowen Hu, Fei Liu, Xun Chen, and Qian Lou. Badrag: Identifying vulnerabilities in retrieval augmented generation of large language models, 2024.

[65] Rishabh Uapadhyay and Marco Viviani. Enhancing health information retrieval with rag by prioritizing topical relevance and factual accuracy, 2025.

[66] Olivier Jeunen, Ivan Potapov, and Aleksei Ustimenko. On (normalised) discounted cumulative gain as an off-policy evaluation metric for top-n recommendation, 2024.

[67] P. Adejumo, P. Thangaraj, S. V. Shankar, L. S. Dhingra, A. Aminorroaya, and R. Khera. Retrieval-augmented generation for extracting chads-vasc risk factors from unstructured clinical notes in patients with atrial fibrillation. *medRxiv [Preprint]*, Sep 2024.

[68] David Bani-Harouni, Nassir Navab, and Matthias Keicher. Magda: Multi-agent guideline-driven diagnostic assistance, 2024.

[69] YuHe Ke, Liyuan Jin, Kabilan Elangovan, Hairil Rizal Abdullah, Nan Liu, Alex Tiong Heng Sia, Chai Rick Soh, Joshua Yi Min Tung, Jasmine Chiat Ling Ong, and Daniel Shu Wei Ting. Development and testing of retrieval augmented generation in large language models – a case study report, 2024.

[70] Mauro Giuffre, Simone Kresevic, Nicola Pugliese, Kisung You, and Dennis L. Shung. Optimizing large language models in digestive disease: strategies and challenges to improve clinical outcomes. *Liver International*, 44(9):2114–2124, September 2024. Epub 2024 May 31.

[71] J. Miao, C. Thongprayoon, S. Suppadungsuk, O. A. Garcia Valencia, and W. Cheungpasitporn. Integrating retrieval-augmented generation with large language models in nephrology: Advancing practical applications. *Medicina (Kaunas)*, 60(3):445, March 8 2024.

.1. Anexo 1. Preguntas PEC2

El grado de cumplimiento de los objetivos ha sido adecuado. Durante el desarrollo del PEC2, se han consolidado los conocimientos sobre los modelos de LLM y la arquitectura RAG, logrando completar las tareas establecidas previamente en el cronograma de Gantt. Se han explorado diferentes enfoques para el preprocesamiento de los documentos de las guías clínicas, incluyendo el análisis de los metadatos, diferentes técnicas de chunking y estrategias de vectorización, embedding y almacenamiento de vectrostore. Además, se han evaluado distintos modelos en entornos clínicos, permitiendo comparar el rendimiento de diferentes retrievers y generators con el objetivo de seleccionar la mejor arquitectura RAG para el caso de estudio. Las principales actividades realizadas durante el período del PEC2 han incluido:

- Exploración de metadatos procedentes de las guías clínicas obtenidos durante el proceso de parseo.
- Implementación y ajuste de estrategias de chunking para la fragmentación eficiente de los documentos clínicos.
- Aplicación de técnicas de vectorización y almacenamiento de datos en la base de datos vectorial.

- Validación preliminar de la calidad del vectorstore mediante métricas que evalúan la recuperación y evaluación de los embeddings empleados.
- Selección de modelos de lenguaje para el sistema RAG y análisis de sus capacidades en entornos clínicos.
- Selección de la arquitectura más adecuada, basada en experimentos y evaluaciones realizadas.

Como parte del desarrollo del PEC2, se ha creado un repositorio de GitHub que centraliza el código, los experimentos y la documentación del proyecto. De este modo, el repositorio facilita la trazabilidad del trabajo y permite futuras mejoras. El repositorio consta de un notebook con el código empleado, así como el repositorio de la guías clínicas empleadas, los documentos parseados y limpiados, los vectores y el vectorstore.
Repositorio de GitHub: https://github.com/dcamacmon/DCM$_A RQUITECTURA-RAG/$

.2. Anexo 2. Tabla de metadatos

Name	Year	DOI	PubMed
2019 Update of the ACR/SAA/SRTN Recommendations for the Treatment of Ankylosing Spondylitis and Nonradiographic Axial Spondyloarthritis	2019	10.1002/art.41042	https://pubmed.ncbi.nlm.nih.gov/31436026/
2022 ACR Guideline for the Prevention and Treatment of Glucocorticoid-Induced Osteoporosis	2022	10.1002/art.42646	https://pubmed.ncbi.nlm.nih.gov/37845798/
2020 ACR Guideline for the Management of Gout	2020	10.1002/acr.24180	https://pubmed.ncbi.nlm.nih.gov/32391934/
2023 ACR Guideline for Exercise, Rehabilitation, Diet, and Additional Integrative Interventions for Rheumatoid Arthritis	2023	10.1002/acr.25117	https://pubmed.ncbi.nlm.nih.gov/37227116/
2023 ACR/CHEST Guideline for the Screening and Monitoring of Interstitial Lung Disease in Systemic Autoimmune Rheumatic Diseases	2023	10.1002/art.42860	https://pubmed.ncbi.nlm.nih.gov/38973714/
2024 ACR Guideline for the Screening, Treatment, and Management of Lupus Nephritis	2024	10.1002/acr.21664	https://pubmed.ncbi.nlm.nih.gov/22556106/
2020 ACR Guideline for the Management of Reproductive Health in Rheumatic and Musculoskeletal Diseases	2020	10.1002/acr.24130	https://pubmed.ncbi.nlm.nih.gov/32090466/

Name	Year	DOI	PubMed
2021 ACR Guideline for the Treatment of Rheumatoid Arthritis	2021	10.1002/art.41752	https://pubmed.ncbi.nlm.nih.gov/34101376/
2023 ACR Guideline for Vaccinations in Patients With Rheumatic and Musculoskeletal Diseases	2023	10.1002/acr.25045	https://pubmed.ncbi.nlm.nih.gov/36597813/
2021 ACR/Vasculitis Foundation Guideline for the Management of Antineutrophil Cytoplasmic Antibody–Associated Vasculitis	2021	10.1002/art.41773	https://pubmed.ncbi.nlm.nih.gov/34235894/
2019 ACR/Arthritis Foundation Guideline for the Management of Osteoarthritis of the Hand, Hip, and Knee	2019	10.1002/art.41142	https://pubmed.ncbi.nlm.nih.gov/31908163/
2022 ACR/AAHKS Guideline for the Perioperative Management of Antirheumatic Medication in Patients Undergoing Elective Total Hip or Knee Arthroplasty	2022	10.1016/j.arth.2022.05.043	https://pubmed.ncbi.nlm.nih.gov/35732511/
2015 Recommendations for the Management of Polymyalgia Rheumatica	2015	10.1002/art.39333	https://pubmed.ncbi.nlm.nih.gov/26352874/
2018 ACR/National Psoriasis Foundation Guideline for the Treatment of Psoriatic Arthritis	2018	10.1002/art.40726	https://pubmed.ncbi.nlm.nih.gov/30499246/

Name	Year	DOI	PubMed
2023 ACR/AAHKS Clinical Practice Guideline for the Optimal Timing of Elective Total Hip or Knee Arthroplasty for Patients with Symptomatic Moderate to Severe Osteoarthritis or Osteonecrosis	2023	10.1002/acr.25175	https://pubmed.ncbi.nlm.nih.gov/37743767/
2022 ACR Guideline for Vaccinations in Patients with Rheumatic and Musculoskeletal Diseases	2023	10.1002/art.42386	https://pubmed.ncbi.nlm.nih.gov/36597810/
Clinical Practice Guidelines by IDSA, AAN, and ACR: 2020 Guidelines for the Prevention, Diagnosis, and Treatment of Lyme Disease	2020	10.1002/art.41562	https://pubmed.ncbi.nlm.nih.gov/33251716/
2021 ACR Guideline for the Treatment of Juvenile Idiopathic Arthritis	2021	10.1002/acr.24853	https://pubmed.ncbi.nlm.nih.gov/35233986/
2023 ACR/AAHKS Guideline for the Perioperative Management of Antirheumatic Medication in Patients Undergoing Elective Hip or Knee Arthroplasty	2023	10.1002/acr.25175	https://pubmed.ncbi.nlm.nih.gov/37743767/
2019 ACR Guideline for the Management of Osteoarthritis	2019	10.1002/art.41142	https://pubmed.ncbi.nlm.nih.gov/31908163/
2023 ACR Guideline for the Management of Systemic Lupus Erythematosus	2023	10.1002/acr.25117	https://pubmed.ncbi.nlm.nih.gov/37227116/

Name	Year	DOI	PubMed
2023 ACR Guideline for the Management of Antiphospholipid Syndrome	2023	10.1002/acr.25117	https://pubmed.ncbi.nlm.nih.gov/37227116/
2023 ACR Guideline for the Treatment of Sjögren's Syndrome	2023	10.1002/acr.25117	https://pubmed.ncbi.nlm.nih.gov/37227116/