



UNIVERSITAT OBERTA DE CATALUNYA (UOC)
MÁSTER UNIVERSITARIO EN CIENCIA DE DATOS (*Data Science*)

TRABAJO FINAL DE MÁSTER

ÁREA: NATURAL LANGUAGE PROCESSING AND VISUAL ANALYTICS
DATA MINING, GRAPHS AND NATURAL LANGUAGE PROCESSING

IA Generativa para la recuperación de información de convocatorias de ayudas a empresas

Autor: José Luis Rodríguez Andreu

Tutor: Diego Calvo Barreno

Profesor: Josep Anton Mir Tutusaus

Barcelona, 16 de mayo de 2025



Esta obra está sujeta a una licencia de Reconocimiento - NoComercial - SinObraDerivada
3.0 España de Creative Commons.

FICHA DEL TRABAJO FINAL

Título del trabajo:	IA Generativa para la recuperación de información de convocatorias de ayudas a empresas
Nombre del autor:	José Luis Rodríguez Andreu
Nombre del colaborador/a docente:	Diego Calvo Barreno
Nombre del PRA:	Josep Anton Mir Tutusaus
Fecha de entrega (mm/aaaa):	06/2025
Titulación o programa:	Máster Universitario en Ciencia de Datos
Área del Trabajo Final:	Trabajo Fin de Máster
Idioma del trabajo:	Español
Palabras clave	LLM, RAG, AI

Dedicatoria/Cita

Breves palabras de dedicatoria y/o una cita.

Agradecimientos

Si se considera oportuno, mencionar a las personas, empresas o instituciones que hayan contribuido en la realización de este proyecto.

Abstract

In recent years, it has become increasingly difficult to find calls for financial aid from governmental institutions focused on companies and organizations. This growing need makes it essential to have systems that optimize the identification of calls for financial aid.

Currently, the lack of automated tools capable of interpreting and synthesizing available information hinders efficient access to these resources, forcing organizations to conduct manual searches that consume both time and effort.

This work presents the development of an Artificial Intelligence (AI)-based tool for extracting and retrieving information from economic aid calls.

By leveraging advanced Natural Language Processing (NLP) techniques and Generative AI, the solution can analyze, structure, and filter information automatically, providing relevant results based on the specific characteristics of each entity.

The main objective of the tool is to process and transform scattered aid calls into a structured dataset, facilitating their consultation and retrieval. This organized structure will allow companies to quickly and efficiently access the most relevant information, enhancing strategic decision-making.

In this way, the project addresses the challenge of filtering and synthesizing large volumes of unstructured and dispersed data from various platforms, streamlining the search process and improving access to funding opportunities.

Keywords: LLMs, IA, RAG

Resumen

En los últimos años cada vez es mas complicado encontrar convocatorias de ayuda económica por parte de instituciones gubernamentales enfocadas a empresas y entidades. Esta creciente necesidad hace imprescindible contar con sistemas que optimicen la identificación de convocatorias de ayudas económicas.

Actualmente, la ausencia de herramientas automatizadas que interpreten y sintetizen la información disponible dificulta el acceso eficiente a estos recursos, obligando a las organizaciones a realizar búsquedas manuales que consumen tiempo y recursos. Este trabajo presenta el desarrollo de una herramienta basada en Inteligencia Artificial (IA) para la extracción y recuperación de información de convocatorias de ayudas económicas.

Utilizando técnicas avanzadas de Procesamiento de Lenguaje Natural (NLP) e IA Generativa, la solución permite analizar, estructurar y filtrar la información de manera automatizada, proporcionando resultados relevantes en función de las características específicas de cada entidad.

El objetivo principal de la herramienta es procesar y convertir las convocatorias de ayudas dispersas en un conjunto de datos estructurados, lo que facilita su consulta y recuperación. Esta estructura organizada permitirá a las empresas acceder de forma rápida y eficiente a la información más relevante, mejorando la toma de decisiones estratégicas.

De este modo, se aborda el desafío de filtrar y sintetizar grandes volúmenes de datos no estructurados y dispersos de diversas plataformas, simplificando el proceso de búsqueda y optimizando el acceso a oportunidades de financiamiento.

Palabras clave: LLMs, IA, RAG

Índice general

Abstract	IX
Resumen	XI
Índice	XIII
Lista de Figuras	XVII
Lista de Tablas	1
1. Introducción	3
1. Descripción general del problema	3
2. Explicación de la motivación personal	4
3. Definición de los objetivos	5
3.1. Objetivo Principal	5
3.2. Objetivos Específicos	5
4. Descripción de la metodología empleada en el desarrollo del proyecto	6
5. Planificación o plan de investigación del proyecto	8
2. Estado del Arte	9
1. Introducción	9
2. Problemática a resolver	9
3. Soluciones disponibles	10
3.1. Plataformas de convocatorias	10
3.2. Web Scraping	11
3.3. Procesamiento de Lenguaje Natural	14
4. Inteligencia Artificial Generativa	19
5. Grandes Modelos del Lenguaje	21
5.1. Primeros modelos del lenguaje	22
5.2. ChatGPT	22

5.3.	LLaMA	22
5.4.	GPT-4	23
5.5.	DeepSeek	23
5.6.	LLaMA 2	24
5.7.	Mistral	25
5.8.	LLaMA 3	25
5.9.	DeepSeek V2	26
5.10.	GPT-4o	27
5.11.	GPT-4o mini	28
5.12.	Claude 3.5 Sonnet	29
5.13.	DeppSeek V3	30
6.	Prompt Engineering	31
6.1.	Zero-Shot Prompting	31
6.2.	Few-Shot Prompting	32
6.3.	Chain-of-Thought Prompting	32
6.4.	Instruction Following	32
6.5.	Limitaciones de Promt Engineering	33
7.	Retrieval Augmented Generation (RAG)	33
7.1.	Clasificación del RAG	35
7.2.	Técnicas de mejora de RAG	36
8.	Conclusiones	44
3.	Métodos y recursos	45
1.	Introducción	45
1.1.	Descripción general del capítulo	45
1.2.	Importancia del diseño, desarrollo e implementación en la investigación	46
2.	Objetivos y Competencias	46
2.1.	Objetivos específicos de esta fase del proyecto	46
2.2.	Competencias técnicas y metodológicas requeridas	47
3.	Diseño del sistema	48
3.1.	Arquitectura general	48
3.2.	Flujo de trabajo del sistema	50
4.	Tecnologías y herramientas	50
4.1.	Tecnologías seleccionadas	50
4.2.	Implementación del módulo de extracción de datos	53
4.3.	Implementación del módulo de procesamiento de datos	55
4.4.	Implementación del módulo de construcción de las bases de datos	57

4.5.	Implementación del RAG multiagente	58
4.6.	Implementación de la interfaz gráfica	61
5.	Resultados	62
4.	Conclusiones y trabajo futuro	63
1.	Conclusiones y trabajo futuro	63
1.1.	Resumen del trabajo realizado	63
1.2.	Lecciones aprendidas	64
1.3.	Mejoras y trabajo futuro	64
	Bibliografía	66

Índice de figuras

1.1. Timeline de tareas	8
2.1. Arquitectura Transformer	20
2.2. Flujo del prompt engineering	31
2.3. flujo de RAG	34
2.4. Tipos de RAG	37
2.5. Flujo de búsqueda híbrida	38
2.6. Esquema de Reranking	39
2.7. Contextual retrieval propuesto por Anthropic	41
2.8. Flujo de Agentic RAG	42
3.1. Matriz de ayudas del CDTI	55
3.2. Interfaz gráfica con Streamlit	62

Índice de cuadros

Capítulo 1

Introducción

1. Descripción general del problema

En la actualidad, abundan las ofertas de financiación y apoyo económico promovidas por organismos tanto públicos como privados. Sin embargo, las organizaciones empresariales encuentran dificultades para determinar qué oportunidades realmente se ajustan a su perfil específico. El gran caudal de datos y su heterogeneidad crean un panorama confuso, agravado por la carencia de sistemas automatizados que faciliten una búsqueda eficiente.

Entre las dificultades no solo se encuentra el hallazgo de convocatorias apropiadas, sino también entender la documentación requerida, los cronogramas de presentación, y determinar si estas ayudas son realmente aplicables al contexto empresarial particular. Adicionalmente, las compañías deben gestionar información fragmentada y frecuentemente desorganizada distribuida en múltiples fuentes digitales, lo que incrementa la complejidad del proceso de filtrado y selección.

Este proyecto plantea una solución que agilice, simplifique y optimice este proceso de búsqueda de financiación. Mediante la combinación de tecnologías como Inteligencia Artificial Generativa [60], Procesamiento de Lenguaje Natural (NLP) [30] y métodos de extracción de datos web o scraping [43], es posible extraer información precisa, relevante y estructurada sobre la documentación de estas convocatorias.

La importancia de esta solución se encuentra en su potencial para reducir tiempos y aumentar la efectividad al elegir opciones de financiamiento adecuadas. El hecho de implementar una solución como esta influirá positivamente en las tasas de éxito de las solicitudes presentadas y en la consecución de recursos económicos. El propósito de esta solución es democratizar el acceso a oportunidades financieras, fomentando condiciones más favorables para el desarrollo y la continuidad de las empresas.

2. Explicación de la motivación personal

La motivación tras este proyecto nace del interés personal sobre el uso de las nuevas tecnologías, especialmente la Inteligencia Artificial, en aspectos de la vida, tanto personales como profesionales, donde pueden suponer un cambio importante en la forma de realizar ciertas tareas: Agilizando procesos, reduciendo la dificultad en algunos casos y, en resumen, facilitar y hacer más accesible ciertos aspectos de la vida personal y profesional que pueden resultar tediosos.

El uso de tecnologías como el Procesamiento de Lenguaje Natural y la Inteligencia Artificial Generativa están cambiando desde hace unos años nuestra tecnología a un ritmo nunca visto. Prácticamente cada pocos meses aparecen nuevas tecnologías basadas en este campo de conocimiento, pasando por nuevos Grandes Modelos del Lenguaje como GPT-4o o DeepSeek, nuevas herramientas de desarrollo como Langchain, LLamaIndex, u Ollama, e incluso aplicaciones basadas en IA como Cursor, NotebookLM, o diferentes aplicaciones que te permiten generar texto, imágenes o música sin ser un experto.

Todas estas tecnologías están cambiando la forma en la que vemos el mundo, y aunque es necesario cierto control y regulación para no acabar en unos años en una sociedad distópica digna de la ciencia ficción, sí que considero que tenemos que aprovechar el potencial de estas tecnologías para seguir el camino hacia una sociedad más justa, equitativa, y donde tenga más peso la calidad de nuestras vidas y los derechos sociales, que las obligaciones económicas y laborales que marcan nuestro día a día.

En este caso concreto del proyecto, el uso de estas técnicas permite democratizar y hacer más accesible este tipo de ayudas. Crear una empresa y mantenerla a flote no es fácil, y en muchos casos sólo unas pocas sobreviven más de unos pocos años tras su creación, generalmente porque parten de unas capacidades económicas por detrás que no disponen el resto. Este tipo de ayudas económicas permiten a empresas con menos recursos de partida salir adelante, y herramientas como estas facilitan la búsqueda y su participación en éstas.

3. Definición de los objetivos

3.1. Objetivo Principal

El objetivo principal de este proyecto es el desarrollo de una solución automática basada en Inteligencia Artificial Generativa, que permita la indexación de información a partir de unas fuentes de datos concretas, en este caso convocatorias de ayudas a empresas, y realice tareas de extracción y estructuración de la información. De esta forma, se generará un barrido de todas las posibles convocatorias y se generará una base de datos con información relevante para su consulta y explotación.

3.2. Objetivos Específicos

- **Implementación de una herramienta de extracción de información:**

En primer lugar se diseñará una herramienta que sea capaz de identificar las diferentes convocatorias de ayudas a partir de las fuentes disponibles y extraer la información necesaria:

- Código fuente de la página web de convocatorias.
- Ficha técnica de las convocatorias.
- Documentos asociados.

Esta herramienta será una combinación de soluciones basadas en Inteligencia Artificial Generativa y Web Scraping.

- **Sistema NLP de extracción y procesamiento de información:**

Una vez extraída la información de la convocatoria, se emplearán diferentes técnicas de Procesamiento de Lenguaje Natural para diferentes tareas de procesamiento de texto y extracción de información. Este sistema empleará Grandes Modelos del Lenguaje (LLMs) en combinación con diferentes frameworks de orquestación, como Langchain o LLamaIndex. Se emplearán diferentes propuestas de LLMs para evaluar su eficacia en la extracción de información.

- **Herramienta de consulta de información:**

Finalmente, se implementará una solución de consulta y recuperación de información sobre las diferentes convocatorias, partiendo tanto de los datos estructurados generados en el paso anterior como de las fuentes originales de datos. Esta solución se basará en un RAG multiagente, empleando técnicas avanzadas en cuanto a Question Answering y procesamiento de texto.

4. Descripción de la metodología empleada en el desarrollo del proyecto

En este proyecto, se ha optado por implementar la metodología Agile debido a su enfoque iterativo y flexible, lo que nos permitirá adaptarnos rápidamente a cambios en los requisitos y mejorar continuamente el resultado a través de entregas incrementales. Agile fomenta la colaboración, la comunicación y la retroalimentación continua, asegurando que el desarrollo se mantenga alineado con las necesidades del proyecto. Además, esta metodología promueve la eficiencia y la optimización del tiempo, reduciendo riesgos y mejorando la calidad del resultado final.

■ Estrategia de investigación

La estrategia de investigación sigue el enfoque propuesto por Oates en su libro *Researching Information Systems and Computing* [47], combinando técnicas de análisis de datos cualitativos y cuantitativos para asegurar una comprensión holística del dominio. Durante el proyecto, se aplicarán estrategias presentadas en el enfoque anterior para la obtención de datos de las convocatorias. Las principales tecnologías empleadas en el desarrollo serán Python como lenguaje de programación, frameworks como Langchain, LLamaIndex o HuggingFace, y diferentes librerías de NLP y Web Scraping, así como diferentes Grandes Modelos del Lenguaje, algunos explotados desde su propia API, y otros desde orquestadores locales, como Ollama o LMStudio.

■ Fases del desarrollo

El proyecto se dividirá en diferentes fases, cada una de las cuales se centrará en una tarea específica.

- **Fase de investigación:** Revisión y análisis de las fuentes de datos disponibles, que en este caso son las diferentes webs proporcionadas de convocatorias de ayudas. A partir de los datos disponibles en éstas, se podrán definir los requisitos y funcionalidades que tiene que tener el sistema en cuanto a extracción y procesamiento de información.
- **Desarrollo de la herramienta de identificación y extracción de convocatorias:** En esta fase se desarrollará la herramienta de extracción de datos de convocatorias. Esta herramienta empleará una combinación de web scraping e IA Generativa para acceder a los sitios web de las ayudas, identificar las diferentes convocatorias y extraer las fuentes de datos en formato textual.

- **Desarrollo del sistema de procesamiento:** Una vez extraída la información, se desarrollará un sistema de procesamiento basado en IA Generativa que permita la identificación y extracción de información relevante de las convocatorias. Por un lado, extraerá datos en formato de texto web, que será procesado para ser accesible mediante Grandes Modelos del Lenguaje. Por otro lado, también procesará los ficheros PDF generalmente asociados a estas convocatorias. Como resultado, se generará una base de datos con información estructurada, y una base de datos vectorial que almacenará los embeddings de los documentos.
- **Desarrollo del sistema de consulta de información:** A partir de las bases de datos generadas en el paso anterior, se construirá un sistema basado en RAG que permita acceder a esas fuentes y realizar consultas sobre los datos. Esas consultas permitirán extraer información estructurada en el formato solicitado.

5. Planificación o plan de investigación del proyecto

El plan de desarrollo de este proyecto va marcado por las diferentes etapas que establece la metodología de la UOC. En cada una de esos bloques de trabajo se abordarán las diferentes etapas indicadas del proyecto:

- **19/02/2025 al 09/03/2025:** Definición del TFM: Enunciado y entrega (M1). Definición de los requisitos del proyecto, análisis de fuentes de datos y tecnologías disponibles.
- **10/03/2025 al 30/03/2025:** Estado del Arte: Enunciado y entrega de la actividad (M2). En este bloque de trabajo se redactará el capítulo del Estado del Arte, en base a un trabajo de investigación donde se recopilarán las herramientas y tecnologías con potencial de ser empleadas en el proyecto. Este capítulo principalmente recopilará los últimos avances en Inteligencia Artificial Generativa, Grandes Modelos del Lenguaje, y frameworks asociados. Paralelamente comenzará el desarrollo de la herramienta de identificación y extracción de convocatorias.
- **31/03/2025 al 04/05/2025:** Implementación: Enunciado y entrega de la actividad (M3). Implementación de los diferentes bloques ya definidos: Herramienta de identificación de convocatorias, Sistema de procesamiento y la solución de consulta de información.
- **05/05/2025 al 18/05/2025:** Redacción de la memoria: Entrega preliminar (M4).
- **19/05/2025 al 25/05/2025:** Redacción de la memoria: Entrega final (M4).
- **26/05/2025 al 03/06/2025:** Presentación audiovisual del trabajo (M4).
- **04/06/2025 al 06/06/2025:** Entrega de la documentación al tribunal (M5).
- **07/06/2025 al 27/06/2025:** Defensa pública del trabajo (M5).

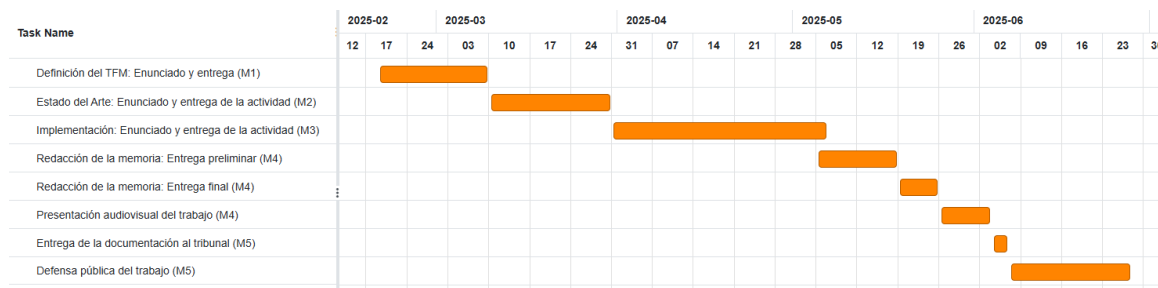


Figura 1.1: Timeline de tareas

Capítulo 2

Estado del Arte

1. Introducción

El objetivo de este capítulo es realizar un análisis de los diferentes avances, desarrollos y tecnologías disponibles en el ámbito de la solución planteada.

Este análisis tiene como objetivo identificar enfoques y metodologías en distintas áreas, como la extracción de información a partir de fuentes web, el análisis y procesado de texto y el uso de técnicas de Inteligencia Artificial aplicadas al Procesamiento de Lenguaje Natural.

De esta forma se puede establecer un contexto para el problema a resolver y justificar la elección de las tecnologías y metodologías a utilizar en el desarrollo de la solución propuesta.

2. Problemática a resolver

La búsqueda de ayudas y subvenciones es una tarea que la mayoría de las empresas, sobre todo las que tienen menos recursos, realizan en su día a día. Para ello, existen diferentes plataformas de ayudas a empresas, algunas nacionales y otras de carácter internacional. Sin embargo, esta tarea puede resultar complicada y tediosa, ya que implica una búsqueda constante de nuevas posibilidades de financiación a través de distintas fuentes. Además, la información sobre estas convocatorias suele estar distribuida en diferentes fuentes, desde las propias plataformas a documentación oficial del estado. Esto supone que a la hora de realizar una búsqueda de posibles convocatorias de financiación, se acabe con un conjunto de fuentes con diferentes estructuras y formatos.

En la mayoría de los casos, las convocatorias suelen tener asociados diferentes documentos, en su mayoría en formato PDF, los cuales pueden ser extensos, y además usan un lenguaje técnico, típico de este tipo de documentos, que dificulta su comprensión. Esto al final supone una complicación por parte de las empresas a la hora de acceder a información clave de las

convocatorias de forma mas rápida, como requisitos, plazos, presupuesto o condiciones de participación. Estos problemas de accesibilidad y estandarización de las convocatorias de ayudas suponen una barrera de acceso importante, que reduce las oportunidades de acceso a financiación para algunas empresas, y suponen una inversión en tiempo y esfuerzo en la tarea de búsqueda y filtrado por parte de éstas.

El desarrollo planteado en este proyecto pretende ser una solución a esta problemática, proporcionando una herramienta que sea capaz de identificar y extraer la documentación de las convocatorias, y aplicar técnicas de Inteligencia Artificial para extraer la información clave y dotarla de una estructura mas estandarizada, así como permitir la consulta de esta información de forma sencilla a partir de un agente conversacional.

3. Soluciones disponibles

Dejando a parte de momento las soluciones basadas en Inteligencia Artificial, las cuales se comentarán en secciones posteriores, existen diferentes metodologías para la búsqueda de información sobre estas convocatorias:

3.1. Plataformas de convocatorias

Existen diferentes plataformas que recopilan información sobre convocatorias de ayudas y subvenciones, a las cuales las empresas pueden acceder para explorar las diferentes opciones de convocatorias, y valorar si se ajustan a su situación. Estos portales de convocatorias suelen estar disponibles en plataformas tanto gubernamentales como privadas, que recopilan y organizan la información sobre diferentes ayudas disponibles. Además, estas herramientas suelen permitir aplicar filtros en las búsquedas por diferentes características, como el área geográfica, el perfil de la empresa solicitante, o el tipo de ayuda.

- **Portales de ayudas gubernamentales:**

Las diferentes instituciones públicas suelen ofrecer portales informativos donde publican este tipo de convocatorias, ya sean a nivel local, regional o nacional. Estos portales permiten visualizar estas convocatorias, pero a un nivel básico en cuanto a experiencia de usuario, y aunque la totalidad de la información siempre está disponible, ya sea en el propio portal o mediante enlaces a diferentes fuentes documentales, el análisis y búsqueda de información clave es tediosa y lenta.

- CDTI: Centro para el Desarrollo Tecnológico y la Innovación[2].
- Grupo SPRI[4].

- SODERCAN: Sociedad para el desarrollo regional de Cantabria[7].
 - Portal de ayudas del Ministerio para la Transformación Digital y de la Función Pública[6].
 - Andalucía Trade: Incentivos para Desarrollo Industrial y Proyectos de I+D+i Empresarial[1].
- **Plataformas privadas de información sobre subvenciones:**
- Algunas empresas recopilan información sobre ayudas a empresas, las estructuran en bases de datos y ofrecen el acceso a esta información como servicio, garantizando en éste la calidad de la información y una actualización constante del listado de ayudas disponibles. El inconveniente de estas plataformas es que, pese a ofrecer servicios de búsqueda que suelen tener interfaces mas amigables e información mas directa, suelen ser herramientas de pago, y el acceso completo a la información puede suponer un coste económico adicional. Algunos ejemplos de estas plataformas son Fandit [3] u OpenGrants [5].

3.2. Web Scraping

Una solución alternativa a la búsqueda manual en portales de ayudas es el uso de herramientas de Web Scraping. El Web Scraping [38] es una técnica utilizada para extraer información de sitios web de manera automatizada. Consiste en el uso de programas o scripts que navegan por páginas web, recopilan datos estructurados y los almacenan en un formato más accesible, como bases de datos o archivos locales JSON o CSV, por ejemplo. Esta práctica es ampliamente utilizada en diversos sectores para la recopilación y análisis de información a gran escala.

Generalmente el proceso de Web Scraping se desarrolla en varias etapas:

- **Solicitud HTTP:** La herramienta de scraping envía una solicitud HTTP a una página web para obtener su contenido.
- **Extracción de datos:** Se analiza el código fuente de la página, y según la configuración establecida en el scraper, se extraen los datos requeridos mediante comandos de parseo propios de la herramienta, expresiones regulares, o bots de navegación automatizada.
- **Almacenamiento de la información:** Una vez obtenidos los datos, estos se pueden formatear y almacenar según convenga en el caso de uso.

Existen diferentes metodologías de Web Scraping:

- **Análisis HTML:** En múltiples sitios web, se generan automáticamente grandes volúmenes de páginas a partir de fuentes de datos estructuradas, como bases de datos, mediante

scripts o plantillas que organizan la información en formatos homogéneos. En minería de datos, un wrapper es un programa que identifica plantillas en una fuente de datos, extrae su contenido y lo transforma en una estructura relacional. La inducción de wrappers asume que las páginas de entrada siguen un patrón identificable, usualmente a través de formatos de URL comunes. Además, lenguajes de consulta para datos semiestructurados, como XQuery y HTQL, permiten analizar, extraer y modificar información en sitios web HTML [10].

- **Análisis DOM:** Los programas pueden acceder a contenido dinámico generado por scripts del lado del cliente mediante la integración de un navegador web, como Internet Explorer o Mozilla browser control [12]. Estas aplicaciones analizan las páginas web y las estructuran en un árbol del Document Object Model (DOM), lo que permite extraer secciones específicas del contenido. El modelo DOM organiza una página web en una estructura arbórea, permitiendo su interpretación y almacenamiento a partir de una dirección web especificada, como ocurre en los motores de búsqueda. Este enfoque ofrece gran flexibilidad y agilidad, ya que permite rastrear elementos presentes en la página sin depender de que el equipo de desarrollo web los exponga explícitamente en la capa de datos.
- **HTML DOM (Hyper Text Markup Language Document Object Model):** Es un estándar para la obtención, manipulación y modificación de elementos HTML [24]. Define objetos y propiedades para cada componente HTML, así como métodos para acceder a ellos, optimizando la eficiencia del DOM. JavaScript, como lenguaje principal, permite acceder y manipular todos los elementos de un documento HTML a través del DOM. En este modelo, cada elemento HTML se trata como un objeto, cuya interfaz de programación está compuesta por métodos y propiedades específicas.
- **Expresiones regulares (Regex):** Las expresiones regulares son fórmulas que definen patrones específicos para identificar conjuntos de caracteres en diversas cadenas de texto [44]. Se componen de caracteres ordinarios y metacaracteres, los cuales modifican la interpretación del patrón. Aunque su sintaxis puede parecer compleja, las expresiones regulares son una herramienta esencial para el análisis y procesamiento de datos en cadenas de texto, por lo que es fundamental comprenderlas al menos a nivel básico.
- **XPath:** XPath es el componente principal del estándar XSLT (Stylesheet Language Transformation) y se utiliza para navegar y seleccionar elementos y atributos dentro de documentos XML [11]. Además, puede aplicarse en documentos HTML. XPath funciona como un lenguaje de selección de nodos en estructuras XML, siendo la expresión

más utilizada la ruta de ubicación (location path). Esta ruta emplea al menos un paso de ubicación para identificar un conjunto de nodos dentro de un documento. La forma más simple es la selección del nodo raíz del documento, representada por el símbolo `/`, que también es el indicador del directorio raíz en sistemas de archivos Unix.

- **Reconocimiento de anotaciones semánticas:** Las páginas extraídas pueden incluir metadatos, marcas semánticas y anotaciones que permiten identificar datos específicos [38]. Por ejemplo, esta técnica puede considerarse un caso particular del análisis DOM si las anotaciones están integradas en las páginas, como ocurre con Microformat. En otro caso, las anotaciones se almacenan y gestionan de manera independiente de las páginas web, organizándose en una capa semántica, de modo que los scrapers pueden obtener el esquema e instrucciones desde esta capa antes de realizar el raspado de las páginas.

En el ámbito del Web Scraping, específicamente con el lenguaje de Programación Python, podemos encontrar las siguientes librerías:

- **BeautifulSoup:** BeautifulSoup es una biblioteca de Python diseñada para el análisis, extracción y manipulación de datos en documentos HTML y XML. Su funcionamiento se basa en la creación de un árbol de análisis sintáctico (parse tree), que estructura el contenido de la página web de manera jerárquica, permitiendo navegar por los nodos, buscar elementos específicos y modificar el contenido. BeautifulSoup admite múltiples analizadores (parsers), como `lxml`, `html.parser` y `html5lib`, cada uno con diferentes niveles de velocidad y compatibilidad. Su sintaxis flexible permite localizar elementos a través de etiquetas, atributos y selectores CSS, facilitando la extracción de datos estructurados de páginas web. Además, cuenta con métodos para limpiar el contenido, eliminar etiquetas HTML y exportar la información en diversos formatos [40].
- **Scrapy:** Scrapy es un framework de Python diseñado para la extracción estructurada de datos mediante web scraping y crawling. Su arquitectura modular permite gestionar solicitudes HTTP, procesar respuestas y almacenar datos de manera eficiente. Scrapy opera a través de un flujo de trabajo basado en spiders, que son clases definidas por el usuario encargadas de especificar la lógica de extracción [12].

El motor de Scrapy (Scrapy Engine) coordina los componentes principales:

- Scheduler, que organiza las solicitudes pendientes.
- Downloader, que ejecuta las peticiones HTTP y recibe las respuestas.
- Spiders, que analizan y extraen información relevante.

- Item Pipeline, que transforma, valida y almacena los datos obtenidos en formatos como JSON, CSV o bases de datos SQL y NoSQL.

Además, Scrapy admite el uso de middlewares, tanto en el Downloader como en el Spider, para modificar solicitudes y respuestas, gestionar sesiones y evitar bloqueos mediante técnicas como rotación de proxies y user agents. Su diseño asincrónico optimiza el rendimiento, permitiendo la extracción masiva de datos con alta eficiencia.

- **Selenium:** Selenium es un framework de automatización de navegadores de código abierto utilizado para la ejecución de pruebas y la extracción de datos mediante web scraping [42]. Su funcionamiento se basa en la interacción con páginas web a través de un WebDriver, que actúa como un controlador para manipular elementos de la interfaz de usuario, simular clics, completar formularios y desplazarse por el contenido dinámico generado mediante JavaScript. El ecosistema de Selenium está compuesto por varios módulos:

- Selenium WebDriver, que permite la automatización de navegadores como Chrome, Firefox y Edge mediante controladores específicos.
- Selenium Grid, que posibilita la ejecución distribuida de pruebas y scraping en múltiples máquinas.
- Selenium IDE, una extensión que facilita la grabación y reproducción de secuencias de prueba en navegadores.

Para realizar web scraping con Selenium, se inicia una sesión de navegador con el WebDriver, se navega a la URL objetivo, y se localizan los elementos deseados mediante selectores XPath o CSS. A diferencia de frameworks como Scrapy o BeautifulSoup, Selenium es ideal para interactuar con sitios que requieren ejecución de JavaScript o carga dinámica de contenido, aunque su rendimiento puede ser inferior debido a la sobrecarga computacional del manejo de un navegador real.

3.3. Procesamiento de Lenguaje Natural

El Procesamiento de Lenguaje Natural (NLP) es una disciplina de la inteligencia artificial y la lingüística computacional que permite a las máquinas interpretar, comprender, generar y manipular el lenguaje humano de manera estructurada [31]. Su aplicación abarca desde la traducción automática y el análisis de sentimientos hasta la generación de texto y los asistentes virtuales.

El Procesamiento de Lenguaje Natural emplea diversas técnicas computacionales para la interpretación y manipulación del lenguaje humano. Estas técnicas pueden dividirse en métodos

estadísticos, basados en reglas y de aprendizaje profundo, siendo ampliamente utilizadas en tareas como la clasificación de textos, el análisis de sentimientos y la traducción automática.

■ Preprocesamiento de Texto

- **Tokenización:** La tokenización es un proceso esencial en el Procesamiento de Lenguaje Natural que divide un texto en unidades denominadas tokens. Existen varios tipos [58], siendo los principales: tokenización por palabras, que separa el texto en palabras individuales y es efectiva en idiomas con delimitadores claros como el inglés; tokenización por subpalabras, utilizada en modelos como BERT y GPT, que emplea técnicas como Byte Pair Encoding (BPE) para manejar vocabularios extensos y palabras fuera de vocabulario (OOV); tokenización por oraciones, que segmenta el texto en unidades sintácticas más grandes basándose en puntuación y reglas lingüísticas; y tokenización por caracteres, útil en modelos de aprendizaje profundo y generación de texto. La selección del método adecuado depende del idioma y la tarea específica, siendo crucial en aplicaciones como traducción automática, análisis de sentimientos y recuperación de información.
- **Lematización y Stemming:** Son técnicas empleadas para normalizar palabras reduciéndolas a su raíz morfológica [32]. El stemming elimina afijos mediante reglas predefinidas, sin considerar el contexto, lo que lo hace rápido pero propenso a errores (ejemplo: "running" → "run", pero "better" → "bet"). En contraste, la lematización utiliza análisis morfológico y diccionarios lingüísticos para obtener la forma base correcta (ejemplo: "better" → "good"), ofreciendo mayor precisión. Mientras que el stemming es más eficiente en grandes volúmenes de datos, la lematización es preferida en tareas como análisis de sentimientos, recuperación de información y traducción automática, donde la precisión semántica es crucial.
- **Eliminación de stopwords:** Las stopwords son palabras de alta frecuencia en un idioma que generalmente no aportan significado relevante en tareas de Procesamiento de Lenguaje Natural, como artículos, preposiciones y pronombres (ejemplo: "el", "de", "y", "pero"). Se eliminan para reducir la dimensionalidad del texto y mejorar la eficiencia de los modelos de análisis de texto [57]. Bibliotecas como NLTK, SpaCy y Scikit-learn incluyen listas de stopwords predefinidas, aunque pueden personalizarse según la aplicación. Si bien su eliminación es útil en tareas como búsqueda de información y clasificación de textos, en ciertos casos, como en análisis de sentimientos o generación de texto, pueden ser necesarias para preservar el contexto semántico.

■ Análisis Morfosintáctico y Semántico

- **Etiquetado gramatical (POS Tagging):** Es una técnica del Procesamiento de Lenguaje Natural que asigna a cada palabra de un texto su categoría gramatical correspondiente (sustantivo, verbo, adjetivo, etc.) en función de su contexto [33]. Se basa en reglas lingüísticas, modelos estadísticos o redes neuronales para mejorar la precisión del análisis. Herramientas como NLTK, SpaCy y Stanford NLP utilizan algoritmos como Hidden Markov Models (HMM) o Redes Neuronales Recurrentes (RNNs) para realizar esta tarea. El POS Tagging es clave en aplicaciones como análisis de sentimientos, desambiguación semántica y traducción automática, ya que ayuda a comprender la estructura y significado del lenguaje.
- **Parsing sintáctico:** El parsing sintáctico o análisis sintáctico consiste en la construcción de la estructura jerárquica de una oración para comprender su sintaxis, identificando la relación jerárquica entre las palabras mediante árboles sintácticos o dependencias gramaticales [75]. Existen dos enfoques principales: el parsing basado en constituyentes, que descompone la oración en frases (sintagmas nominales, verbales, etc.), y el parsing basado en dependencias, que representa las relaciones entre palabras mediante un grafo dirigido. Herramientas como NLTK, SpaCy y Stanford Parser emplean algoritmos como CYK, Earley o modelos de redes neuronales para esta tarea. El parsing sintáctico es esencial en aplicaciones como traducción automática, generación de texto y comprensión del lenguaje natural, donde la estructura de la oración influye en su interpretación.
- **Reconocimiento de Entidades Nombradas (NER):** Es una técnica que tiene como objetivo identificar y clasificar entidades dentro de un texto, como nombres de personas, lugares, organizaciones, fechas, entre otros [54]. Utiliza enfoques basados en reglas lingüísticas, modelos estadísticos o aprendizaje automático para detectar estas entidades en el contexto del texto. Herramientas como SpaCy, NLTK y Stanford NER emplean modelos entrenados en grandes corpus de datos para reconocer entidades y asignarles una etiqueta adecuada. El NER es crucial en aplicaciones como extracción de información, análisis de noticias, y búsqueda semántica, ya que facilita la identificación y categorización de información relevante dentro de grandes volúmenes de datos.

■ Representación de Texto

- **Bag of Words (BoW):** Es una técnica de representación de texto que convierte éste en una matriz de características, donde cada documento se representa como un conjunto de palabras sin tener en cuenta el orden o la gramática [50]. En el modelo BoW, cada palabra única en el corpus se convierte en una característica

(o columna) y cada documento se representa como un vector en el que el valor de cada entrada corresponde a la frecuencia de la palabra en ese documento. Aunque es simple y eficiente, BoW presenta varios inconvenientes, como la pérdida de contexto y el orden de las palabras, la alta dimensionalidad, la falta de captura de relaciones semánticas y la presencia de ruido en los datos, lo que puede afectar la precisión y la interpretación del modelo. Esta técnica, sin embargo, sigue siendo útil en tareas como clasificación de texto, análisis de sentimientos y recuperación de información, pero sus limitaciones han llevado al desarrollo de enfoques más avanzados.

- **TF-IDF (Term Frequency - Inverse Document Frequency):** Método que pondera términos relevantes dentro de un corpus [56]. Es una técnica de representación de texto que asigna un peso a cada término de un documento, basándose en dos componentes: la frecuencia de término (TF), que mide cuántas veces aparece un término en un documento, y la frecuencia inversa de documento (IDF), que mide la importancia de un término dentro de un conjunto de documentos. El cálculo de TF es sencillo y se basa en la cantidad de veces que un término aparece en un documento en comparación con el total de términos del documento, mientras que IDF ajusta el peso del término según su frecuencia en todos los documentos, penalizando las palabras comunes que aparecen en muchos documentos. El resultado es un valor que refleja la relevancia de un término en un documento en particular dentro de un corpus. TF-IDF es ampliamente utilizado en tareas como clasificación de texto, búsqueda de información y análisis de contenido, ya que ayuda a identificar términos significativos que son relevantes en el contexto de un conjunto de documentos. Sin embargo, también presenta algunas limitaciones, como la incapacidad para capturar relaciones semánticas entre palabras y su dependencia de la estructura del corpus.
- **Word Embeddings:** Técnicas avanzadas de representación de palabras que convierten las palabras en vectores numéricos de alta dimensión [8]. A diferencia de las representaciones tradicionales como Bag of Words o TF-IDF, que tratan las palabras de manera independiente, los word embeddings capturan las relaciones semánticas y contextuales entre palabras, representando palabras similares en espacios vectoriales cercanos. Modelos como Word2Vec, GloVe, FastText o los posteriores modelos de Embeddings asociados a Grandes Modelos del Lenguaje aprenden estas representaciones mediante redes neuronales entrenadas sobre grandes corpus de texto, aprovechando el contexto de las palabras en las oraciones para generar sus vectores. Los word embeddings permiten capturar propiedades lingüísticas, como sinónimos, analogías y jerarquías semánticas, mejorando el rendimiento en tareas como traducción automática, análisis de sentimientos, clasificación de texto y respuestas

automáticas. Aunque potentes, los embeddings también presentan desafíos, como la dificultad para representar términos poco frecuentes o palabras con múltiples significados (polisemia).

■ Modelos de Aprendizaje Automático y Profundo

- **Modelos basados en aprendizaje automático clásico:** Utilización de algoritmos tanto supervisados como no supervisados, como regresiones logísticas, árboles de decisión, SVM, k-means o Random Forest, entre otros, para tareas de clasificación o clusterización de texto [26]. Estos modelos están diseñados para trabajar con datos numéricos, por lo que es necesario aplicar técnicas de representación numérica de texto como las ya anteriormente comentadas.
- **Redes Neuronales Recurrentes (RNN) y LSTM:** Modelos diseñados para procesar secuencias de texto y capturar dependencias contextuales. Son arquitecturas de Deep Learning diseñadas para procesar datos secuenciales, lo que las hace especialmente útiles en Procesamiento de Lenguaje Natural. A diferencia de los modelos tradicionales de Machine Learning, las RNN pueden capturar dependencias temporales en el texto, ya que su estructura permite que la información de estados previos influya en la interpretación de los siguientes [59]. Sin embargo, las RNN convencionales presentan problemas con secuencias largas debido a la desaparición o explosión del gradiente. Para solucionar esto, surgieron las LSTM, que incorporan puertas de memoria que regulan el flujo de información, permitiendo recordar dependencias a largo plazo de manera más eficiente [62]. Estas redes han sido ampliamente utilizadas en tareas como traducción automática, generación de texto, análisis de sentimientos y reconocimiento de voz. Aunque las LSTM han mejorado el manejo de secuencias largas, han sido en gran parte reemplazadas por arquitecturas más avanzadas como Transformers, que manejan contexto de manera más eficiente mediante mecanismos de atención.
- **Arquitecturas de Deep Learning basadas en Transformers:** Estas nuevas arquitecturas revolucionaron el Procesamiento de Lenguaje Natural al superar las limitaciones de las arquitecturas de Redes Neuronales anteriores, gracias a su capacidad para procesar secuencias en paralelo y capturar relaciones a largo plazo mediante el mecanismo de atención. Introducidos en el paper "Attention Is All You Need" [68], los Transformers utilizan mecanismos de self-attention para asignar pesos a cada palabra en función de su relevancia dentro del contexto, lo que permite una comprensión más profunda del significado del texto. Modelos como BERT (Bidirectional Encoder Representations from Transformers [19]), GPT (Generative Pre-trained Transformer

[73]) y T5 (Text-to-Text Transfer Transformer [53]) lograron en su momento avances significativos en tareas como traducción automática, generación de texto, análisis de sentimientos y respuesta a preguntas. Estas arquitecturas destacan por su capacidad de preentrenamiento en grandes corpus de datos y posterior ajuste fino en tareas específicas, lo que ha permitido obtener resultados de vanguardia en múltiples aplicaciones de PLN. Tomando como base estas arquitecturas, han aparecido en los últimos años los Grandes Modelos del Lenguaje (LLMs), dando origen al desarrollo de la llamada Inteligencia Artificial Generativa.

Estas técnicas han permitido avances en aplicaciones como asistentes virtuales, generación de texto automatizada y motores de búsqueda, optimizando la interacción entre humanos y sistemas computacionales.

4. Inteligencia Artificial Generativa

La inteligencia artificial generativa (IA generativa) es un campo de la inteligencia artificial que se centra en la creación de contenido nuevo y original a partir de datos existentes. A través del uso de modelos de aprendizaje profundo y técnicas avanzadas de procesamiento de datos, la IA generativa es capaz de generar texto, imágenes, audio, video y otros tipos de contenido de manera autónoma y coherente. El desarrollo de la IA generativa ha sido impulsado por la evolución de arquitecturas de redes neuronales como las redes generativas adversarias (GANs) [22], los modelos basados en transformers [68] y los modelos de difusión [72]. Estas tecnologías han permitido la generación de contenido con una calidad cada vez mayor, logrando resultados que pueden ser indistinguibles de los creados por humanos.

Su impacto ha sido significativo en múltiples sectores. En la industria creativa, ha transformado el diseño gráfico, la producción audiovisual y la generación de música, facilitando la automatización de tareas y ampliando las posibilidades de expresión artística [65]. En el ámbito de la ciencia y la tecnología, se ha utilizado para la síntesis de datos, el descubrimiento de fármacos y la optimización de algoritmos [20]. En el sector empresarial, la IA generativa ha revolucionado el marketing, la generación de contenido automatizado y la personalización de experiencias para los usuarios [63].

A medida que la tecnología continúa evolucionando, la IA generativa se enfrenta a desafíos éticos y regulatorios [25], incluyendo cuestiones sobre derechos de autor, desinformación y uso responsable. Sin embargo, su potencial sigue expandiéndose, con nuevos avances que permiten aplicaciones aún más sofisticadas y accesibles para diversos campos.

Los modelos de IA Generativa emplean diferentes técnicas de aprendizaje profundo para generar contenido, optimizando la calidad, coherencia y realismo de los resultados. Las archi-

estructuras de redes neuronales basadas en transformers, introducidos en el artículo "Attention Is All You Need" (Vaswani et al., 2017) [68] son la base de muchos de los modelos de IA generativa actuales, incluyendo los modelos de la familia GPT, Gemini de Google y Claude de Anthropic. Su principal ventaja radica en su capacidad para procesar y generar texto, imágenes, código y otros tipos de datos de manera eficiente y con una gran coherencia contextual.

Los transformers emplean un mecanismo llamado self-attention, que les permite asignar pesos a diferentes partes de la secuencia de entrada para comprender mejor las relaciones entre palabras o elementos. el flujo de esta arquitectura funciona de la siguiente manera:

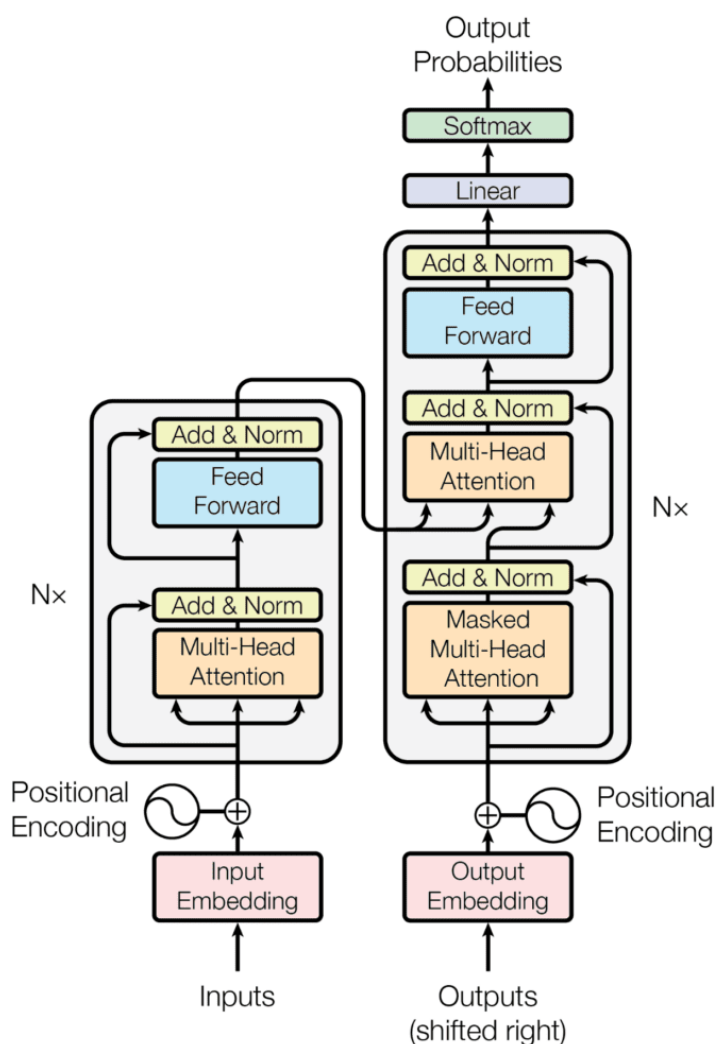


Figura 2.1: Arquitectura Transformer

- **Codificación de entrada (Embedding Layer):** Convierte los datos de entrada (por ejemplo, texto o imágenes) en vectores numéricos que representan cada elemento en un

espacio multidimensional. En el caso del texto, se utiliza un embedding de palabras, que transforma cada token en un vector denso de alta dimensión.

- **Mecanismo de Self-Attention:** Permite que el modelo asigne importancia a diferentes palabras dentro de una oración o documento. Se basa en tres matrices: Query (Q), Key (K) y Value (V), que permiten calcular la relevancia de cada palabra con respecto a las demás. Se utiliza una función de Softmax para asignar pesos a cada conexión, ajustando la atención que se le da a diferentes partes del contexto.
- **Capas de Normalización y Feed-Forward:** Cada capa de atención es seguida por una red neuronal completamente conectada (feed-forward network) y una normalización de los datos para mejorar la estabilidad del entrenamiento.
- **Mecanismo de Positional Encoding:** Dado que los transformers no tienen memoria secuencial como las redes recurrentes, utilizan codificaciones posicionales para mantener el orden de los elementos en la secuencia. Estas codificaciones se agregan a los embeddings de entrada y permiten que el modelo distinga la estructura del texto.
- **Capa de Decodificación (para modelos generativos):** En los modelos generativos, la salida del transformer se pasa a una capa de decodificación que genera una secuencia de salida basada en los tokens previos. Utiliza un enfoque autoregresivo, donde se genera un token a la vez y se retroalimenta al modelo para predecir el siguiente.

La inteligencia artificial generativa abarca diversas aplicaciones, como la creación de imágenes, texto, audio, video y modelos 3D. Sin embargo, debido a la naturaleza de este proyecto, nos centraremos principalmente en los grandes modelos de lenguaje, que son capaces de generar texto coherente y contextual a partir de indicaciones, transformando la interacción con la tecnología y facilitando la creación de contenido escrito de manera automática.

5. Grandes Modelos del Lenguaje

Los Grandes Modelos del Language (LLMs) han generado gran atención debido a su sólido rendimiento en una amplia gama de tareas de lenguaje natural desde el lanzamiento de ChatGPT en noviembre de 2022. La capacidad de comprensión y generación de lenguaje de propósito general de los LLMs se adquiere mediante el entrenamiento de miles de millones de parámetros del modelo con grandes volúmenes de datos textuales. Aunque el área de investigación de los LLMs es muy reciente, está evolucionando rápidamente.

5.1. Primeros modelos del lenguaje

Con el auge de los Transformers, empezaron a surgir nuevos modelos cada vez mas complejos, incluyendo cada vez un volumen de parámetros mayor. BERT en 2018 marcó un punto de inflexión al introducir un preentrenamiento bidireccional basado en masked language modeling (MLM), mejorando la comprensión contextual en diversas tareas. Posteriormente, GPT-2 en 2019 [51] se centró en la generación de texto autoregresiva, destacando por su coherencia y fluidez, aunque con preocupaciones sobre su posible uso indebido. En paralelo, T5 en 2019 [53] propuso un enfoque texto a texto, permitiendo abordar múltiples tareas con una arquitectura unificada. La llegada de GPT-3 en 2020 [15] con 175B parámetros supuso un avance significativo en few-shot learning, posibilitando la generación de contenido sin necesidad de ajuste fino específico. Finalmente, LaMDA en 2021 [64] optimizó la interacción en diálogos abiertos, mejorando la coherencia en conversaciones de largo alcance y consolidando los avances en modelos conversacionales.

5.2. ChatGPT

El lanzamiento de ChatGPT por OpenAI en noviembre de 2022 marcó un hito significativo en la evolución de los modelos de lenguaje, especialmente en aplicaciones de interacción conversacional [35]. ChatGPT es una versión mejorada de GPT-3, que fue afinada específicamente para mejorar la calidad de los diálogos y la coherencia en respuestas a preguntas abiertas y conversaciones continuas. ChatGPT se entrenó utilizando un enfoque de Reinforcement Learning from Human Feedback (RLHF), donde los humanos proporcionaron retroalimentación activa sobre las respuestas generadas, lo que permitió al modelo mejorar su capacidad de ofrecer respuestas más relevantes, precisas y menos propensas a producir información errónea o sesgada.

5.3. LLaMA

LLaMA (Large Language Model Meta AI), lanzado por Meta (anteriormente Facebook) en 2023, se presentó como un competidor directo de GPT-3 [66]. LLaMA se desarrolló con un enfoque en ser un modelo de gran escala, pero de mayor accesibilidad para la investigación académica y la comunidad de código abierto. LLaMA está diseñado para ser más eficiente y eficaz, permitiendo a los investigadores utilizar modelos de lenguaje a gran escala sin la necesidad de grandes cantidades de infraestructura.

Características clave de LLaMA:

- **Accesibilidad:** Meta liberó versiones de LLaMA con diferentes tamaños de parámetros, haciendo posible que se usaran en equipos de investigación más pequeños.

- **Desempeño a gran escala:** A pesar de que LLaMA no es tan grande como GPT-4, mostró un rendimiento comparable en tareas de procesamiento de lenguaje natural (PLN).
- **Eficiencia:** El modelo fue diseñado para ser más eficiente en términos de uso de recursos, lo que lo hacía más accesible para investigadores en lugar de depender de poderosas infraestructuras.

LLaMA ayudó a democratizar el acceso a modelos de lenguaje de gran escala, permitiendo que más organizaciones y académicos participaran en la investigación y desarrollo de modelos avanzados.

5.4. GPT-4

En marzo de 2023, OpenAI lanzó GPT-4, un modelo significativamente más avanzado que GPT-3 y ChatGPT [48]. Con una capacidad de procesar entradas multimodales (texto e imágenes), GPT-4 permitió nuevas aplicaciones en áreas como análisis de contenido visual y descripciones detalladas. Además, mostró mejoras significativas en razonamiento complejo, en la resolución de problemas matemáticos avanzados y en la comprensión de contextos más largos. sus principales características son:

- **Multimodalidad:** GPT-4 podía procesar imágenes junto con texto, lo que le permitió realizar tareas como la interpretación de gráficos, la descripción de imágenes y la creación de contenido visual a partir de texto.
- **Razonamiento mejorado:** El modelo mostró un mejor desempeño en tareas que requieren razonamiento lógico y la capacidad de resolver problemas complejos.
- **Mayor capacidad de manejo de contexto:** A diferencia de GPT-3, GPT-4 pudo gestionar contextos más largos, lo que resultó en una mayor coherencia durante conversaciones extensas.

5.5. DeepSeek

DeepSeek es un modelo propuesto en 2023, cuyo enfoque se centra en la búsqueda semántica profunda y la comprensión avanzada de consultas complejas [17]. Este modelo es particularmente útil en sistemas de búsqueda y recuperación de información, donde se requiere un entendimiento más allá de las coincidencias exactas de palabras clave, permitiendo la interpretación semántica de las consultas.

Características principales de DeepSeek:

- **Búsqueda semántica avanzada:** DeepSeek está optimizado para interpretar y responder a consultas complejas, mejorando la precisión de las búsquedas más allá de simples coincidencias de términos.
- **Comprensión del contexto:** Utiliza técnicas de procesamiento de lenguaje natural para interpretar el significado detrás de las consultas, lo que permite ofrecer resultados más relevantes y contextualizados.
- **Aplicaciones en la búsqueda web y bases de datos:** DeepSeek ha demostrado ser particularmente útil en la mejora de los motores de búsqueda, brindando una mejor experiencia al usuario mediante la entrega de resultados más precisos y personalizados.

DeepSeek es un ejemplo del progreso continuo hacia el entendimiento más profundo del lenguaje y su aplicación en la búsqueda semántica, un área clave en la IA.

5.6. LLaMA 2

En julio de 2023, Meta AI lanzó LLaMA 2, la segunda generación de su familia de modelos de lenguaje de código abierto [67]. Este lanzamiento representó un avance significativo en la democratización de modelos de lenguaje de gran escala, proporcionando a la comunidad investigadora y a empresas un acceso sin restricciones a modelos altamente capaces para diversas aplicaciones.

LLaMA 2 conserva la arquitectura basada en transformers de su predecesor, pero con mejoras en eficiencia, entrenamiento y alineación. Se lanzó en tres versiones principales, diferenciadas por su tamaño en parámetros: 7B (7 mil millones de parámetros), 13B y 65B, permitiendo su implementación en distintos entornos según las necesidades de hardware y rendimiento.

Las mejoras clave en LLaMA 2 incluyen:

- **Entrenamiento con un corpus de datos significativamente más amplio**, lo que mejoró su capacidad de generalización.
- **Optimización de eficiencia computacional**, logrando una mejor relación entre calidad de respuesta y costos computacionales.
- **Mejor alineación y reducción de sesgos**, gracias a un refinamiento en su entrenamiento mediante aprendizaje por refuerzo con retroalimentación humana (RLHF).
- **Mayor estabilidad en generación de texto**, proporcionando respuestas más coherentes y precisas en conversaciones prolongadas.

En términos de rendimiento, LLaMA 2-13B demostró ser comparable a modelos comerciales más grandes como GPT-3.5, mientras que la versión 65B se acercó en capacidad a GPT-4 en ciertas tareas de procesamiento de lenguaje natural.

5.7. Mistral

En septiembre de 2023, la empresa Mistral AI lanzó Mistral 7B, un modelo de lenguaje de código abierto que destacó por su eficiencia y rendimiento optimizado, superando a otros modelos de tamaño similar [28]. Con un enfoque en alta calidad y accesibilidad, Mistral 7B se convirtió rápidamente en una de las opciones más utilizadas en la comunidad de IA de código abierto.

Mistral 7B se diseñó con una arquitectura optimizada basada en transformers densos, pero con mejoras específicas:

- **Uso de bloques de atención eficiente:** Lo que permitió un mejor manejo del contexto y una mayor capacidad de generalización.
- **Reducción del tamaño del modelo sin sacrificar el rendimiento:** Optimizando la relación entre calidad y consumo de recursos.
- **Mejor respuesta en tareas de razonamiento y generación de texto:** Alcanzando un rendimiento comparable a modelos más grandes como LLaMA 2-13B.
- **Diseño modular y altamente escalable:** Facilitando la integración en diversas aplicaciones comerciales y de investigación.

En comparación con modelos de tamaño similar, Mistral 7B mostró superioridad en generación de código, razonamiento matemático y comprensión de instrucciones, convirtiéndolo en una opción viable frente a modelos más grandes como LLaMA 2-13B y Falcon 40B. Si bien Mistral 7B es más pequeño en parámetros que modelos como DeepSeek V2 (16B) o LLaMA 2-13B, su eficiencia le permitió superar a ambos en múltiples tareas. Su rendimiento fue similar al de modelos cerrados como GPT-3.5, pero con la ventaja de ser de código abierto y optimizado para ejecución en hardware menos costoso.

5.8. LLaMA 3

LLaMA 3 fue lanzado por Meta en marzo de 2024, mejorando la versión anterior de LLaMA con capacidades avanzadas y optimizaciones aún más eficientes [23]. Este modelo continúa con la tendencia de facilitar el acceso a modelos de gran escala mientras aumenta el rendimiento en tareas de procesamiento del lenguaje natural y la generación de texto.

Principales avances en LLaMA 3:

- **Mejora en eficiencia computacional:** LLaMA 3 se diseñó para ser aún más eficiente que sus predecesores, permitiendo que más instituciones, especialmente académicas y de investigación, puedan usarlo para diversos proyectos.
- **Mayor capacidad de razonamiento:** LLaMA 3 incluye avances en el razonamiento lógico y la resolución de problemas complejos, lo que lo hace más competitivo frente a modelos como GPT-4.
- **Entrenamiento más accesible:** Meta optimizó aún más los procesos de entrenamiento, lo que permite a los investigadores entrenar versiones del modelo en un rango más amplio de infraestructuras.

LLaMA 3 sigue siendo una opción accesible para la investigación en el ámbito de los LLMs, brindando una combinación de potencia y eficiencia.

5.9. DeepSeek V2

En 2024, DeepSeek AI presentó DeepSeek V2, una versión mejorada de su modelo de búsqueda semántica [18]. Esta versión avanzó notablemente en la precisión y adaptabilidad de los resultados, especialmente al integrar capacidades multimodales y mejorar la generación de respuestas contextualizadas en tiempo real.

Características de DeepSeek V2:

- **Mejor capacidad de comprensión multimodal:** DeepSeek v2 ahora es capaz de interpretar no solo texto, sino también imágenes, audio y otros tipos de datos, lo que le permite ofrecer respuestas aún más completas.
- **Generación de respuestas en tiempo real:** El modelo mejora la interacción con los usuarios al generar respuestas más rápidas y adaptadas a consultas complejas en tiempo real.
- **Mayor precisión en tareas de búsqueda y análisis:** La nueva versión presenta una mejora significativa en la precisión de la recuperación de información y la generación de respuestas relevantes, ajustadas al contexto.

DeepSeek v2 se destacó como una herramienta esencial para sistemas de búsqueda avanzados y la recuperación semántica de información, adaptándose a las necesidades de los usuarios de manera más efectiva.

5.10. GPT-4o

En mayo de 2024, OpenAI presentó GPT-4o (omni), una nueva iteración de su serie de modelos GPT, con mejoras significativas en velocidad, eficiencia y capacidades multimodales [49]. A diferencia de su predecesor GPT-4, que dependía de una arquitectura con distintos expertos (Mixture of Experts, MoE) y procesamiento separado para texto, imágenes y audio, GPT-4o adopta un enfoque nativamente multimodal, permitiendo procesar y generar texto, audio e imágenes con latencias mucho más bajas.

Uno de los aspectos más innovadores de GPT-4o es su capacidad para manejar múltiples modalidades de entrada y salida en tiempo real. A diferencia de modelos anteriores que requerían módulos separados para audio y visión, GPT-4o emplea un solo modelo unificado, optimizado para la inferencia eficiente en diversas tareas. Esto permite interacciones más naturales y rápidas, especialmente en aplicaciones conversacionales y de asistencia virtual.

Además, OpenAI mejoró la velocidad de respuesta del modelo, alcanzando tiempos de procesamiento hasta $2\times$ más rápidos que GPT-4-turbo, con una reducción considerable en los costos computacionales. Esto ha permitido desplegar GPT-4o como la opción por defecto en la API de OpenAI y en ChatGPT, democratizando el acceso a un modelo de alto rendimiento sin restricciones significativas.

GPT-4o es el primer modelo de OpenAI diseñado desde cero para integrar texto, audio e imágenes de manera fluida. En términos de procesamiento de audio, es capaz de reconocer y generar voz en tiempo real, permitiendo conversaciones fluidas sin la latencia que caracterizaba a versiones previas. En el ámbito visual, mejora la comprensión de imágenes y videos, ofreciendo capacidades avanzadas en reconocimiento de objetos, descripción de escenas y generación de contenido visual.

En tareas de procesamiento de lenguaje natural, GPT-4o supera a sus predecesores en comprensión, generación de texto, traducción automática y razonamiento contextual. Su capacidad para manejar consultas más complejas, junto con una mayor coherencia en respuestas largas, lo posiciona como uno de los modelos más potentes en la actualidad. También ha demostrado mejoras en la reducción de alucinaciones y en la precisión factual, aunque sigue enfrentando desafíos en ciertas tareas de razonamiento avanzado.

Gracias a su rapidez y capacidades mejoradas, GPT-4o ha sido implementado en una variedad de aplicaciones, desde asistentes virtuales avanzados hasta herramientas de productividad basadas en IA. Su integración en la aplicación web ChatGPT ha permitido a los usuarios experimentar una IA más interactiva y versátil, con respuestas más rápidas y contextualizadas. Además, la reducción en los costos de inferencia ha facilitado su adopción en productos comerciales y empresariales.

Con GPT-4o, OpenAI ha dado un paso importante hacia una inteligencia artificial más acce-

sible, interactiva y eficiente, consolidando su liderazgo en el desarrollo de modelos multimodales de alto rendimiento.

5.11. GPT-4o mini

En junio de 2024, OpenAI lanzó GPT-4o Mini, una versión optimizada y más ligera de GPT-4o, diseñada para ofrecer un equilibrio entre rendimiento, eficiencia y accesibilidad. Este modelo mantiene muchas de las capacidades avanzadas de su predecesor, pero con un enfoque en menor consumo computacional, lo que lo hace ideal para aplicaciones en dispositivos con recursos limitados y para despliegues comerciales a gran escala.

A diferencia del modelo completo GPT-4o, que está diseñado para tareas de alta demanda en procesamiento de texto, imagen y audio en tiempo real, GPT-4o Mini reduce la complejidad computacional, permitiendo tiempos de respuesta aún más rápidos con un menor costo energético. OpenAI ha optimizado este modelo para ofrecer rendimiento competitivo en tareas de Procesamiento de Lenguaje natural, sin comprometer significativamente la calidad de las respuestas.

Esta versión sigue el principio de modelos nativamente multimodales, lo que significa que puede manejar texto, imágenes y audio de manera integrada, aunque con menor capacidad de procesamiento en tareas extremadamente complejas. Esto lo hace ideal para asistentes virtuales, chatbots, herramientas de productividad y aplicaciones móviles, donde la latencia y el consumo de recursos son factores clave.

Si bien GPT-4o Mini conserva muchas de las mejoras de su versión completa, presenta algunas diferencias en términos de profundidad de razonamiento y capacidad de generación en tareas extensas. Entre sus características principales se destacan:

- **Mayor eficiencia en dispositivos con menos recursos:** facilitando su implementación en plataformas con hardware limitado.
- **Rendimiento optimizado para tareas de conversación y asistencia virtual:** con respuestas más rápidas y coherentes en diálogos cotidianos.
- **Menor latencia en comparación con modelos más grandes:** lo que lo hace ideal para aplicaciones en tiempo real.
- **Capacidades multimodales básicas:** aunque con menor profundidad en la comprensión y generación de contenido visual y auditivo en comparación con GPT-4o completo.

Gracias a su balance entre velocidad y capacidad, GPT-4o Mini ha sido adoptado en múltiples industrias, desde atención al cliente automatizada hasta educación y asistentes personales.

Empresas tecnológicas han comenzado a integrarlo en productos de bajo consumo computacional, asegurando interacciones de IA más fluidas y económicas.

Con el lanzamiento de GPT-4o Mini, OpenAI ha reforzado su enfoque en modelos accesibles y eficientes, ofreciendo una opción poderosa para usuarios y desarrolladores que requieren inteligencia artificial de alto nivel sin los costos computacionales de modelos más grandes.

5.12. Claude 3.5 Sonnet

En julio de 2024, Anthropic lanzó Claude 3.5 Sonnet, una evolución significativa dentro de la serie Claude 3, mejorando notablemente el rendimiento en razonamiento complejo, generación de texto y eficiencia computacional. Este modelo se posiciona como un competidor directo de GPT-4o, ofreciendo capacidades avanzadas en comprensión contextual y generación fluida de contenido.

Claude 3.5 Sonnet mantiene la arquitectura basada en transformers, pero con optimización en la gestión de memoria y eficiencia en la atención. Se ha reportado una notable mejora en la velocidad de procesamiento y en la capacidad para manejar contextos extensos sin degradación del rendimiento. Además, se han implementado técnicas avanzadas para reducir alucinaciones y mejorar la precisión factual.

En comparación con sus predecesores, Claude 3.5 Sonnet sobresale en:

- Mayor precisión en tareas de razonamiento lógico y matemático.
- Mejora en la comprensión de documentos extensos y consultas complejas.
- Optimización en generación de código, con respuestas más estructuradas y precisas.
- Reducción de sesgos y alineación mejorada con instrucciones del usuario.

Si bien Claude 3.5 Sonnet aún no ofrece capacidades multimodales tan avanzadas como GPT-4o, su procesamiento de texto sigue siendo de alto nivel, lo que lo hace ideal para tareas como asistencia en redacción, generación de informes, investigación académica y programación.

Comparado con modelos anteriores de Anthropic, Claude 3.5 Sonnet es más rápido y preciso, acercándose en rendimiento a modelos de última generación de GPT y DeepSeek.

Debido a su balance entre rendimiento y eficiencia, Claude 3.5 Sonnet ha sido adoptado en entornos empresariales, plataformas de asistencia virtual y generación de contenido automatizado. Además, su lanzamiento ha fortalecido la posición de Anthropic en la competencia por modelos de IA de alto rendimiento, consolidándose como una de las opciones más avanzadas para usuarios que requieren modelos precisos y eficientes.

5.13. DeppSeek V3

En enero de 2025, DeepSeek AI lanzó DeepSeek V3, su modelo más avanzado hasta la fecha, que marca una evolución significativa respecto a las versiones anteriores. DeepSeek V3 está diseñado para ofrecer capacidades de razonamiento avanzado, generación de contenido multimodal y aprendizaje más rápido, con un enfoque en mejorar la precisión en tareas complejas y optimizar el rendimiento en plataformas comerciales.

DeepSeek V3 utiliza una arquitectura de transformers de última generación, mejorando los mecanismos de atención y la gestión de memoria para trabajar con secuencias más largas y complejas. Su principal avance radica en la optimización de la interacción multimodal, donde puede procesar texto, imágenes y audio de manera más eficiente, permitiendo un rendimiento fluido en tareas que combinan estas modalidades.

Además, DeepSeek V3 ha mejorado el razonamiento lógico y la generación de código, lo que lo hace particularmente adecuado para aplicaciones en áreas de desarrollo de software y análisis técnico. Comparado con versiones anteriores, su capacidad de generalización y alineación con las intenciones del usuario es más precisa, lo que reduce significativamente los errores de interpretación.

Aunque DeepSeek V2 ya ofrecía un rendimiento sólido en múltiples dominios, DeepSeek V3 introduce mejoras importantes:

- **Mejor razonamiento contextual y manejo de tareas complejas:** El modelo ha optimizado su capacidad para resolver problemas que requieren múltiples pasos de razonamiento.
- **Generación de contenido más coherente y preciso:** Especialmente en tareas de escritura técnica y generación de código.
- **Mayor capacidad de alineación con instrucciones específicas del usuario:** Lo que lo hace más eficiente en tareas de personalización.
- **Mayor capacidad multimodal:** Con mejoras en la interpretación y creación de imágenes y audio a partir de texto, aunque sigue siendo un modelo más centrado en texto en comparación con modelos completamente multimodales como GPT-4o.

Gracias a su rendimiento mejorado, DeepSeek V3 ha sido adoptado en sectores que requieren precisión avanzada, como la generación de código, análisis de datos, desarrollo de software y asistencia en investigación. Su integración con plataformas de desarrollo de software automatizado ha permitido a empresas optimizar la creación de soluciones tecnológicas más rápidamente.

Además, DeepSeek V3 ha ganado popularidad en herramientas de productividad y asistentes virtuales, gracias a su capacidad para mantener diálogos más coherentes y generar respuestas más precisas en conversaciones prolongadas. Con este lanzamiento, DeepSeek se consolida como un jugador clave en el ámbito de la inteligencia artificial, ofreciendo soluciones de alto rendimiento para una variedad de aplicaciones comerciales.

6. Prompt Engineering

El Prompt Engineering (ingeniería de prompts) ha surgido como una técnica clave para mejorar las capacidades de los Grandes Modelos de Lenguaje preentrenados [55]. Consiste en diseñar de manera estratégica instrucciones específicas para tareas, conocidas como prompts, que guían la salida del modelo sin modificar sus parámetros. La importancia del prompt engineering es especialmente evidente en su impacto transformador sobre la adaptabilidad de estos modelos. Al ofrecer un mecanismo para ajustar los resultados del modelo mediante instrucciones cuidadosamente elaboradas, el prompt engineering permite a estos modelos sobresalir en una amplia gama de tareas y dominios. Esta adaptabilidad se diferencia de los paradigmas tradicionales, donde a menudo se requiere un reentrenamiento del modelo o un ajuste fino extenso para lograr un rendimiento específico para una tarea.

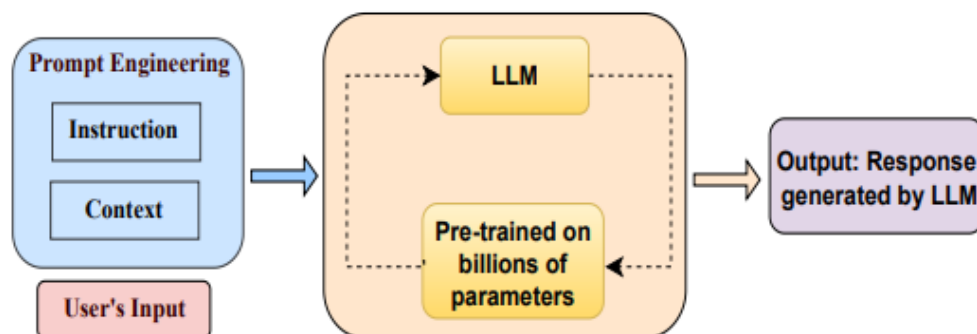


Figura 2.2: Flujo del prompt engineering

A continuación se describen algunas de las técnicas de Prompt Engineering mas comunes.

6.1. Zero-Shot Prompting

El zero-shot prompting representa un cambio de paradigma en la utilización de Grandes Modelos del Lenguaje [52].

Esta técnica elimina la necesidad de grandes volúmenes de datos de entrenamiento, basándose en prompts diseñados cuidadosamente para guiar al modelo en la resolución de tareas nove-

dosas. Específicamente, el modelo recibe una descripción de la tarea dentro del prompt, pero no dispone de datos etiquetados para entrenarse en mapeos específicos de entrada-salida.

En su lugar, el modelo aprovecha su conocimiento preexistente para generar predicciones basadas en el prompt proporcionado para la nueva tarea.

6.2. Few-Shot Prompting

El few-shot prompting proporciona a los modelos algunos ejemplos de entrada-salida para inducir la comprensión de una tarea específica [14], a diferencia del zero-shot prompting, donde no se suministran ejemplos. Incluir incluso unos pocos ejemplos de alta calidad ha demostrado mejorar el rendimiento del modelo en tareas complejas en comparación con la ausencia de estos.

Sin embargo, el few-shot prompting requiere tokens adicionales para incorporar los ejemplos, lo que puede encarecer el coste de la tarea en entradas de texto más extensas. Además, la selección y composición de los ejemplos dentro del prompt pueden influir significativamente en el comportamiento del modelo, y sesgos como la preferencia por palabras frecuentes pueden seguir afectando los resultados.

Aunque el few-shot prompting mejora las capacidades en tareas complejas, una ingeniería de prompts cuidadosa es esencial para alcanzar un rendimiento óptimo y mitigar sesgos no intencionados.

6.3. Chain-of-Thought Prompting

Los Grandes Modelos del Lenguaje suelen enfrentar dificultades en tareas de razonamiento complejo, lo que limita su potencial. Con el objetivo de abordar esta limitación, se desarrolló el concepto de Chain-of-Thought (CoT) prompting [71], una técnica que permite estructurar prompts de manera que faciliten procesos de razonamiento coherentes y paso a paso.

La principal contribución de este enfoque radica en su capacidad para inducir respuestas más estructuradas y reflexivas en comparación con los prompts tradicionales. A través de una serie de experimentos, se demostró que el CoT prompting guía a los modelos a seguir una cadena lógica de razonamiento, lo que resulta en respuestas que reflejan una comprensión más profunda de los prompts dados. Por ejemplo, en un problema matemático de múltiples pasos, el prompt incluiría tanto el proceso de razonamiento como la respuesta final, imitando la forma en que los humanos descomponen los problemas en pasos intermedios lógicos.

6.4. Instruction Following

La técnica de instruction following (seguimiento de instrucciones) se centra en diseñar prompts que especifiquen explícitamente las instrucciones sobre cómo el modelo debe com-

portarse durante el proceso de inferencia [39]. Esto incluye detalles como el formato de la respuesta esperada, el estilo de escritura, e incluso las restricciones de tiempo para la generación. La capacidad de un modelo para seguir instrucciones con precisión depende tanto de la claridad del prompt como de la estructura subyacente del modelo.

Esta técnica es especialmente útil en tareas de generación de texto controlada, como la redacción de resúmenes, traducción automática o generación de contenido específico en un formato predeterminado. La implementación efectiva de instruction following depende de la habilidad del modelo para interpretar y ejecutar las instrucciones de forma coherente, lo que requiere un equilibrio entre la precisión del modelo y la claridad del prompt.

6.5. Limitaciones de Prompt Engineering

El prompt engineering ha revolucionado la forma en que los modelos preentrenados son utilizados en diversas aplicaciones, permitiendo un alto grado de adaptabilidad sin la necesidad de reentrenamiento. Las técnicas presentadas representan solo una fracción del potencial que ofrece el campo.

Sin embargo, aunque estas estrategias optimizan la generación de respuestas, los modelos siguen enfrentando limitaciones cuando se requiere información actualizada o específica que no está presente en sus datos de entrenamiento. Para abordar esta limitación, surge el enfoque de Retrieval-Augmented Generation (RAG), que combina la capacidad generativa de los modelos con sistemas de recuperación de información en tiempo real.

7. Retrieval Augmented Generation (RAG)

Los Grandes Modelos del Lenguaje han logrado un éxito notable; sin embargo, aún presentan limitaciones significativas, especialmente en tareas específicas de dominio o con alta demanda de conocimiento. En particular, generan alucinaciones al procesar consultas que exceden sus datos de entrenamiento o requieren información actualizada.

Para abordar estos desafíos, se desarrolla la técnica Retrieval Augmented Generation (Generación Aumentada por Recuperación, o RAG) [34], que optimiza el desempeño de los LLMs mediante la recuperación de fragmentos de documentos relevantes desde bases de conocimiento externas, utilizando cálculos de similitud semántica. Al referenciar información externa, RAG mitiga la generación de contenido incorrecto o alucinado. Su integración con los LLMs ha impulsado su adopción generalizada, consolidando a RAG como una tecnología clave en la evolución de los chatbots y mejorando la aplicabilidad de los LLMs en entornos reales.

El funcionamiento de un sistema RAG se basa en la combinación de recuperación de información y generación de texto, optimizando la precisión y actualidad de las respuestas mediante LLMs.

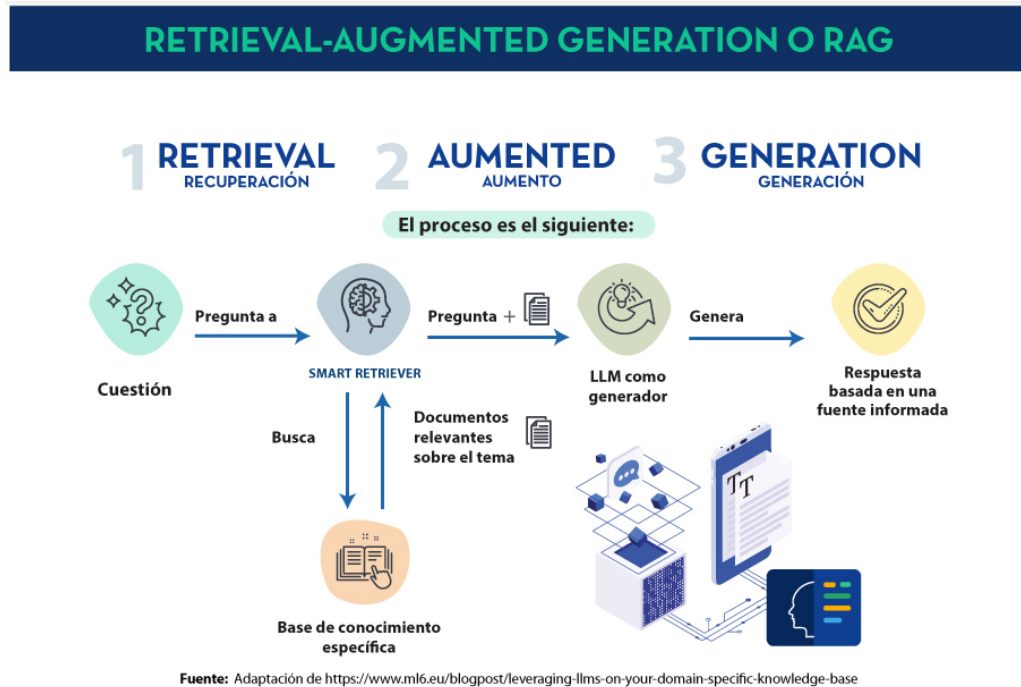


Figura 2.3: flujo de RAG

- En la fase de **recuperación**, la consulta del usuario se convierte en un vector mediante un modelo de embeddings y se compara con una base de datos vectorizada, que almacena documentos representados como vectores en un espacio de alta dimensión.
- Estas bases de datos permiten realizar recuperación de conocimiento relevante mediante algoritmos de similitud, reduciendo el costo computacional en la recuperación de información relevante.
- Los fragmentos más cercanos semánticamente a la consulta son recuperados y concatenados con el prompt, ingresando a la fase de **generación**, donde un LLM procesa la información contextualizada y genera una respuesta fundamentada.
- Finalmente, en la fase de **aumento**, se pueden aplicar diferentes técnicas para optimizar la coherencia y factualidad de la generación.

Este enfoque permite superar limitaciones de los LLMs, como la generación de contenido alucinatorio o el acceso a información desactualizada, consolidando a RAG como una archi-

tectura clave en la integración de modelos de lenguaje con bases de conocimiento dinámicas y eficientes.

7.1. Clasificación del RAG

El paradigma de investigación en RAG evoluciona continuamente y se clasifica en tres bloques:

7.1.1. Naive RAG

Representa la metodología inicial de RAG. Se basa en un esquema “Retrieve-Read”, compuesto por tres fases: indexación, recuperación y generación [41]. En la fase de indexación, se procesan datos en diversos formatos (PDF, HTML, Word, Markdown), normalizándolos a texto plano. Luego, se segmentan en fragmentos más pequeños para ajustarse a las limitaciones de contexto de los LLMs. Estos fragmentos son convertidos en vectores mediante un modelo de embeddings y almacenados en una base de datos vectorizada, lo que permite búsquedas eficientes en la fase de recuperación. Durante la recuperación, la consulta del usuario se codifica en un vector utilizando el mismo modelo de embeddings. Posteriormente, se calculan similitudes semánticas entre la consulta y los fragmentos indexados, seleccionando los K fragmentos más relevantes para ampliar el contexto del prompt. En la fase de generación, la consulta y los documentos recuperados se combinan en un prompt que el LLM procesa para generar una respuesta. Dependiendo de la tarea, el modelo puede combinar su conocimiento paramétrico con la información recuperada o limitarse exclusivamente a esta última. En diálogos continuos, el historial conversacional se incorpora para mejorar la coherencia en interacciones multi-turno.

7.1.2. Advanced RAG

Introduce mejoras específicas para superar las limitaciones de Naive RAG, con un enfoque en la optimización de la recuperación de información mediante estrategias pre-retrieval y post-retrieval [27]. Para abordar problemas de indexación, Advanced RAG refina sus técnicas mediante el uso de ventanas deslizantes, segmentación granular y metadatos, además de implementar métodos de optimización para mejorar la eficiencia del proceso de recuperación.

- **Pre-Retrieval Process:** En esta etapa, se optimizan tanto la estructura de indexación como la consulta original del usuario. La mejora en la indexación busca aumentar la calidad del contenido indexado mediante estrategias como mayor granularidad de datos, optimización de estructuras de índices, adición de metadatos, alineación de información y recuperación híbrida. Por otro lado, la optimización de consultas tiene como objetivo hacer

que la pregunta del usuario sea más clara y adecuada para la recuperación, utilizando técnicas como reescritura de consultas, transformación, expansión de consultas y otras estrategias.

- **Post-Retrieval Process:** Una vez recuperado el contexto relevante, es fundamental integrarlo de manera efectiva con la consulta original. Las principales estrategias incluyen re-ranking y context compression. El re-ranking reorganiza los fragmentos recuperados, priorizando la información más relevante en los extremos del prompt, una técnica utilizada en frameworks como LlamaIndex, LangChain y HayStack. Dado que alimentar todos los documentos recuperados a un LLM puede generar sobrecarga de información, diluyendo la relevancia del contenido, el proceso post-retrieval se centra en la selección de información clave, priorización de secciones críticas y reducción del contexto para mejorar la precisión y eficiencia en la generación de respuestas.

7.1.3. Modular RAG

Optimiza la adaptabilidad y precisión del paradigma RAG al introducir una arquitectura flexible con módulos especializados [74]. Mejora la recuperación mediante técnicas avanzadas como búsqueda híbrida, multi-query retrieval y re-ranking inteligente. Sus nuevos módulos, como Search, Memory, Routing y Predict, permiten una recuperación más precisa y contextualizada, mientras que el Task Adapter personaliza la generación para diversas tareas. A nivel estructural, Modular RAG reemplaza el flujo tradicional "Retrieve-Read" por esquemas dinámicos como Rewrite-Retrieve-Read, Generate-Read y Recite-Read, mejorando la integración de datos y reduciendo sesgos. Además, incorpora estrategias iterativas y autoajustables como Demonstrate-Search-Predict (DSP) y ITER-RETGEN, optimizando la interacción entre módulos. Su flexibilidad permite la integración con técnicas de fine-tuning y reinforcement learning, optimizando tanto el retriever como el generador para mejorar la relevancia y coherencia de las respuestas.

7.2. Técnicas de mejora de RAG

Los sistemas RAG han ganado mucha atención debido a su capacidad para combinar el poder de los modelos generativos con la recuperación de información. Estos sistemas permiten a los modelos generar respuestas de alta calidad basadas en documentos relevantes recuperados de grandes bases de datos. Sin embargo, para mejorar aún más la precisión y la relevancia de las respuestas generadas, se han desarrollado técnicas avanzadas que optimizan tanto la recuperación como la generación de información. Entre estas técnicas destacan la búsqueda híbrida o hybrid search, el reordenamiento o reranking y el RAG contextual.

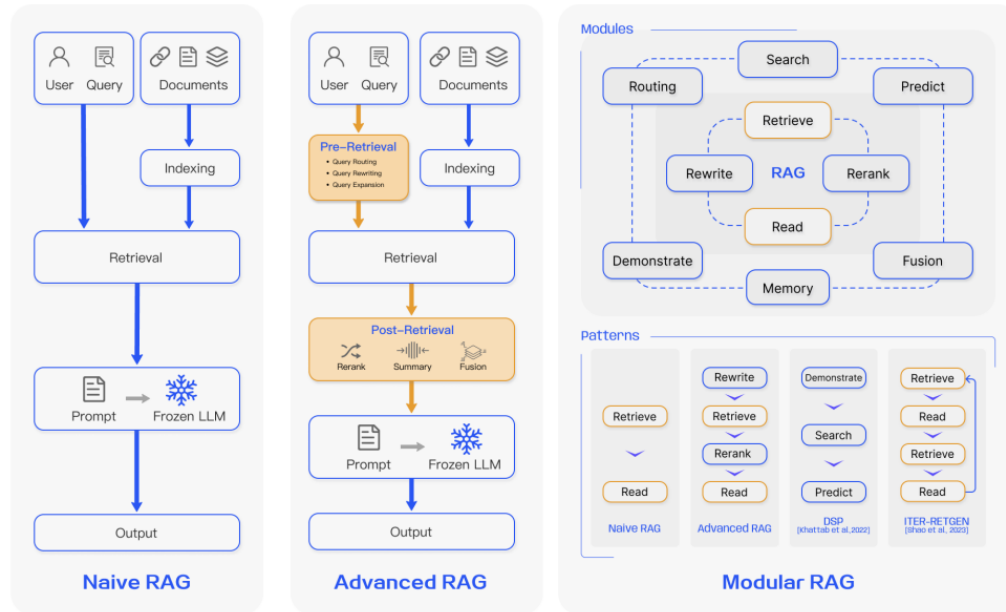


Figura 2.4: Tipos de RAG

7.2.1. Hybrid Search

La Búsqueda Híbrida combina dos enfoques clave de recuperación de información: la búsqueda exacta basada en palabras clave y la búsqueda densa basada en embeddings [13]. Cada uno de estos métodos tiene ventajas y limitaciones, por lo que su combinación permite mejorar significativamente la calidad de la recuperación de documentos y, por ende, la precisión de las respuestas generadas por los modelos de lenguaje.

- La **búsqueda exacta** utiliza palabras clave para localizar documentos mediante técnicas como el índice invertido, BM25 y TF-IDF, lo que permite respuestas rápidas y eficientes en grandes bases de datos. Sin embargo, su limitación radica en la falta de comprensión semántica, lo que puede excluir documentos relevantes si no hay coincidencias exactas.
- Por otro lado, la **búsqueda densa** convierte consultas y documentos en embeddings vectoriales utilizando LLMs, lo que permite medir similitudes semánticas y recuperar información más flexible. Aunque más precisa en términos de significado, esta técnica requiere mayor capacidad computacional y modelos preentrenados, dependiendo de herramientas como bases de datos vectoriales para optimizar la búsqueda en grandes volúmenes de datos.
- La combinación de búsqueda exacta y búsqueda densa en un **modelo híbrido** aprovecha sus fortalezas complementarias, logrando un equilibrio entre eficiencia y precisión en la recuperación de documentos dentro de sistemas RAG. En este enfoque, la búsqueda exacta

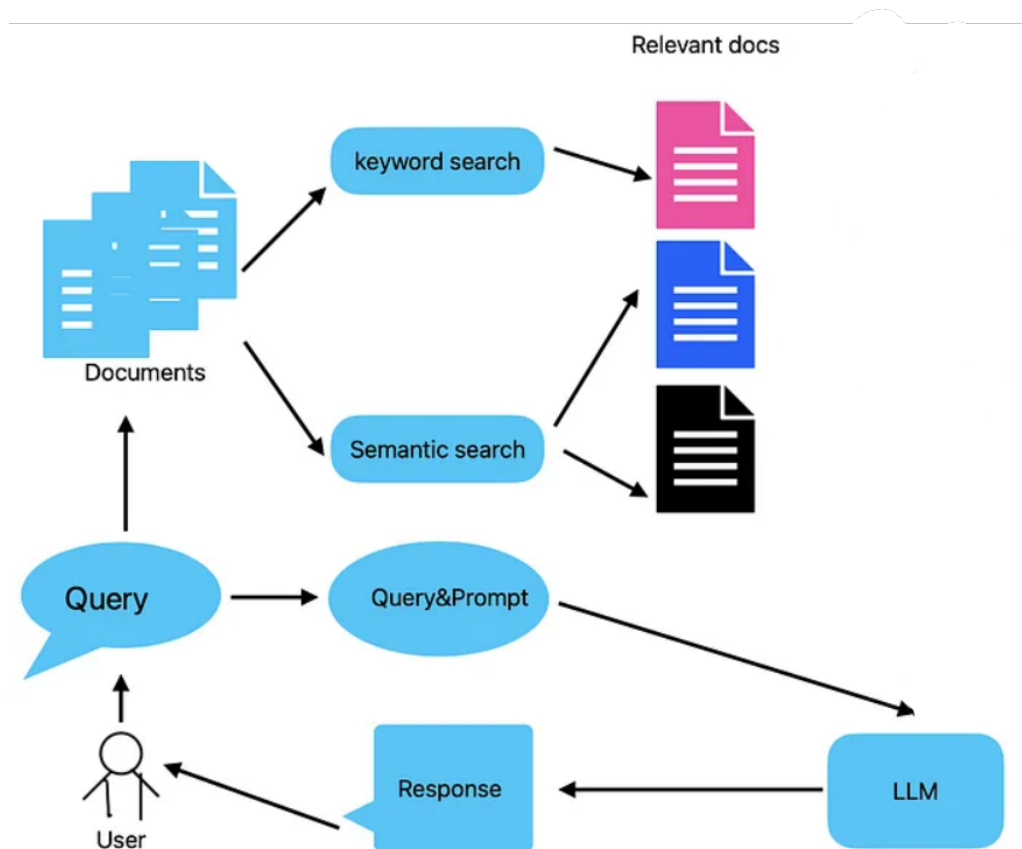


Figura 2.5: Flujo de búsqueda híbrida

actúa como un filtro inicial rápido, identificando documentos potencialmente relevantes. Posteriormente, la búsqueda densa refina estos resultados, asegurando que la información seleccionada sea semánticamente relevante para la consulta. Para fusionar ambas metodologías, se pueden emplear estrategias como la fusión de puntajes, que normaliza y combina los resultados de cada método en un ranking unificado, o el filtrado progresivo, donde la búsqueda exacta reduce el conjunto de documentos antes de aplicar la búsqueda densa para su reordenamiento. Al integrar estas técnicas, los sistemas RAG mejoran la precisión de sus respuestas y reducen la recuperación de documentos irrelevantes, optimizando tanto la velocidad del proceso como la calidad de la información presentada al usuario.

En esencia, la búsqueda híbrida maximiza la eficiencia del proceso de recuperación al utilizar la rapidez de la búsqueda exacta junto con la capacidad semántica de la búsqueda densa. Este enfoque es particularmente útil en escenarios donde la consulta puede formularse de múltiples maneras y no siempre coincide textualmente con los documentos relevantes disponibles en la base de datos.

7.2.2. Reranking

El re-ranking es una técnica utilizada para mejorar la calidad de los documentos recuperados antes de ser utilizados en la generación de respuestas. Su propósito es reorganizar la lista de documentos obtenida en la fase de recuperación, priorizando aquellos que son más relevantes para la consulta del usuario.

En los sistemas RAG, la primera fase de búsqueda (ya sea exacta, densa o híbrida) devuelve un conjunto de documentos ordenados por su similitud con la consulta. Sin embargo, este orden inicial no siempre es el óptimo, ya que los métodos de recuperación pueden devolver documentos parcialmente relevantes o incluso irrelevantes. Aquí es donde entra el re-ranking, aplicando técnicas avanzadas para mejorar la selección final de documentos que serán procesados por el modelo generador [45].

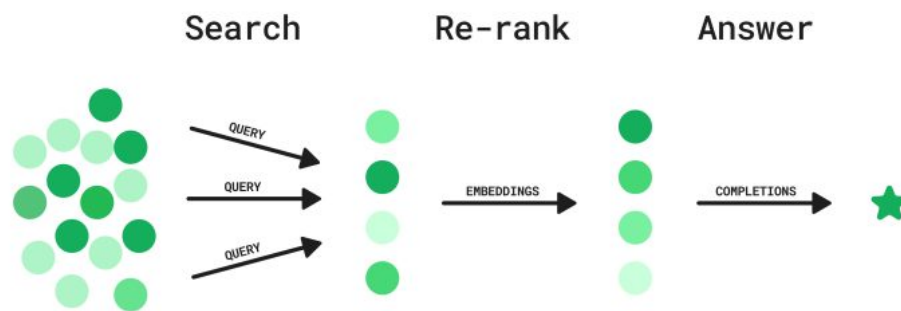


Figura 2.6: Esquema de Reranking

Existen varios enfoques para realizar el re-ranking en sistemas RAG. Estos métodos pueden variar en complejidad y costo computacional, dependiendo del nivel de precisión requerido.

- **Re-ranking basado en heurísticas:** Este enfoque utiliza reglas predefinidas y métricas estadísticas para reordenar los documentos recuperados sin necesidad de modelos de aprendizaje profundo. Algunos métodos comunes incluyen algoritmos como TF-IDF o BM25 [37], que prioriza documentos donde los términos clave aparecen con mayor frecuencia y relevancia, y el criterio de posición de aparición, que favorece documentos donde las palabras clave se encuentran al inicio del texto. También puede aplicarse un filtro basado en la longitud del documento, dando prioridad a textos más concisos si se asume que contienen información más relevante. Aunque este método es rápido y eficiente, su principal limitación es que no tiene en cuenta el significado semántico de las palabras, lo que puede afectar la precisión de la recuperación en consultas más complejas.

- **Re-ranking con modelos de lenguaje:** En este método, se utilizan modelos del lenguaje para evaluar la relevancia semántica de los documentos con respecto a la consulta. Modelos como BERT Cross-Encoder procesan cada consulta junto con un documento y generan una puntuación de relevancia basada en su relación semántica. MonoT5, por otro lado, reformula la tarea de re-ranking como un problema de clasificación, asignando una probabilidad a cada documento según su pertinencia. Otra alternativa es ColBERT (Contextualized Late Interaction BERT) [29], que optimiza la comparación de embeddings para equilibrar precisión y eficiencia. Aunque estos modelos mejoran significativamente la calidad del ranking, requieren más recursos computacionales, lo que puede dificultar su implementación en sistemas de respuesta en tiempo real sin optimización adecuada.
- **Re-ranking basado en feedback y aprendizaje reforzado:** Este enfoque se basa en el ajuste dinámico del ranking mediante el aprendizaje a partir de datos históricos o interacciones del usuario [46]. En el aprendizaje supervisado, los modelos son entrenados con conjuntos de datos etiquetados, donde se identifican manualmente los documentos más relevantes para ciertas consultas. Por otro lado, los sistemas interactivos pueden aprovechar el feedback del usuario, analizando clics, tiempo de lectura o valoraciones para ajustar la selección de documentos. Además, el aprendizaje por refuerzo permite que el modelo refine su estrategia de re-ranking con el tiempo, optimizando la recuperación mediante técnicas como bandits contextuales o algoritmos de refuerzo como REINFORCE. Este enfoque es ideal para sistemas en evolución continua, ya que mejora la precisión de la información recuperada conforme se acumulan datos de uso real.

7.2.3. Contextual RAG

En el RAG tradicional, los documentos suelen dividirse en fragmentos más pequeños para facilitar la recuperación. Aunque este enfoque funciona bien para muchas aplicaciones, puede generar problemas cuando los fragmentos individuales carecen de suficiente contexto.

La Recuperación Contextual (Contextual Retrieval) resuelve este problema añadiendo un contexto explicativo específico del fragmento antes de realizar el embedding [69].

Anthropic presenta el concepto de Recuperación Contextual como una mejora en los sistemas RAG [9]. La técnica agrega contexto específico a los fragmentos de información mediante embeddings contextuales y la creación de un índice BM25 contextual, lo que reduce significativamente los errores de recuperación.

La Recuperación Contextual soluciona los problemas asociados con la falta de contexto al agregar información relevante específica para cada fragmento recuperado antes de su procesamiento. Esto se logra mediante dos sub-técnicas clave:

- **Embeddings Contextuales:** Se mejora la representación semántica de los fragmentos al añadir información adicional antes de la codificación. Esto asegura que cada fragmento de texto mantenga la relevancia contextual al momento de ser procesado.
- **BM25 Contextual:** Se utiliza el índice BM25, un algoritmo tradicional de recuperación, mejorado con contexto adicional. Esto permite realizar una búsqueda más precisa y relevante mediante coincidencias léxicas, integrando la semántica de los fragmentos.

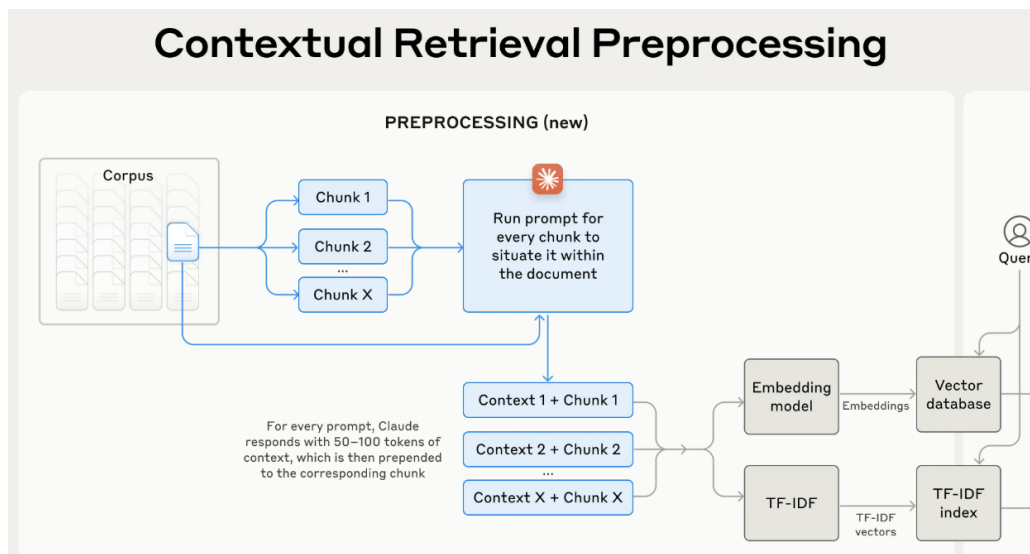


Figura 2.7: Contextual retrieval propuesto por Anthropic

Al integrar estas técnicas, se logran mejoras significativas en la precisión y relevancia de la recuperación de información. En estudios y pruebas, se ha demostrado que esta metodología reduce los errores de recuperación en un 49 %. Además, cuando se combina con técnicas de re-ranking, los resultados mejoran aún más, alcanzando un 67 % de mejora en la precisión de las respuestas generadas por los modelos.

Este enfoque es especialmente útil en sistemas que manejan grandes volúmenes de datos, como bases de conocimiento extensas, ya que el contexto adicional mejora la relevancia de los fragmentos recuperados. A su vez, esto garantiza que las respuestas generadas sean más precisas y alineadas con la consulta original del usuario.

Además, la implementación de **prompt caching** (almacenamiento de fragmentos frecuentes) en este sistema optimiza los costos y reduce la latencia. Con esta técnica, los desarrolladores pueden almacenar consultas previas y reutilizarlas, mejorando el rendimiento y haciendo que las respuestas sean más rápidas y menos costosas de generar.

La Recuperación Contextual mejora sustancialmente los sistemas RAG al permitir que los fragmentos de información sean procesados de manera más inteligente y precisa. Esto no solo

reduce los fallos en la recuperación de datos, sino que también optimiza la calidad de las respuestas generadas por modelos de IA. Es una solución eficiente y escalable para aplicaciones que requieren una alta precisión en la recuperación de información.

Se han propuesto otros enfoques para utilizar el contexto y mejorar la recuperación. Algunas propuestas incluyen: añadir resúmenes genéricos de documentos a los fragmentos [70], embedding hipotético de documentos [21] e indexación basada en resúmenes [36].

7.2.4. Agentic RAG

El RAG Agéntico representa un cambio de paradigma al incorporar agentes autónomos capaces de tomar decisiones dinámicas y optimizar flujos de trabajo [61]. A diferencia de los sistemas estáticos, el RAG Agéntico implementa refinamiento iterativo y estrategias de recuperación adaptativa para abordar consultas complejas, en tiempo real y multidominio. Este enfoque aprovecha la modularidad de los procesos de recuperación y generación, al tiempo que introduce autonomía basada en agentes.

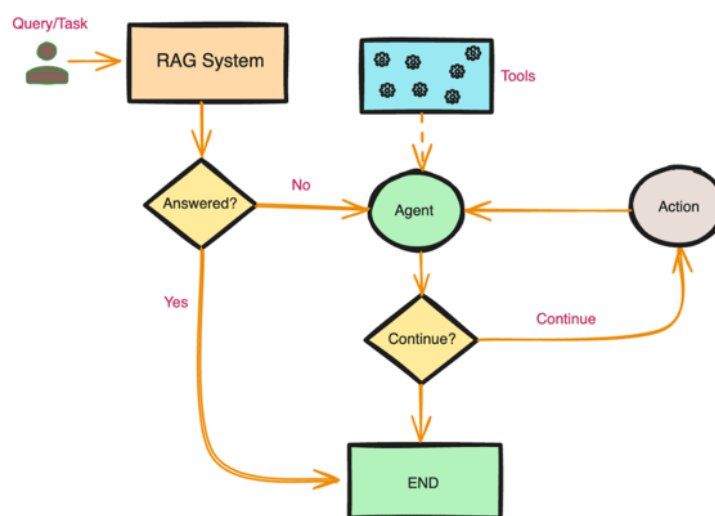


Figura 2.8: Flujo de Agentic RAG

Las características clave del Agentic RAG incluyen:

- **Toma de Decisiones Autónoma:** Los agentes evalúan y gestionan de manera independiente las estrategias de recuperación según la complejidad de la consulta.
- **Refinamiento Iterativo:** Incorpora bucles de retroalimentación para mejorar la precisión en la recuperación y la relevancia de las respuestas.
- **Optimización del Flujo de Trabajo:** Orquesta dinámicamente las tareas, permitiendo eficiencia en aplicaciones en tiempo real.

Si bien los agentes pueden incorporarse en diferentes etapas del pipeline de RAG, el término RAG Agéntico generalmente se refiere al uso de agentes en el componente de recuperación. Específicamente, el componente de recuperación se vuelve agéntico mediante el uso de agentes de recuperación con acceso a diversas herramientas de recuperación, tales como:

- Motores de búsqueda vectorial (también llamados motores de consulta) que realizan búsquedas vectoriales sobre un índice vectorial (como en pipelines RAG tradicionales).
- Búsqueda web.
- Calculadoras.
- APIs para acceder a software de manera programática, como programas de correo electrónico o chat.

El agente de RAG puede razonar y actuar en diversos escenarios de recuperación, por ejemplo:

- Decidir si es necesario recuperar información o no.
- Elegir qué herramienta utilizar para recuperar información relevante.
- Formular la consulta de búsqueda.
- Evaluar el contexto recuperado y determinar si es necesario realizar una nueva recuperación.

A diferencia de la arquitectura secuencial y simplista del RAG tradicional, el núcleo del Agentic RAG es el agente. Las arquitecturas de RAG Agéntico pueden presentar distintos niveles de complejidad. En su forma más simple, una arquitectura RAG de agente único actúa como un simple enrutador. Sin embargo, también es posible incorporar múltiples agentes dentro de una arquitectura RAG multiagente.

Existen por lo tanto dos arquitecturas principales en Agentic RAG [16]:

- **RAG de Agente Único (Router):** En su forma más simple, el RAG Agéntico funciona como un router. Esto significa que hay al menos dos fuentes de conocimiento externas, y el agente decide de cuál recuperar contexto adicional. Sin embargo, estas fuentes de conocimiento externas no se limitan únicamente a bases de datos (vectoriales). También es posible recuperar información a través de herramientas adicionales. Por ejemplo, se puede realizar una búsqueda web o utilizar una API para obtener información de diversas fuentes.

- **Sistemas RAG Multiagente:** Como es de esperar, el sistema de agente único tiene sus limitaciones, ya que concentra en un solo agente las tareas de razonamiento, recuperación y generación de respuestas. Por ello, resulta beneficioso encadenar múltiples agentes dentro de una aplicación RAG multiagente. Por ejemplo, se puede emplear un agente maestro que coordine la recuperación de información entre múltiples agentes especializados. Un agente podría encargarse de recuperar información de fuentes de datos internas propietarias. Otro agente podría especializarse en la recuperación de datos de cuentas personales, como correos electrónicos o chats. Un tercer agente podría enfocarse en recuperar información pública a través de búsquedas web.

A pesar de sus avances, el RAG Agéntico enfrenta algunos desafíos:

- **Complejidad en la Coordinación:** La gestión de interacciones entre agentes requiere mecanismos de orquestación sofisticados.
- **Sobrecarga Computacional:** El uso de múltiples agentes incrementa los requisitos de recursos para flujos de trabajo complejos.
- **Limitaciones de Escalabilidad:** Aunque es escalable, la naturaleza dinámica del sistema puede generar una carga significativa en los recursos computacionales ante volúmenes elevados de consultas.

El RAG Agéntico sobresale en dominios como el soporte al cliente, el análisis financiero y las plataformas de aprendizaje adaptativo, donde la adaptabilidad dinámica y la precisión contextual son fundamentales.

8. Conclusiones

Con esta revisión del estado del arte se ha hecho una recopilación de las principales herramientas y tecnologías mas clásicas asociadas a soluciones como la que abarca el proyecto. Del mismo modo, se ha realizado una revisión de los últimos avances en Inteligencia Artificial y Procesamiento de Lenguaje Natural, así como una evolución de los distintos Grandes Modelos del Lenguaje que han ido surgiendo, junto a sus tecnologías asociadas. Todo esto nos servirá de base a la hora de abordar el desarrollo de la solución final, donde nos centraremos en las tecnologías mas específicas que se emplearán.

Capítulo 3

Métodos y recursos

1. Introducción

Este capítulo presenta la metodología y las herramientas empleada en el desarrollo del proyecto, detallando cada fase de diseño, implementación y pruebas. Tras realizar una planificación exhaustiva y un análisis del estado del arte en el capítulo anterior, se ha adquirido un marco preciso sobre los objetivos y las etapas necesarias para alcanzar los objetivos de este proyecto. Con esta base sólida, se procede a estructurar y ejecutar el desarrollo del producto utilizando metodologías y recursos que se ajusten a las necesidades de automatización y procesamiento de datos que plantea el proyecto.

1.1. Descripción general del capítulo

El capítulo muestra un desglose de las decisiones metodológicas y técnicas que sustentan el diseño y desarrollo de la solución. Incluye una descripción de los recursos tecnológicos, tanto de hardware como de software, empleados para asegurar la eficiencia y precisión en el proceso de identificación y clasificación de convocatorias de ayudas. Cada sección aborda de manera específica las técnicas y herramientas implementadas en cada fase del proyecto, desde la recopilación de datos con el extractor de ayudas, la generación de datos estructurados con el bloque de procesamiento, y la construcción de la herramienta de consulta de información basada en un sistema conversacional basado en agentes inteligentes.

1.2. Importancia del diseño, desarrollo e implementación en la investigación

El desarrollo de una solución automatizada para sintetizar y recuperar información de convocatorias es esencial en un entorno con abundancia de datos y formatos diversos. Un diseño eficiente asegura que cada componente cumpla su función, maximizando la precisión y minimizando redundancias. La implementación adapta técnicas de recuperación de información y procesamiento de lenguaje natural basadas en Inteligencia Artificial Generativa, del estado del arte a las necesidades del proyecto, mientras que una metodología ágil permite ajustes continuos y mejora constante. Este capítulo describe los métodos y recursos que sustentan la construcción de la solución, transformando la planificación teórica en un producto funcional capaz de procesar grandes volúmenes de datos y facilitar su análisis para entidades específicas.

2. Objetivos y Competencias

El propósito de esta sección es detallar los objetivos específicos de la fase de desarrollo e implementación de la solución, y las competencias técnicas y metodológicas necesarias para cumplir con dichos objetivos. Estos aspectos son fundamentales para asegurar que cada etapa del proyecto esté alineada con los resultados esperados y que los participantes en el desarrollo cuenten con las habilidades necesarias para abordar los desafíos que puedan surgir.

2.1. Objetivos específicos de esta fase del proyecto

Los objetivos de la fase de desarrollo e implementación son los siguientes:

- **Implementación de un sistema automático de extracción de datos de ayudas:** Desarrollo de una herramienta basada en web scraping para la extracción de información. Esta herramienta es modular, cada uno de estos módulos está adaptado a una fuente de convocatorias específica. La herramienta navega por el sitio web, identifica las diferentes convocatorias disponibles y extrae la información necesaria, como el texto resumen de la ayuda, la ficha técnica, y los documentos asociados a la convocatoria.
- **Desarrollo de un módulo de procesamiento de datos:** Este módulo carga las fuentes de datos asociadas a cada ayuda y las procesa mediante técnicas de extracción basadas en IA Generativa. El objetivo es extraer diferentes parámetros de las fuentes de datos y obtener como resultado un conjunto de datos estructurados en formato tabla.
- **Desarrollo de un módulo de construcción de bases de datos:** Este módulo parte de las fuentes de datos originales y de los datos estructurados y contruye una base de

datos documental y una base de datos estructurados. Mientras que en este caso, la base de datos estructural se crea de manera directa, para la base de datos documental es necesario implementar un bloque de procesamiento, donde los diferentes documentos se dividen en chunks y se generan sus correspondientes embeddings.

- **Implementación de una aplicación RAG mediante técnicas avanzadas:** Construcción de una solución de recuperación de información basada en un RAG multiagente, donde a través de lenguaje natural pueden generarse consultas de datos tanto a la base de datos estructurada como a la documental.
- **Implementación de una aplicación web para el uso del sistema multiagente como asistente conversacional:** Construcción de una pequeña interfaz de chat para la explotación de la herramienta de recuperación de información.

Estos objetivos son claves para asegurar que el sistema cumpla con los requisitos del proyecto y proporcione una herramienta robusta y confiable para la identificación y análisis de convocatorias de ayudas.

2.2. Competencias técnicas y metodológicas requeridas

Para la implementación de la solución de este proyecto, es necesario estar capacitado en una serie de competencias técnicas, tanto en el ámbito del desarrollo de software como en relación a diferentes técnicas de Inteligencia Artificial Generativa, Agentes inteligentes y frameworks específicos.

- **Desarrollo de sistemas de captura de datos basados en web scraping:** Se requiere conocimiento y experiencia en herramientas y frameworks de scraping para la extracción de datos en múltiples entornos web, como son el caso de BeautifulSoup y Selenium.
- **Desarrollo de sistemas de procesamiento de datos:** Conocimiento en implementación de pipelines de extracción, transformación y carga de datos. Conocimiento de las principales herramientas de tratamiento de datos dentro del ecosistema Python (Numpy, Pandas, sqlalchemy).
- **Implementación de sistemas basados en Inteligencia Artificial Generativa:** Es necesaria una base de Procesamiento de Lenguaje Natural, que permita entender el funcionamiento de los Grandes Modelos del Lenguaje y las diferentes aplicaciones que se pueden construir con éstos. Se requiere, además, conocimiento de las diferentes opciones

a emplear en cuanto a modelos del lenguaje, arquitecturas de uso, y vectorización de texto mediante embeddings (Prompt engineering, Retrieval Augmented Generation, Data Extraction, etc).

- **Frameworks de orquestación de soluciones basadas en LLMs y Agentes Inteligentes:** Partiendo del punto anterior, es necesario el uso de frameworks específicos de IA generativa, los cuales permiten simplificar y normalizar el uso de estas herramientas. Frameworks como LangChain y LanGgraph permiten implementar soluciones estandarizadas y funcionales con cualquier tipo de arquitectura, desde fuentes de datos a uso de diferentes modelos del lenguaje. También permiten la construcción de grafos de procesamiento donde se pueden implementar diferentes agentes, cada uno con una función específica.
- **Bases de datos relacionales y vectoriales:** Se requiere conocimientos en diseño y gestión de bases de datos relacionales para el almacenamiento y la información estructurada generada por el sistema. Así mismo, se requieren conocimientos en bases de datos vectoriales (FAISS, Choma, etc) para el almacenamiento y consulta de los diferentes documentos, y sus embeddings asociados.
- **Implementación de interfaz gráfica:** Capacidad para implementar una interfaz gráfica que permita el uso de la aplicación mas allá de una consola de comandos.

Estas competencias son fundamentales para abordar los objetivos de este proyecto, y obtener como resultado un sistema robusto y eficiente que sea capaz de cumplir las necesidades establecidas.

3. Diseño del sistema

La solución desarrollada está estructurada en tres bloques principales: El sistema de extracción, el módulo de procesamiento, y la aplicación de consulta. Estos tres módulos trabajan en conjunto para formar un sistema automatizable que extraiga información a partir de diferentes fuentes de convocatorias, las procese debidamente, y las incluya en el asistente conversacional de consulta.

3.1. Arquitectura general

La arquitectura general del sistema se compone de los siguientes módulos principales:

- **Módulo de Extracción de Datos:** La principal funcionalidad de este módulo es la búsqueda, en cada uno de los portales de convocatorias incluidos en el sistema, de diferentes convocatorias de ayudas disponibles. Para cada ayuda encontrada, el sistema

identifica diferentes fuentes de datos necesarias: Texto resumen de la convocatoria, ficha técnica, y documentos (generalmente pdf) asociados a la convocatoria o a sus bases. Este módulo se basa principalmente en herramientas de web scraping, específicamente BeautifulSoup para la interpretación de código HTML, y Selenium para la navegación por sitios web dinámicos. Además de la extracción, también implementa una pequeña sección de preprocesamiento, para construir diferentes documentos de formato markdown para el texto plano extraído del sitio web, así como la descarga correcta de los documentos pdf, y la organización de estos documentos.

- **Módulo de Procesamiento de Datos:** Este módulo implementa un flujo de procesamiento basado en técnicas de procesamiento de datos estandar, combinadas con técnicas basadas en IA generativa para la extracción de un conjunto de parámetros específicos para cada ayuda. Internamente se han empleado diferentes técnicas implementadas desde el framework LangChain: Prompt Engineering para la construcción de prompts robustos de extracción de datos, cadenas de procesamiento de LangChain para la implementación de pipelines, y técnicas de RAG para la extracción de información de documentos mas extensos.
- **Módulo de Construcción de Bases de Datos:** Este módulo parte de los datos de convocatorias en crudo y de los datos estructurados extraídos del módulo anterior, y los integra en las diferentes bases de datos del sistema. En este caso, la integración de los datos estructurados se realiza en una base de datos SQL, mientras que los documentos pasan por un proceso de segmentación en chunks, generación de embeddings y integración en una base de datos vectorial.
- **Sistema de Recuperación de Información (RAG multiagente):** Esta aplicación consiste en una arquitectura RAG multiagente avanzada, donde dispone de la base de datos vectorial y la base de datos SQL como fuentes documentales, a las cuales puede acceder, realizar búsquedas y extraer información contextual en base a las consultas realizadas. Está construida desde los frameworks LangChain y LlanGraph y permite enlazar un Gran Modelo del Lenguaje a las fuentes de datos.
- **Interfaz de Usuario:** Se ha desarrollado una interfaz de usuario sencilla empleando la librería Streamlit, la cual permite levantar en pocas líneas de código una aplicación gráfica a modo de chatbot, e interactuar con el RAG multiagente.

3.2. Flujo de trabajo del sistema

La solución planteada está diseñada para que se pueda ejecutar de forma eficiente siguiendo un flujo de trabajo, integrando todo el proceso ETL que supone la búsqueda de convocatorias con la herramienta de consulta. Los pasos clave del flujo de trabajo son los siguientes:

- **Extracción de datos:** El módulo de extracción emplea web scraping para la búsqueda y descarga de las diferentes ayudas encontradas, siguiendo un algoritmo de búsqueda específico para cada portal de ayudas, ya que éste está ligado al formato de la página web.
- **Procesamiento de datos:** A partir de los datos en crudo se generan dos flujos de trabajo: El primero consiste en la extracción de parámetros y variables específicas de las fuentes de cada ayuda, mientras que el segundo realiza un preprocesado de los documentos, los divide en chunks y genera sus embeddings asociados.
- **Carga en las bases de datos:** A partir de los datos procesados, se actualizan tanto la base de datos SQL como la vectorial, además de generar los ficheros binarios necesarios para la aplicación RAG.
- **Integración de la aplicación RAG con las bases de datos:** Una vez han sido actualizadas las bases de datos, la aplicación RAG ya tiene acceso al contenido actualizado.

La solución está planteada de forma que se ejecuten periódicamente los tres primeros pasos para mantener actualizadas las bases de datos.

4. Tecnologías y herramientas

En esta sección se describen las herramientas y tecnologías seleccionadas para la implementación del sistema y el desarrollo de cada módulo. Las tecnologías empleadas han sido seleccionadas en base a su eficacia, su posición dentro de cada grupo de tecnologías específicas que se emplean en el mercado, y su facilidad para implementar en un entorno local con pocos recursos.

4.1. Tecnologías seleccionadas

A continuación, se detallan las tecnologías clave empleadas:

- **Python:** Se ha seleccionado Python como lenguaje de desarrollo principal por razones obvias: Es el lenguaje de programación mas empleado en el campo de la Ciencia de Datos

y la Inteligencia Artificial. Esto implica que la gran mayoría de herramientas, frameworks y librerías de estos ámbitos están disponibles en Python. Esto, junto a la simplicidad que presenta el propio lenguaje, facilita y simplifica el desarrollo de cualquier solución de esta índole.

- **Beautifulsoup y Selenium:** Para la extracción de datos de sitios web, se utilizan BeautifulSoup y Selenium. BeautifulSoup permite analizar y manipular la estructura HTML de las páginas, mientras que Selenium se emplea para interactuar con sitios web que requieren carga dinámica de contenido. Juntos, estos paquetes permiten implementar un sistema de extracción robusto, capaz de navegar y extraer información relevante de manera eficiente.
- **SQLite:** SQLite es un sistema de gestión de bases de datos relacional embebido, de código abierto, que implementa una base de datos transaccional autocontenida, sin necesidad de un servidor dedicado. Se caracteriza por su ligereza, portabilidad y facilidad de integración en aplicaciones de escritorio, móviles y sistemas embebidos. SQLite almacena los datos en un único archivo en disco, lo que simplifica la gestión y el despliegue, mientras que soporta operaciones ACID, garantizando la integridad y fiabilidad de los datos. Su arquitectura libre de configuración lo convierte en una solución óptima para aplicaciones que requieren una base de datos local con bajo consumo de recursos y alta eficiencia en operaciones de lectura y escritura.
- **FAISS (Facebook AI Similarity Search):** es una librería de código abierto desarrollada por Meta AI, diseñada para realizar búsquedas eficientes de vectores de alta dimensión, optimizando la recuperación de similitudes en grandes volúmenes de datos. FAISS permite construir índices que soportan tanto búsquedas exactas como aproximadas, utilizando técnicas avanzadas de cuantización y particionamiento, lo que la hace especialmente adecuada para aplicaciones de recuperación de información. Gracias a su arquitectura optimizada para CPU y GPU, FAISS ofrece un alto rendimiento en tareas de búsqueda a gran escala, permitiendo gestionar millones o incluso miles de millones de vectores con tiempos de respuesta reducidos y consumo eficiente de recursos.
- **FAISS - BM25:** La extensión de FAISS para soporte de modelos de recuperación basados en términos, como BM25 (Best Matching 25), permite combinar las capacidades de búsqueda vectorial de alta dimensión con técnicas clásicas de recuperación de información basadas en el modelo probabilístico BM25. Esta integración posibilita la indexación y búsqueda eficiente de documentos textuales, aplicando BM25 sobre representaciones de términos discretos mediante mecanismos optimizados dentro del entorno de FAISS. De esta manera, se obtiene una solución híbrida que aprovecha la eficiencia y escalabilidad de

FAISS para búsquedas aproximadas, al tiempo que incorpora un modelo estadístico robusto y ampliamente probado para la recuperación de documentos basada en la relevancia por frecuencia de término e inverso de frecuencia documental (TF-IDF mejorado). Esta capacidad permite diseñar sistemas de búsqueda más versátiles y precisos en escenarios donde se requiere combinar búsquedas semánticas con búsquedas léxicas tradicionales.

- **LangChain:** Es un framework de código abierto orientado a la creación de aplicaciones que integra Grandes Modelos del Lenguaje con fuentes de datos externas, herramientas y flujos de trabajo complejos. Su arquitectura modular permite orquestar cadenas de llamadas a modelos, agentes, memorias y herramientas externas, facilitando la construcción de aplicaciones conversacionales, asistentes virtuales, sistemas de pregunta-respuesta sobre documentos y soluciones de razonamiento avanzado. LangChain proporciona abstracciones de alto nivel para gestionar la interacción dinámica entre modelos, usuarios y datos, soportando integraciones con bases de datos vectoriales, motores de búsqueda, APIs y sistemas de recuperación de información. Esto permite desarrollar soluciones contextuales, reactivas y personalizadas que aprovechan el potencial de los LLMs combinándolos con lógica de negocio, flujos condicionales y datos propios de la organización.
- **LanGraph:** Es una extensión de código abierto para LangChain que permite modelar y ejecutar flujos conversacionales complejos mediante grafos de estado dinámicos. Basado en el paradigma de máquinas de estado y grafos dirigidos, LangGraph proporciona una abstracción flexible para definir nodos que encapsulan agentes, cadenas de modelos de lenguaje, funciones o cualquier unidad lógica, permitiendo controlar de manera explícita las transiciones entre estados en función de la entrada, el contexto o condiciones personalizadas. Esta capacidad resulta especialmente útil para construir aplicaciones conversacionales multi-turno, flujos de trabajo ramificados, agentes colaborativos y sistemas de toma de decisiones basados en LLMs. Además, LangGraph permite la reutilización, trazabilidad y depuración de flujos, facilitando el desarrollo de sistemas robustos, auditables y adaptables a escenarios empresariales complejos donde se requiere control granular del comportamiento conversacional y de los procesos impulsados por inteligencia artificial.
- **Azure AI Studio:** Es una plataforma integrada dentro del ecosistema Microsoft Azure, orientada a facilitar el desarrollo, despliegue y gestión de aplicaciones basadas en inteligencia artificial generativa. Proporciona un entorno unificado para construir soluciones personalizadas combinando modelos de lenguaje, visión, voz y decisiones, integrados con datos corporativos y flujos de negocio. Azure AI Studio permite a las organizaciones acelerar el ciclo de vida de desarrollo de aplicaciones de IA generativa mediante herramientas visuales, asistentes guiados y flujos predefinidos, al tiempo que asegura la gobernanza,

seguridad y cumplimiento normativo mediante políticas de Azure. Además, ofrece capacidades avanzadas para la orquestación de flujos de trabajo de IA, integración con Azure OpenAI Service, Azure Machine Learning, y conectividad con fuentes de datos empresariales, permitiendo desarrollar soluciones escalables, seguras y adaptadas a necesidades específicas de negocio.

Dentro de Azure AI Studio se han empleado los siguientes modelos:

- **o3-mini:** Es un modelo de razonamiento ligero y rentable optimizado para tareas de programación, matemáticas y ciencia, que ofrece tres niveles de esfuerzo de razonamiento (bajo, medio y alto) y soporta funcionalidades avanzadas como Structured Outputs y llamada de funciones.
- **text-embedding-3-large:** Es el modelo de incrustaciones de texto de siguiente generación de OpenAI, que genera vectores de hasta 3072 dimensiones para representar conceptos en texto y código.
- **Streamlit:** Es un framework de código abierto orientado al desarrollo ágil de interfaces web interactivas para aplicaciones de ciencia de datos, machine learning y visualización de información. Diseñado con un enfoque declarativo y minimalista, Streamlit permite a los desarrolladores crear aplicaciones web funcionales utilizando exclusivamente Python, sin necesidad de conocimientos avanzados en desarrollo frontend. La plataforma facilita la integración directa con librerías populares como Pandas, Matplotlib, Plotly o TensorFlow, permitiendo desplegar dashboards interactivos, formularios y visualizaciones en tiempo real con mínima complejidad. Además, Streamlit soporta actualizaciones reactivas y dinámicas del contenido en función de la interacción del usuario, habilitando la creación de prototipos rápidos, demostradores y herramientas de toma de decisiones basadas en datos de manera eficiente y escalable.

4.2. Implementación del módulo de extracción de datos

Este módulo ha sido el primero en desarrollarse, y su función principal es la búsqueda y recopilación de datos de plataformas en diversas plataformas. Está diseñado con una arquitectura modular, en la cual se pueden integrar diferentes módulos asociados a cada fuente de convocatorias.

- **Fuentes de datos:** En principio, se ha implementado el módulo asociado a la plataforma CDTI, pudiendo integrarse diferentes módulos posteriormente.

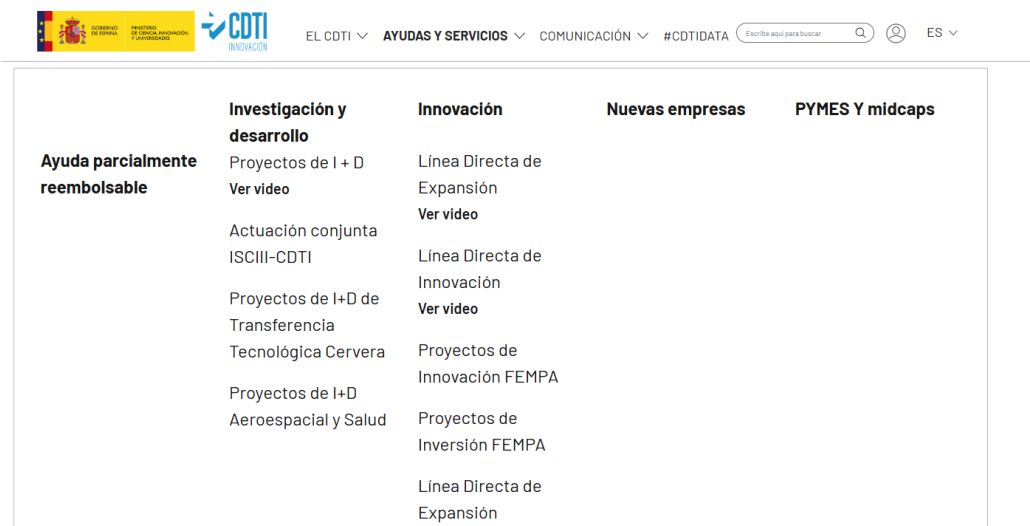
- **Extracción de datos y documentos:** Para la extracción de datos intrínsecos de los sitios web, se emplea la combinación de BeautifulSoup con Selenium. Mientras que BeautifulSoup permite la búsqueda y extracción de información a partir de las etiquetas HTML, Selenium permite interactuar con páginas que requieren carga dinámica, como botones, tablas, o bloques de texto cargados mediante Javascript. Adicionalmente, se identifican posibles enlaces a documentación en formato PDF.
- **Preprocesamiento de datos:** Para cada página asociada a una convocatoria, se extraen diferentes bloques de texto o datos, los cuales son sometidos a una limpieza inicial, eliminando duplicados, normalizando el formato e integrando diferentes bloques para formar un documento de texto en formato Markdown.

4.2.1. Implementación del módulo específico para CDTI

El módulo asociado a CDTI sigue el siguiente flujo de trabajo:

- En primer lugar se identifica la Matriz de ayudas. El portal web de CDTI presenta una tabla con diferentes ayudas disponibles según diferentes categorías. Como primer paso, el módulo extrae esa matriz de ayudas junto a los enlaces correspondientes.
- Para cada ayuda disponible en la matriz de ayudas, El módulo analiza su página principal, e identifica los siguientes elementos:
 - Bloque de resumen de la ayuda.
 - Ficha técnica.
 - Enlaces a documentación.
 - Posibles subpáginas.
- En el caso que existan subpáginas, entra en ellas y repite el proceso de extracción.
- En el caso que existan enlaces a documentación, descarga el documento en el directorio local correspondiente mediante el comando Start-BitsTransfer de Powershell. Tras haber testeado diferentes métodos de descarga desde Python, esta ha sido la única solución desde Windows que permitía extraer el documento pdf correctamente.
- Una vez identificados todos los bloques de texto necesarios, pasan por un preprocesado inicial para generar tres documentos en formato markdown:
 - Description: Documento con el texto resumen y/o introductorio de la ayuda.
 - Card: Ficha técnica de la ayuda.

- Metadata: Enlaces a documentación.



The screenshot shows the CDTI website header with logos for the Spanish Government, Ministry of Economic Affairs, and CDTI. Below the header is a navigation bar with links for 'AYUDAS Y SERVICIOS', 'COMUNICACIÓN', and '#CDTIDATA'. A search bar is also present. The main content area displays a table with four columns: 'Ayuda parcialmente reembolsable', 'Investigación y desarrollo', 'Innovación', 'Nuevas empresas', and 'PYMES Y midcaps'. The table lists various aid programs and their corresponding actions, such as 'Ver video' or 'Linea Directa de Expansión'.

	Investigación y desarrollo	Innovación	Nuevas empresas	PYMES Y midcaps
Ayuda parcialmente reembolsable	Proyectos de I + D	Linea Directa de Expansión		
	Ver video	Ver video		
	Actuación conjunta ISCIII-CDTI	Linea Directa de Innovación		
	Proyectos de I+D de Transferencia Tecnológica Cervera	Ver video		
	Proyectos de I+D Aeroespacial y Salud	Proyectos de Innovación FEMPA		
		Proyectos de Inversión FEMPA		
		Linea Directa de Expansión		

Figura 3.1: Matriz de ayudas del CDTI

Tras la ejecución del módulo, obtenemos un directorio local para cada ayuda con los tres documentos markdown y los pdf asociados a la ayuda.

4.3. Implementación del módulo de procesamiento de datos

La funcionalidad principal de éste módulo es la extracción de determinados valores o parámetros específicos de cada ayuda, con el objetivo de construir una base de datos estructurada con esa información.

Para ello, se ha diseñado una solución basada en IA Generativa que permite analizar los diferentes documentos asociado a la convocatoria de ayuda y extraer los siguientes parámetros:

- **organismo:** Entidad que da la subvención.
- **nombre:** Título de la ayuda o subvención.
- **linea:** Modalidades de la subvención.
- **fecha_inicio:** Fecha de inicio del plazo para presentar la solicitud.
- **fecha_fin:** Fecha de fin del plazo para presentar la solicitud.
- **objetivo:** Objetivo general de la ayuda o actuación.
- **beneficiarios:** Beneficiarios de la ayuda o actuación.

- **anio:** Año de la convocatoria.
- **area:** Clasificación de la ayuda en una de las siguientes áreas: I+D, Innovación, Inversión o Internacional.
- **presupuesto_minimo:** Valor del presupuesto mínimo.
- **presupuesto_maximo:** Valor del presupuesto máximo.
- **duración_mínima:** Duración mínima de la ayuda.
- **duración_máxima:** Duración máxima de la ayuda.
- **intensidad_del_prestamo:** Información sobre la intensidad del préstamo.
- **tipo_financiacion:** Tipo de la ayuda.
- **link_ficha_tecnica:** URL de la página principal de la ayuda.
- **link_convocatoria:** URL de los documentos PDF asociados a la ayuda.

De la misma manera que el módulo de extracción de datos, este módulo ha sido diseñado de forma que exista un submódulo asociado a cada portal de ayudas específico. Esto es debido a que cada portal tiene su propia arquitectura y formato de distribución de la información. En el caso de CDTI, los datos a extraer se encuentran tanto en la ficha técnica como en el PDF de la convocatoria.

4.3.1. Implementación del módulo específico para CDTI

El módulo asociado a CDTI sigue el siguiente flujo de trabajo para cada ayuda:

- Se cargan los documentos markdown como texto plano y se concatenan para tener un único documento. También se carga por otro lado el pdf a texto plano con el loader estándar de LangChain.
- Ese documento pasa por una cadena de LangChain con un prompt de extracción de un subconjunto de las variables anteriores. La cadena genera directamente un JSON con las variables y su contenido.
- Por otro lado, se genera otra cadena con una arquitectura de RAG dinámico” donde a partir de otro conjunto de prompts, se extrae el resto de variables, también en formato JSON.

- Finalmente, se concatenan ambos JSON para tener el registro completo de la ayuda.

El módulo se ejecuta para cada una de las ayudas disponibles de CDTI, y finalmente se concatenan para construir un DataFrame de Pandas, el cual se almacena en local como fichero CSV.

4.4. Implementación del módulo de construcción de las bases de datos

La finalidad de este módulo es la carga de los datos obtenidos para cada ayuda al conjunto de bases de datos que conforman la solución. Estas bases de datos son:

- Una base de datos relacional, donde se almacena una la tabla **aids_table**, con los campos previamente descritos para cada ayuda.
- Una base de datos vectorial, donde se almacenan los documentos pertinentes de cada una de las ayudas.
- Adicionalmente, se emplea un directorio local para almacenar ficheros binarios necesarios para la aplicación.

Aunque ya se ha realizado procesamiento de datos en el módulo anterior, en este módulo se realizan algunas transformaciones para preparar las fuentes de datos para el RAG multiagente.

4.4.1. Construcción de la base de datos SQL

En este caso, consiste simplemente en la creación de la tabla y la subida del csv a ésta. Posteriormente, mediante librerías de Python como SQLAlchemy, podemos acceder a esa base de datos y extraer la tabla como un dataframe de Pandas.

4.4.2. Construcción de la base de datos vectorial

Para la construcción de la base de datos vectorial y partiendo de los documentos markdown y PDF, es necesario un flujo de procesamiento:

- En primer lugar se cargan desde LangChain los documentos de cada ayuda, usando sus módulos específicos para lectura de markdown y Pdf.
- Esos documentos son divididos en chunks, a partir de los siguientes parámetros:

- **Chunk size:** Es el tamaño, en tokens o caracteres, en que se fragmenta un documento para su indexación en bases de datos vectoriales, permitiendo una representación semántica eficiente. Un tamaño mayor aporta más contexto pero puede reducir precisión, mientras que tamaños pequeños mejoran el enfoque pero pueden perder información relevante.
 - **Chunk overlap:** El chunk overlap es el solapamiento intencional entre chunks consecutivos. Se define como el número de caracteres o tokens que se repiten al inicio de cada nuevo chunk respecto al anterior. Este solapamiento es crucial para evitar pérdidas de contexto en los límites de cada fragmento, asegurando que la transición entre secciones del texto no genere huecos semánticos.
- Cada uno de los chunks de cada documento es procesado por un modelo de generación de embeddings, que transforma el contenido textual en un vector numérico de alta dimensión. Este vector captura la representación semántica del chunk, permitiendo que el sistema comprenda y compare su significado con otros textos dentro del espacio vectorial.
 - Estos embeddings son posteriormente almacenados en la base de datos vectorial para facilitar búsquedas semánticas rápidas y precisas.
 - Adicionalmente, se almacena como fichero binario esa matriz de embeddings asociada al conjunto de documentos. Esto es necesario para la implementación de algunas de las técnicas empleadas en el RAG multiagente, ya que el procesado de embeddings requiere un tiempo de procesado y un coste asociado.

4.5. Implementación del RAG multiagente

El sistema de recuperación de información es el núcleo de esta solución. Consiste en un asistente conversacional que recibe consultas en lenguaje natural y tiene la capacidad de interpretar la consulta, decidir que fuente de datos es la mas acertada, extraer información y generar una respuesta precisa. El sistema está formado por dos agentes principales: El Agente SQL y el Agente RAG.

4.5.1. Agente SQL

Este agente consiste en un sistema de recuperación a partir de una base de datos SQL. Aunque desde LangChain es relativamente simple, internamente sigue el siguiente flujo:

- El usuario introduce una consulta en lenguaje natural.

- El LLM interpreta la consulta y genera una consulta SQL a partir de las necesidades del usuario.
- El sistema ejecuta la consulta sobre la base de datos, y obtiene como resultado un conjunto de datos.
- Ese conjunto de datos, la query SQL generada y la consulta inicial en lenguaje natural son pasados como entrada al LLM, el cual genera una respuesta en lenguaje natural.

4.5.2. Agente RAG

Este agente se basa en un sistema Retrieval-Augmented Generation (RAG), el cual combina técnicas de recuperación de información y generación de lenguaje natural para mejorar la precisión y contextualización de las respuestas generadas por los modelos de lenguaje. El proceso comienza con la recepción de una consulta en lenguaje natural, que se utiliza para realizar una búsqueda semántica en una base de datos vectorial, obteniendo documentos, o fragmentos de éstos relevantes. Estos resultados recuperados son luego proporcionados como contexto en la entrada del LLM, que utiliza dicha información para generar una respuesta enriquecida, fundamentada y específica.

En el agente se han incorporado una serie de mejoras que se consideran punteras dentro del estado del arte:

- **BM25:** Best Matching 25 es una técnica clásica de recuperación de información basada en el modelo probabilístico de espacio vectorial, que calcula la relevancia de documentos en función de la frecuencia de aparición de términos (TF) y la inversa de la frecuencia documental (IDF). Al utilizar BM25 como motor de recuperación dentro de un flujo RAG, las consultas en lenguaje natural son transformadas en términos clave que BM25 utiliza para recuperar documentos o fragmentos con coincidencias léxicas precisas. Esta aproximación es especialmente efectiva cuando se requiere alta exactitud en la correspondencia de términos o en dominios con vocabulario técnico específico, asegurando que las evidencias recuperadas contengan explícitamente los términos consultados. Posteriormente, estos documentos son integrados como contexto en el modelo generativo, mejorando la fundamentación factual de las respuestas generadas.
- **Búsqueda híbrida:** La búsqueda híbrida en sistemas RAG combina de manera complementaria técnicas de recuperación basadas en búsqueda vectorial semántica y búsqueda clásica basada en términos (keyword search, BM25, etc). Este enfoque busca aprovechar las fortalezas de ambos métodos: mientras la búsqueda semántica permite recuperar documentos relevantes aunque no coincidan literalmente las palabras clave, la búsqueda basada

en términos asegura precisión léxica y cobertura de coincidencias exactas. En un flujo de RAG híbrido, ambas búsquedas se ejecutan en paralelo o de manera combinada, fusionando los resultados para seleccionar los documentos o fragmentos más relevantes. Estos se integran como contexto enriquecido en el modelo generativo, mejorando la precisión, cobertura y robustez frente a ambigüedades, términos específicos o consultas complejas.

- **Reranking:** El reranking es una etapa crítica que permite optimizar la selección de los documentos recuperados antes de ser entregados al modelo generativo. Tras ejecutar la búsqueda inicial, ya sea semántica, basada en términos o híbrida, se obtiene un conjunto preliminar de documentos candidatos. El proceso de reranking consiste en reordenar estos documentos aplicando modelos de clasificación más sofisticados, que pueden estar basados en aprendizaje profundo, heurísticas personalizadas o combinaciones de puntuaciones semánticas y léxicas. Este proceso evalúa con mayor precisión la relevancia contextual de cada documento respecto a la consulta original, priorizando aquellos que aporten mayor valor semántico o evidencias más directas. Al mejorar la calidad y pertinencia de los documentos suministrados al LLM, el reranking incrementa la exactitud, coherencia y fundamentación de las respuestas generadas en el flujo RAG.

En este caso, se ha optado por emplear **FlashRerank**, una técnica de reranking eficiente que utiliza modelos ligeros tipo cross-encoder optimizados para reordenar grandes volúmenes de documentos candidatos, evaluando de manera cruzada cada fragmento con la consulta para obtener puntuaciones de relevancia más precisas. Diseñado para entornos de alta demanda, permite mejorar la selección de documentos relevantes de forma rápida y escalable, optimizando la calidad y pertinencia del contexto entregado al modelo generativo.

El flujo del agente RAG quedaría de la siguiente forma:

- El usuario introduce una consulta en lenguaje natural.
- El sistema realiza una búsqueda híbrida de documentos relevantes:
 - En la búsqueda semántica, el sistema calcula los embeddings de la consulta, y busca en la base de datos vectorial los K documentos similares.
 - en la búsqueda por keywords, se aplica BM25: Se calculan los parámetros de palabras clave mediante TF-IDF a la consulta, y se comparan con los parámetros almacenados del conjunto de documentos. Se buscan los K documentos mas similares y se devuelven.

- Mediante un proceso de ponderación, que se encuentra al 50 % de peso en ambas modalidades de búsqueda, se devuelve un conjunto de documentos similares.
- Una vez obtenidos los documentos similares, se aplica el algoritmo de reranking. Este vuelve a evaluar la relación entre cada documento y la consulta, asignando nuevas puntuaciones de relevancia.
- Finalmente, se proporciona los documentos relevantes obtenidos junto a la consulta al LLM, el cual genera una respuesta.

Adicionalmente, se incluye un proceso paralelo al flujo principal, donde se extraen los títulos y páginas de los documentos relevantes, se formatea como texto markdown y se añaden a la respuesta.

4.5.3. Grafo orquestador de los agentes

Mediante el uso combinado de LangChain y Langraph, se construye un grafo principal que permite orquestar los dos agentes. La creación de un grafo con LangGraph se basa en la definición de nodos, que representan agentes, funciones o pasos de procesamiento, y aristas que definen las transiciones entre nodos en función del estado o la lógica de control. Cada nodo puede incorporar agentes de LangChain, herramientas externas o lógica personalizada, permitiendo modelar flujos flexibles, iterativos o condicionales. Una vez definidos, el grafo se ejecuta gestionando el paso de estados, controlando la persistencia, la gestión de errores y la toma de decisiones de forma explícita y trazable, habilitando la implementación de workflows de IA modulares, reutilizables y controlados.

El sistema implementado en esta solución se compone de un flujo en el que, a partir de una consulta, pasa por un elemento llamado enrutador o Router. Este router no es mas que una cadena de LangChain que emplea un LLM para decidir si la consulta la debe responder el agente SQL o el agente RAG.

4.6. Implementación de la interfaz gráfica

Finalmente, se implementa mediante Streamlit una interfaz gráfica sencilla a modo de chat, donde se pueden enviar mensajes directamente al sistema. Streamlit permite generar con un único script de Python una aplicación grafica de manera rápida y sencilla, la cual puede ser posteriormente desplegada con rapidez.



Figura 3.2: Interfaz gráfica con Streamlit

5. Resultados

Tras la implementación de cada uno de los módulos lo que obtenemos es una solución completa con las siguientes características:

- Una aplicación de asistencia conversacional, donde conversas con una IA que tiene acceso tanto a los documentos asociados a las ayudas como a los datos estructurados generados por el procesamiento.
- Cuando el usuario realiza una consulta, el sistema decide internamente si el contexto que necesita lo extrae de la base de datos SQL o de la vectorial. En cada caso, realiza una búsqueda de información y en base a esos datos genera una respuesta. En el caso de que la consulta sea a la base de datos vectorial, el sistema te devuelve una lista de fuentes empleadas.
- El propio sistema integra las pipelines de extracción y procesamiento de datos, con los que se puede mantener actualizada la aplicación realizando ejecuciones de manera periódica.
- El sistema está construido con una arquitectura modular, por lo que se pueden integrar nuevas fuentes de datos (portales de ayudas) con facilidad.

Capítulo 4

Conclusiones y trabajo futuro

1. Conclusiones y trabajo futuro

1.1. Resumen del trabajo realizado

En este proyecto se ha desarrollado una solución completa para la gestión, análisis y explotación de convocatorias de ayudas e empresas, donde se han combinado técnicas de extracción de datos web, técnicas de procesamiento del lenguaje natural mediante Grandes Modelos del Lenguaje, y de explotación de la información mediante agentes inteligentes. Los resultados obtenidos muestran que el sistema es capaz de explotar la información obtenida mediante consultas en lenguaje natural, realizando búsquedas tanto sobre fuentes documentales como datos estructurados. Se destacan las siguientes características:

- La solución sigue una arquitectura basada en módulos, por lo que es fácilmente escalable para incluir nuevas fuentes de datos.
- La combinación de bases de datos relacionales y vectoriales permiten realizar búsquedas eficientes, dotando al sistema de capacidad para responder consultas de índole mas cuantitativa sobre el conjunto de ayudas.
- El empleo de los frameworks LangChain y LanGraph permiten diseñar un sistema agnóstico a los modelos del lenguaje utilizados, así como a las bases de datos. La solución permite reemplazar los diferentes servicios empleados por otros de la misma índole y seguir funcionando sin problemas.
- El uso de LLMs como servicio desde Azure, y las bases de datos empleadas permiten la ejecución de la solución en entornos con bajos recursos, sin necesidad de una GPU para la ejecución de las llamadas a los modelos y embeddings.

1.2. Lecciones aprendidas

El desarrollo de esta solución ha supuesto un aprendizaje en diversos aspectos:

- Se ha mejorado la comprensión y el uso de técnicas basadas en IA generativa y grandes modelos del lenguaje.
- Se ha obtenido una imagen global de como funcionan los frameworks LangChain y LanGraph, y como estos se interconectan con distintas opciones de fuentes de datos y modelos.
- El desarrollo ha estado marcado por el uso de buenas prácticas en el ámbito del desarrollo de software, avanzando mas allá de los típicos notebooks empleados en ciencia de datos y diseñando una aplicación completa.
- Se han revisado y testeado diferentes metodologías y arquitecturas dentro del Retrieval Augmented Generation, así como el funcionamiento de los sistemas basados en agentes inteligentes.

1.3. Mejoras y trabajo futuro

La solución planteada en el proyecto es funcional y da unos resultados acordes a las necesidades del proyecto. Sin embargo, hay ciertos puntos en los que podría mejorarse esta solución a nivel global:

- Actualmente está limitada a un único portal de ayudas (CDTI). Sin embargo, el diseño modular de la solución permitiría integrar fácilmente nuevas fuentes de datos en un futuro.
- La infraestructura de bases de datos es sencilla, de forma que pueda emplearse en local sin demasiada dificultad y sin necesidad de recursos computacionales elevados. Como propuesta de mejora, se sugiere el empleo de servicios mas completos y profesionales, como por ejemplo PostgreSQL o similares para los datos estructurados, y otras opciones en cuanto a bases de datos vectoriales, como PGVector o Qdrant.
- En cuanto al uso de los frameworks LangChain y LanGraph, comentar que el desarrollo empleado y las funcionalidades implementadas son funcionales, pero se contempla la posibilidad de emplear un diseño y una arquitectura mejorada, en base a los frecuentes cambios y mejoras que se implementan en estos frameworks.
- Del mismo modo, recientemente está cobrando fuerza en el campo de la IA generativa el concepto de MCP (Model Context Protocol). Es un estándar abierto que funciona como un "USB-C" para modelos de IA, permitiendo conectar de forma uniforme cualquier modelo a

múltiples fuentes de datos y herramientas externas. En lugar de crear integraciones punto a punto entre cada modelo y cada servicio, MCP reduce la complejidad permitiendo a los modelos y herramientas comunicarse mediante un único protocolo común. Utiliza JSON-RPC 2.0 para enviar solicitudes y recibir respuestas; así, las aplicaciones IA (clientes MCP) invocan funciones en servidores MCP que adaptan APIs, bases de datos o sistemas de archivos. Los desarrolladores pueden desplegar servidores MCP para servicios como GitHub, Slack o una base de datos, y los agentes de IA eligen y llaman a estas funciones durante la generación de contenido.

- Como último aporte en cuanto a mejoras, queda comentar que la solución actualmente no es 100 % reproducible, por lo que aplicar ciertas capas de abstracción en diferentes módulos, y complementar con una arquitectura de contenedores daría lugar a una aplicación agóstica del sistema operativo y completamente reproducible.

Con estas líneas de desarrollo, la solución puede seguir en evolución, con el objetivo de ganar robustez y mayor adaptabilidad a las necesidades del problema inicialmente planteado en el proyecto.

Bibliografía

- [1] Andalucía trade, <https://www.andaluciatrade.es/financiacion-empresarial/incentivos-para-las-empresas/>.
- [2] Centro para el desarrollo tecnológico y la innovación, <https://www.cdti.es/>.
- [3] Fandit, <https://fandit.es/>.
- [4] Grupo spri, <https://www.spri.eus/es>.
- [5] Opengrants/opengrants.io.
- [6] Portal de ayudas del ministerio para la transformación digital y de la función pública, <https://portalayudas.digital.gob.es/paginas/convocatorias-ayudas.aspx>.
- [7] Sociedad para el desarrollo ragional de cantabria, <https://ayudas.sodercan.es/ayudas>.
- [8] Felipe Almeida and Geraldo Xexéo. Word embeddings: A survey, 2023.
- [9] Anthropic. Contextual retrieval, 2024. Accessed: 2025-03-29.
- [10] M. El Asikri, S. Krit, H. Chaib, M. Kabrane, H. Ouadani, K. Karimi, K. Bendaouad, and H. Elbousty. Mining the web for learning ontologies: State of art and critical review. pages 1–7, 2017.
- [11] M. El Asikri, S. Krit, H. Chaib, M. Kabrane, H. Ouadani, K. Karimi, K. Bendaouad, and H. Elbousty. Mining the web for learning ontologies: State of art and critical review. pages 1–7, 2017.
- [12] M Asikri¹, S Krit, Hassan Chaib, and Krit Salah-ddine. Using web scraping in a knowledge environment to build ontologies using python and scrapy. *European Journal of Translational and Clinical Medicine*, 7:433–442, 10 2020.
- [13] Ashish Bansal. Optimizing rag with hybrid search and contextual chunking. *Journal of Engineering and Applied Sciences Technology*, pages 1–5, 08 2023.

- [14] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.
- [15] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, et al. Language models are few-shot learners, 2020.
- [16] Erika Cardenas and Leonie Monigatti. What is agentic rag. November 2024.
- [17] DeepSeek-AI, :, Xiao Bi, Deli Chen, Guanting Chen, et al. Deepseek llm: Scaling open-source language models with longtermism, 2024.
- [18] DeepSeek-AI, Aixin Liu, Bei Feng, et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model, 2024.
- [19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.
- [20] Liangping Ding, Cornelia Lawson, and Philip Shapira. Rise of generative artificial intelligence in science, 2024.
- [21] Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. Precise zero-shot dense retrieval without relevance labels, 2022.
- [22] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, et al. Generative adversarial networks, 2014.
- [23] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The llama 3 herd of models, 2024.
- [24] Rohmat Gunawan, Alam Rahmatulloh, Irfan Darmawan, and Firman Firdaus. Comparison of web scraping techniques : Regular expression, html dom and xpath. pages 283–287, 2019/03.
- [25] Thilo Hagendorff. Mapping the ethics of generative ai: A comprehensive scoping review. *Minds and Machines*, 34(4), September 2024.
- [26] Emmanouil Ikonomakis, Sotiris Kotsiantis, and V. Tampakas. Text classification using machine learning techniques. *WSEAS transactions on computers*, 4:966–974, 08 2005.
- [27] Ivan Ilin. Advanced rag techniques: an illustrated overview. *Towards AI*, 2023.
- [28] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, et al. Mistral 7b, 2023.

- [29] Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over bert, 2020.
- [30] Diksha Khurana, Aditya Koli, Kiran Khatter, and Sukhdev Singh. Natural language processing: state of the art, current trends and challenges. *Multimedia Tools and Applications*, 82(3):3713–3744, July 2022.
- [31] Koli Khurana, K Khatter, et al. Natural language processing: State of the art, current trends and challenges. 2023. Available online.
- [32] Divya Khyani, Siddhartha B S, N. Niveditha, Divya M., and Dr Y M. An interpretation of lemmatization and stemming in natural language processing. *Shanghai Ligong Daxue Xuebao/Journal of University of Shanghai for Science and Technology*, 22:350–357, 01 2021.
- [33] Deepika Kumawat and Vinesh Jain. Pos tagging approaches: A comparison. *International Journal of Computer Applications*, 118:32–38, 05 2015.
- [34] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021.
- [35] Yiheng Liu, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, othersiaming, Hao He, Antong Li, Mengshen He, Zhengliang Liu, et al. Summary of chatgpt-related research and perspective towards the future of large language models. *Meta-Radiology*, 1(2):100017, September 2023.
- [36] LlamaIndex. A new document summary index for llm-powered qa systems, 2024. Accessed: 2025-03-29.
- [37] Michael Llordes, Debasis Ganguly, Sumit Bhatia, and Chirag Agarwal. Explain like i am bm25: Interpreting a dense model’s ranked-list with a sparse approximation, 2023.
- [38] Chaimaa Lotfi, Swetha Srinivasan, Myriam Ertz, and Imen Latrous. *Web Scraping Techniques and Applications: A Literature Review*, pages 381–394. 01 2021.
- [39] Renze Lou, Kai Zhang, and Wenpeng Yin. Large language model instruction following: A survey of progresses and challenges, 2024.
- [40] Stephanie Lunn, Jia Zhu, and Monique Ross. Utilizing web scraping and natural language processing to better inform pedagogical practice. pages 1–9, 2020.

- [41] Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. Query rewriting for retrieval-augmented large language models, 2023.
- [42] K Usha Manjari, Syed Rousha, Dasi Sumanth, and J Sirisha Devi. Extractive text summarization from web pages using selenium and tf-idf algorithm. pages 648–652, 2020.
- [43] Laia Subirats Maté and Mireia Calvo González. Web scraping. *Editorial UOC.*, 2019.
- [44] R. (2018) Mitchell. Web scraping with python: Collecting more data from the modern web. o’reilly media, inc., 2018.
- [45] Martin Mortaheb, Mohammad A. Amir Khojastepour, Srimat T. Chakradhar, and Sennur Ulukus. Re-ranking the context for multimodal retrieval augmented generation, 2025.
- [46] M. Nguyen, T.Q. Nguyen, K. KC, Z. Zhang, and T. Vu. Reinforcement learning from answer reranking feedback for retrieval-augmented answer generation. In *Proceedings of Interspeech 2024*, pages 4044–4048, 2024.
- [47] Briony June Oates. Researching information systems and computing. 2005.
- [48] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, et al. Gpt-4 technical report, 2024.
- [49] OpenAI, Aaron Hurst, Adam Lerer, et al. Gpt-4o system card, 2024.
- [50] Wisam Qader, Musa M. Ameen, and Bilal Ahmed. An overview of bag of words;importance, implementation, applications, and challenges. pages 200–204, 06 2019.
- [51] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.
- [52] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, 2019. OpenAI Blog.
- [53] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2023.
- [54] Arya Roy. Recent trends in named entity recognition (ner), 2021.
- [55] Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. A systematic survey of prompt engineering in large language models: Techniques and applications, 2025.

-
- [56] Claude Sammut and Geoffrey I. Webb, editors. *TF-IDF*, pages 986–987. Springer US, Boston, MA, 2010.
- [57] Serhad Sarica and Jianxi Luo. Stopwords in technical language processing. *PLOS ONE*, 16(8):e0254937, August 2021.
- [58] Craig W. Schmidt, Varshini Reddy, Haoran Zhang, Alec Alameddine, Omri Uzan, Yuval Pinter, and Chris Tanner. Tokenization is more than compression, 2024.
- [59] Robin M. Schmidt. Recurrent neural networks (rnns): A gentle introduction and overview, 2019.
- [60] Sandeep Singh Sengar, Affan Bin Hasan, Sanjay Kumar, and Fiona Carroll. Generative artificial intelligence: A systematic review and applications, 2024.
- [61] Aditi Singh, Abul Ehtesham, Saket Kumar, and Tala Talaei Khoei. Agentic retrieval-augmented generation: A survey on agentic rag, 2025.
- [62] Ralf C. Staudemeyer and Eric Rothstein Morris. Understanding lstm – a tutorial into long short-term memory recurrent neural networks, 2019.
- [63] A.D. Stolyarov, A.V. Abramov, and V.I. Abramov. Generative artificial intelligence for business models innovation:opportunities and limitations. *Beneficium*, pages 43–51, 09 2024.
- [64] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, et al. Lamda: Language models for dialog applications, 2022.
- [65] Ivana Tomić, Ivana Jurič, Sandra Dedijer, and Savka Adamovic. Artificial intelligence in graphic design. 09 2023.
- [66] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, et al. Llama: Open and efficient foundation language models, 2023.
- [67] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, et al. Llama 2: Open foundation and fine-tuned chat models, 2023.
- [68] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.
- [69] Sriram Veturi, Saurabh Vaichal, Reshma Lal Jagadheesh, Nafis Irtiza Tripto, and Nian Yan. Rag based question-answering for contextual response prediction system, 2024.

-
- [70] Mark Wasson. Using summaries in document retrieval. In *Proceedings of the ACL-02 Workshop on Automatic Summarization - Volume 4*, AS '02, page 27–44, USA, 2002. Association for Computational Linguistics.
- [71] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.
- [72] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications, 2024.
- [73] Gokul Yenduri, Ramalingam M, Chemmalar Selvi G, Supriya Y, Gautam Srivastava, Praveen Kumar Reddy Maddikunta, Deepti Raj G, Rutvij H Jhaveri, Prabadevi B, Weizheng Wang, Athanasios V. Vasilakos, and Thippa Reddy Gadekallu. Generative pre-trained transformer: A comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions, 2023.
- [74] Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. Generate rather than retrieve: Large language models are strong context generators, 2023.
- [75] Meishan Zhang. A survey of syntactic-semantic parsing based on constituent and dependency structures, 2020.