ntic master
revolucionamos la comunicación

UNIVERSIDAD COMPLUTENSE MADRID

RAG

(Retrieval Augmented Generation)

Eduardo Fernández Carrión, PhD

Information Retrieval (IR)

Es el proceso de acceder a recursos no estructurados, generalmente documentos, con el propósito de recuperar información de forma eficiente entre grandes repositorios.

- Los sistemas IR comienzan con la **indexación** de un gran conjunto de documentos que permite **buscar** rápidamente a través de ellos y **almacenarlos** de forma eficiente.
- Todo parte de una **query** (por ejemplo, una pregunta o una serie de palabras clave), el sistema busca en su índice para encontrar documentos que sean relevantes para esa consulta.
- La relevancia se determina utilizando varios **algoritmos** y **métricas**.
- Los sistemas más avanzados pueden incluso considerar el **contexto** de la consulta o la intención del usuario para mejorar la precisión de los resultados recuperados.
- La búsqueda semántica (**semantic search**) interpreta el significado semántico detrás de las palabras en una consulta de búsqueda.
- Actualmente, no solo sirve para textos. Sirve para imágenes, audios, video, etc.

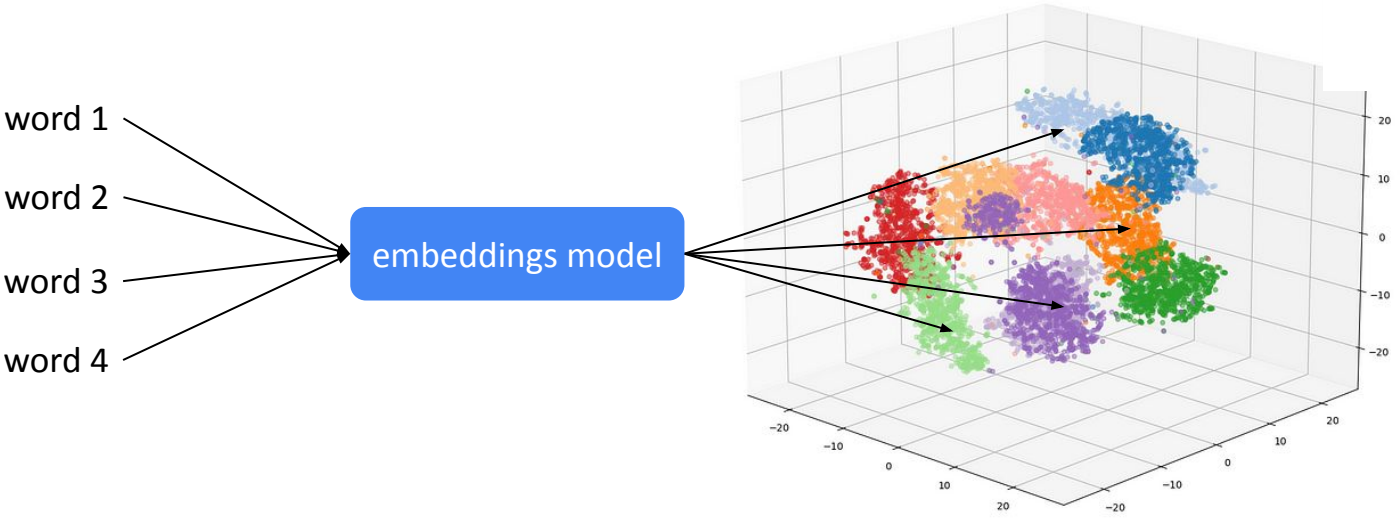
Retrieval Augmented Generation (RAG)

RAG es una **estrategia** que sirve para mejorar la capacidad de los **LLM** para generar respuestas precisas en base a un repositorio de información, generalmente, documentos.

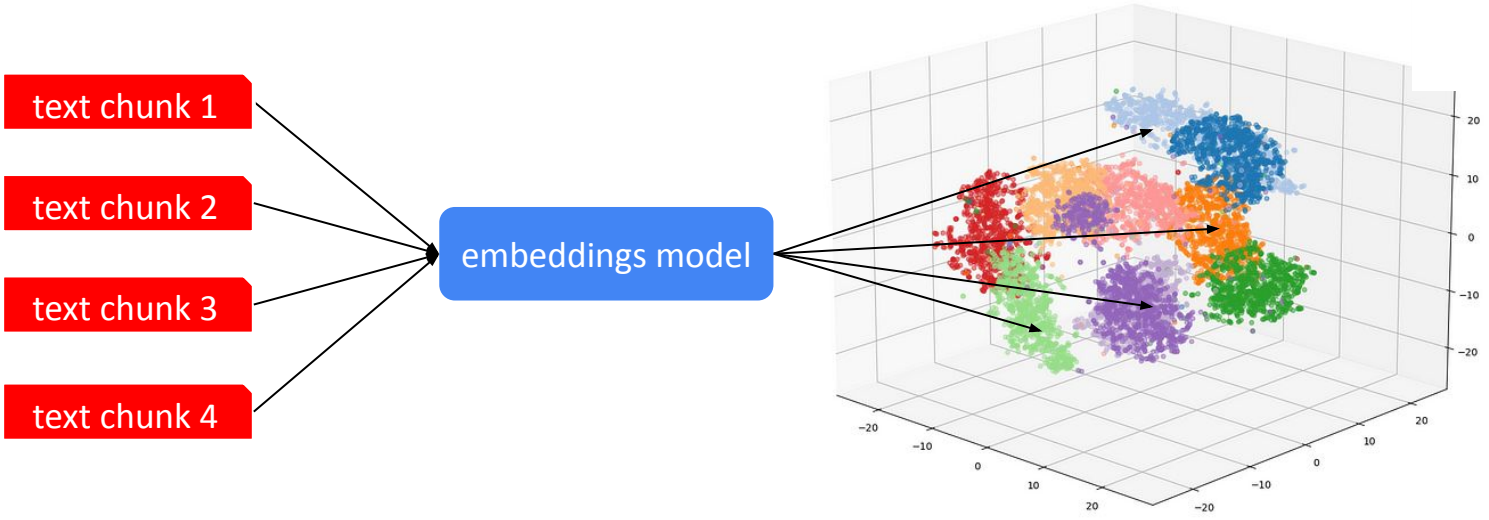
El proceso RAG se basa en 3 partes:

- **Retrieval**: En función de la **query**, se recuperan datos de un repositorio de documentos usando técnicas IR.
- **Augmentation**: Combina la información recuperada junto con un **prompt**.
- **Generation**: Esta información aumentada con el prompt es enviada a un LLM para **generar texto de salida**.

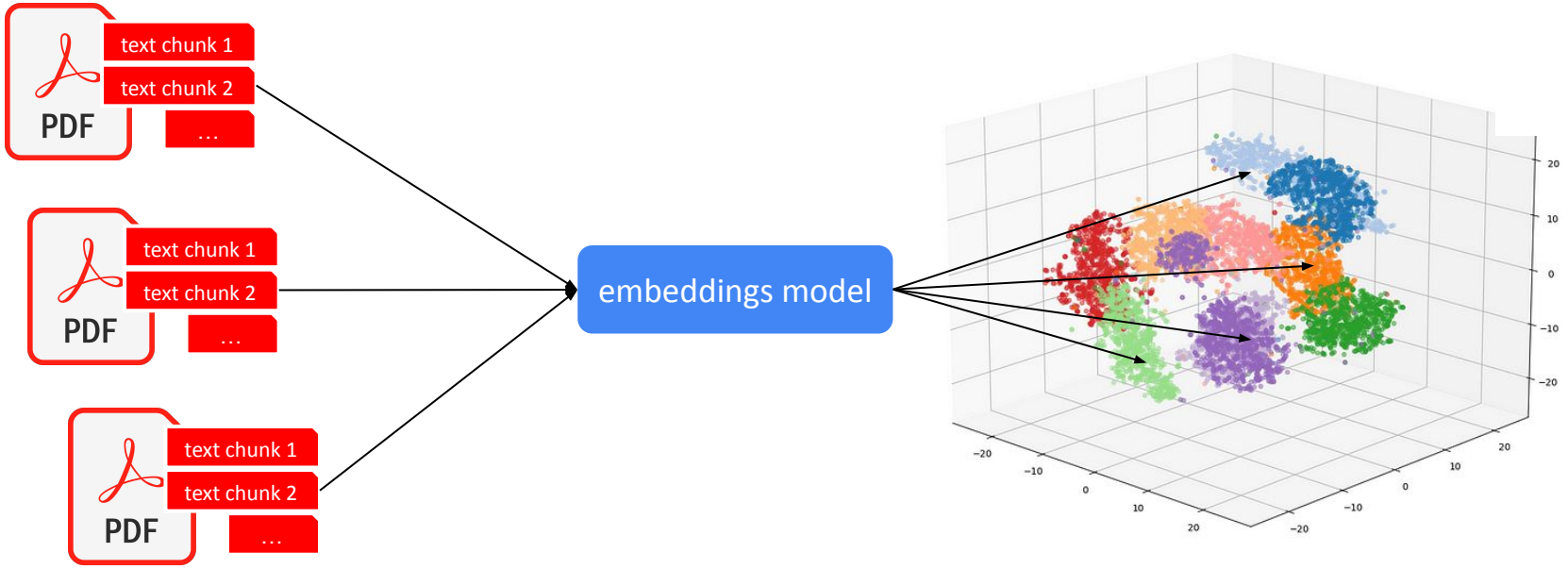
Word Embeddings



Text Embeddings



Document Embeddings

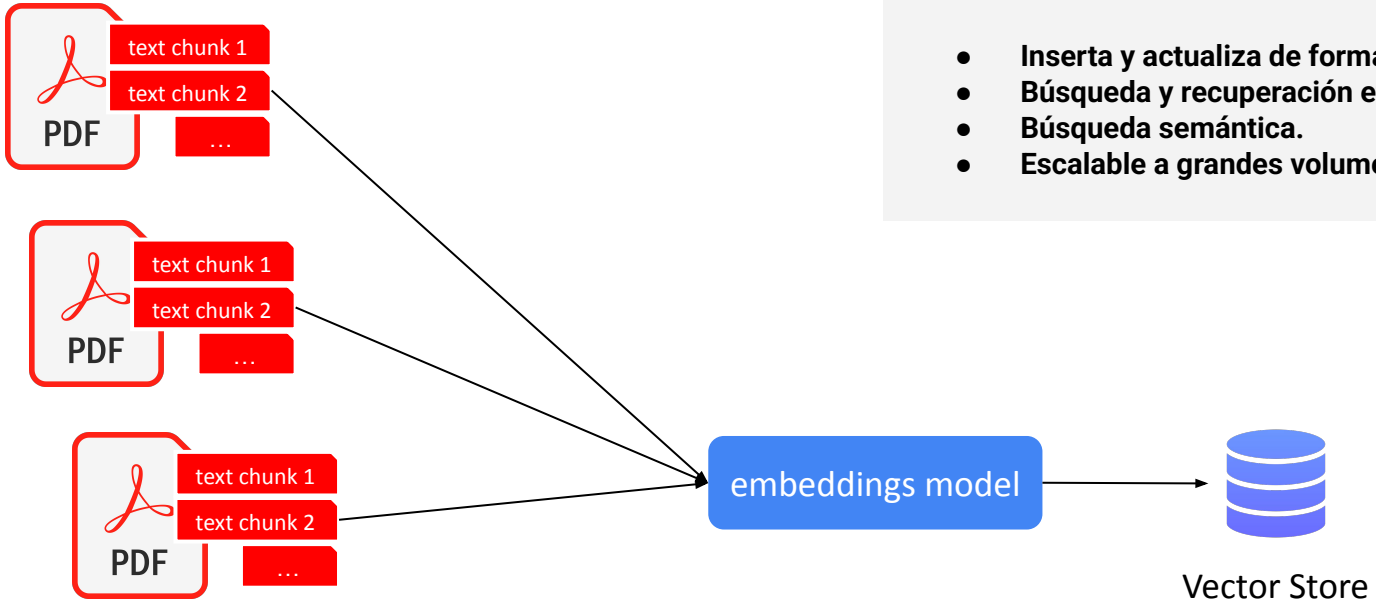


Vector Store

Un VS es una estructura de almacenamiento diseñada para guardar y gestionar vectores de embeddings.

Características:

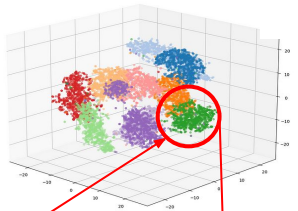
- **Inserta y actualiza de forma rápida embeddings.**
- **Búsqueda y recuperación eficiente (*indexado*).**
- **Búsqueda semántica.**
- **Escalable a grandes volumetrías.**



Retrieval information

user query

embeddings model



index

Vector Store

text chunk 2

Vector Store

text chunk 1

text chunk 2

...

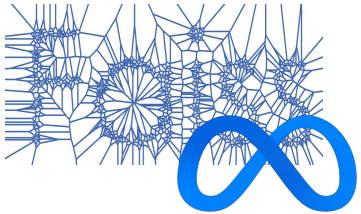
Similarity Search

What is similarity search?

Given a set of vectors x_i in dimension d, Faiss builds a data structure in RAM from it. After the structure is constructed, when given a new vector x in dimension d it performs efficiently the operation:

$$j = argmin_i \|x - x_i\|$$

where $\|\cdot\|$ is the Euclidean distance (L^2).



Algoritmos de búsqueda semántica

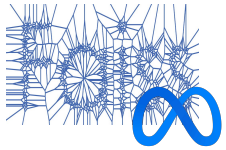
- **Índice de búsqueda exhaustiva (Flat)**: Es el método más simple y directo. Utiliza la búsqueda lineal para encontrar los vecinos más cercanos y, aunque no es el más rápido para grandes conjuntos de datos, es muy utilizado por su simplicidad y precisión completa.
- **Índice IVF (Inverted File Index)**: Este método divide el espacio de los vectores en regiones mediante un cuantificador de vectores. Los vectores se almacenan en listas invertidas asociadas a cada región, lo que permite una búsqueda más rápida al limitar la búsqueda a las regiones más prometedoras.
- **Índice LSH (Locality-Sensitive Hashing)**: Utiliza funciones hash que agrupan vectores similares en el mismo "bucket", reduciendo el espacio de búsqueda. Es particularmente útil para aplicaciones donde las aproximaciones son aceptables.
- **Índice Scalar Quantizer, Índice IVFPQ, Índice PQ (Product Quantization)**...

Marketplace de VS

Vector Stores

Chroma

Azure Cognitive Search



Weaviate

Pinecone

vespa

Embeddings Models

OpenAI
text-embedding-ada-002

1. GTE-Base (Graft Default)
2. GTE-Large
3. GTE-Small
4. E5-Small
5. MultiLingual
6. RoBERTa (2022)
7. MPNet V2
8. Scibert Science-Vocabulary Uncased
9. Longformer Base 4096
10. Distilbert Base Uncased
11. Bert Base Uncased
12. MultiLingual BERT

LLM

OpenAI

Flagship chat models Our versatile, high-intelligence flagship models.

GPT-4o
Fast, intelligent, flexible GPT model

ChatGPT-4o
GPT-4o model used in ChatGPT

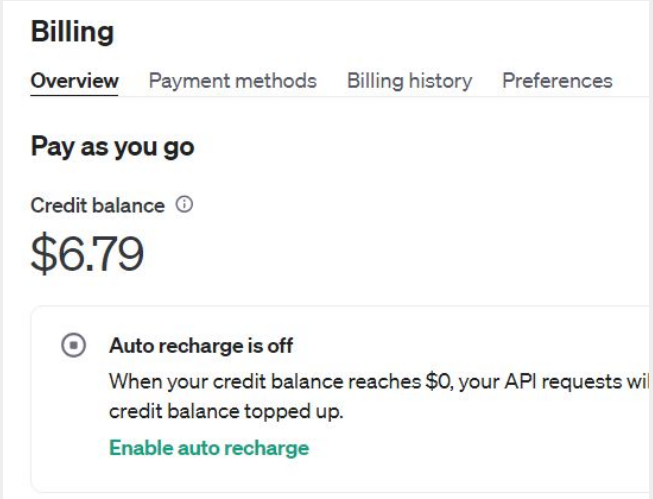
Cost-optimized models Smaller, faster models that cost less to run.

GPT-4o mini
Fast, affordable small model for focused tasks

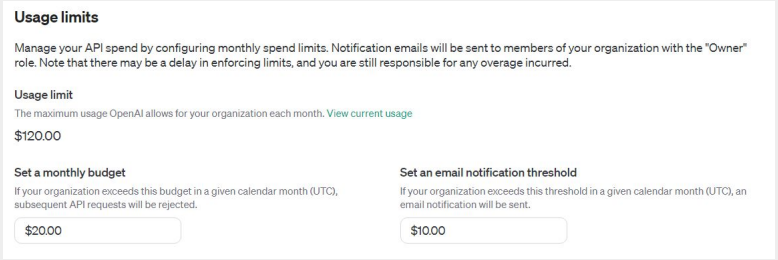
https://platform.openai.com/docs/models/overview

Creación de cuenta

Crédito



Límite de uso



OpenAI query

Test a basic API request

```python
from openai import OpenAI
client = OpenAI()

response = client.responses.create(
    model="gpt-4o",
    input="Write a one-sentence bedtime story about a unicorn."
)

print(response.output_text)
```

https://platform.openai.com/docs/libraries

api-key (*borrar después del curso)

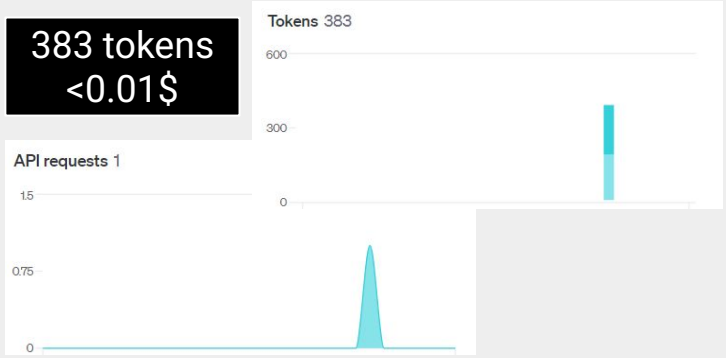
nlp-course1	sk-...GiBr	Enabled

+ Create new secret key

pricing

Las API cobran por cada *call*, y el precio de cada *call* va en función del número de tokens de entrada y de salida (que facturan de forma diferente)

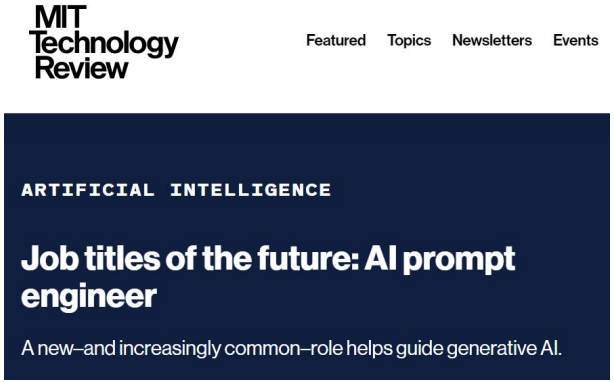
https://openai.com/api/pricing/



Prompting

Los sistemas de **GenIA** están diseñados para generar **salidas basadas en la calidad de los prompts** proporcionados.

La ingeniería de prompts ayuda a los modelos de IA generativa a comprender mejor y responder a una amplia gama de consultas, desde las más simples hasta las altamente técnicas.



Tipos de Prompting

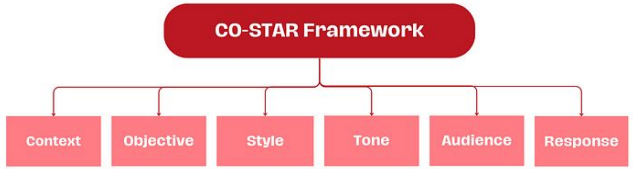
- **Zero-shot**: El modelo recibe una solicitud sin ningún ejemplo previo o contexto adicional. Basa su respuesta únicamente en la información contenida en el prompt.
- **Few-shot:** Se proporcionan uno o varios ejemplos al modelo antes de la pregunta o tarea actual. Esto ayuda al modelo a entender mejor el contexto o el estilo de respuesta deseado.
- **Chain of thought:** Este tipo de prompting implica guiar al modelo para que explique su proceso de pensamiento paso a paso antes de llegar a una conclusión → tareas complejas de **razonamiento**.

Argumentos

- **model**: El modelo de lenguaje que se utilizará para la generación. Ejemplos comunes son gpt-4, gpt-4o-mini, etc.
- **messages**: Una lista de mensajes que componen la conversación. Cada mensaje es un diccionario que incluye un rol (role) y contenido (content). Los roles pueden ser system, user, o assistant.
- **temperature**: Controla la aleatoriedad de las respuestas. Valores más bajos harán que el modelo sea más determinista, mientras que valores más altos incrementan la creatividad y variabilidad en las respuestas.
- **max_tokens**: El número máximo de tokens que se permite en la respuesta generada. Esto incluye tanto los tokens en la entrada como en la salida.
- **top_p**: También conocido como nucleus sampling. Este parámetro define el límite para la suma de las probabilidades de los tokens seleccionados. Se suele usar en combinación o como alternativa a temperature.
- **n**: El número de completions que deseas que el modelo genere para cada entrada dada. Es útil para comparar diferentes respuestas posibles.
- **frequency_penalty**: Penaliza las repeticiones en la respuesta. Un valor alto de este parámetro reducirá la repetición de palabras o frases en la salida generada.

Prompt engineering

La estructura eficaz de los mensajes es crucial para obtener respuestas óptimas de un LLM. El marco CO-STAR, creado por el equipo de Ciencia de Datos e IA de GovTech Singapore, es **una plantilla** útil para estructurar estos mensajes. Considera todos los aspectos clave que influyen en la efectividad y relevancia de la respuesta de un LLM, lo que conduce a respuestas más óptimas.



- **Contexto (C):** Se refiere a proporcionar información de fondo sobre la tarea. Esto ayuda al Modelo de Lenguaje de Gran Tamaño (LLM) a comprender el escenario específico que se está discutiendo, asegurando que su respuesta sea relevante.
- **Objetivo (O):** Define lo que se quiere que el LLM realice. Ser claro sobre el objetivo ayuda al LLM a enfocar su respuesta en cumplir ese objetivo específico.
- **Estilo (S):** Especifica el estilo de escritura que deseas que el LLM use. Podría ser el estilo de una persona famosa, o de un experto en una profesión, como un analista de negocios o un CEO. Esto guía al LLM para responder con la manera y elección de palabras alineadas con tus necesidades.
- **Tono (T):** Establece la actitud de la respuesta. Esto asegura que la respuesta del LLM resuene con el sentimiento o contexto emocional pretendido. Ejemplos son formal, humorístico, empático, entre otros.
- **Audiencia (A):** Identifica para quién está destinada la respuesta. Adaptar la respuesta del LLM a una audiencia, como expertos en un campo, principiantes, niños, etc., asegura que sea apropiada y comprensible en tu contexto requerido.
- **Respuesta (R):** Proporciona el formato de la respuesta. Esto asegura que el LLM entregue la salida en el formato exacto que necesitas para tareas posteriores. Ejemplos incluyen una lista, un JSON, un informe profesional, etc. Para la mayoría de las aplicaciones de LLM que trabajan programáticamente con las respuestas del LLM para manipulaciones posteriores, un formato de salida JSON sería ideal.

https://towardsdatascience.com/how-i-won-singapores-gpt-4-prompt-engineering-competition-34c195a93d41

Ejemplo

 # CONTEXT #
I want to advertise my company's new product. My company's name is Alpha and the product is called Beta, which is a new ultra-fast hairdryer.

 # OBJECTIVE #
Create a Facebook post for me, which aims to get people to click on the product link to purchase it.

 # STYLE #
Follow the writing style of successful companies that advertise similar products, such as Dyson.

 # TONE #
Persuasive

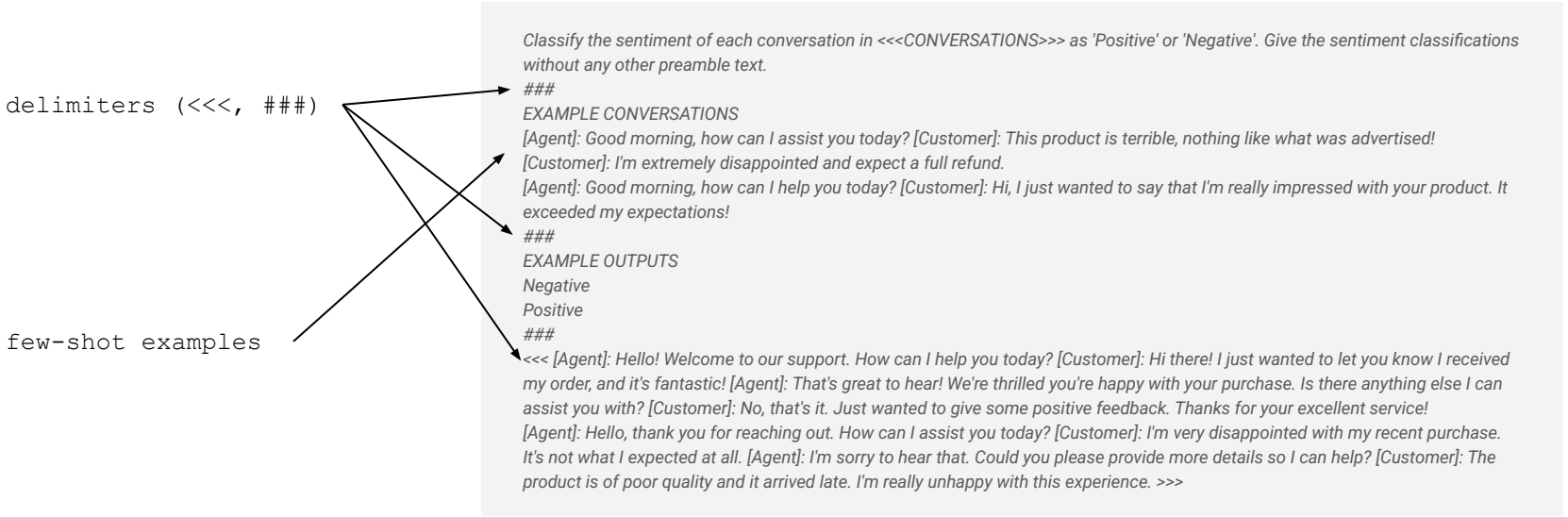
 # AUDIENCE #
My company's audience profile on Facebook is typically the older generation. Tailor your post to target what this audience typically looks out for in hair products.

 # RESPONSE #
The Facebook post, kept concise yet impactful.

Prompt delimiters

Los delimitadores son **tokens especiales** que ayudan al LLM a distinguir qué partes de tu mensaje debe considerar como una sola unidad de significado.

Esto es importante porque todo tu mensaje llega al LLM como una única secuencia larga de tokens. Los delimitadores proporcionan estructura a esta secuencia de tokens al delimitar partes específicas de tu mensaje para que sean tratadas de manera diferente.



LLM Guardrails

Los LLM poseen una función de Prompt llamada **System prompt**, que son mensajes adicionales en los que se proporcionan instrucciones sobre cómo debe comportarse el chatbot **a lo largo de toda la conversación**.

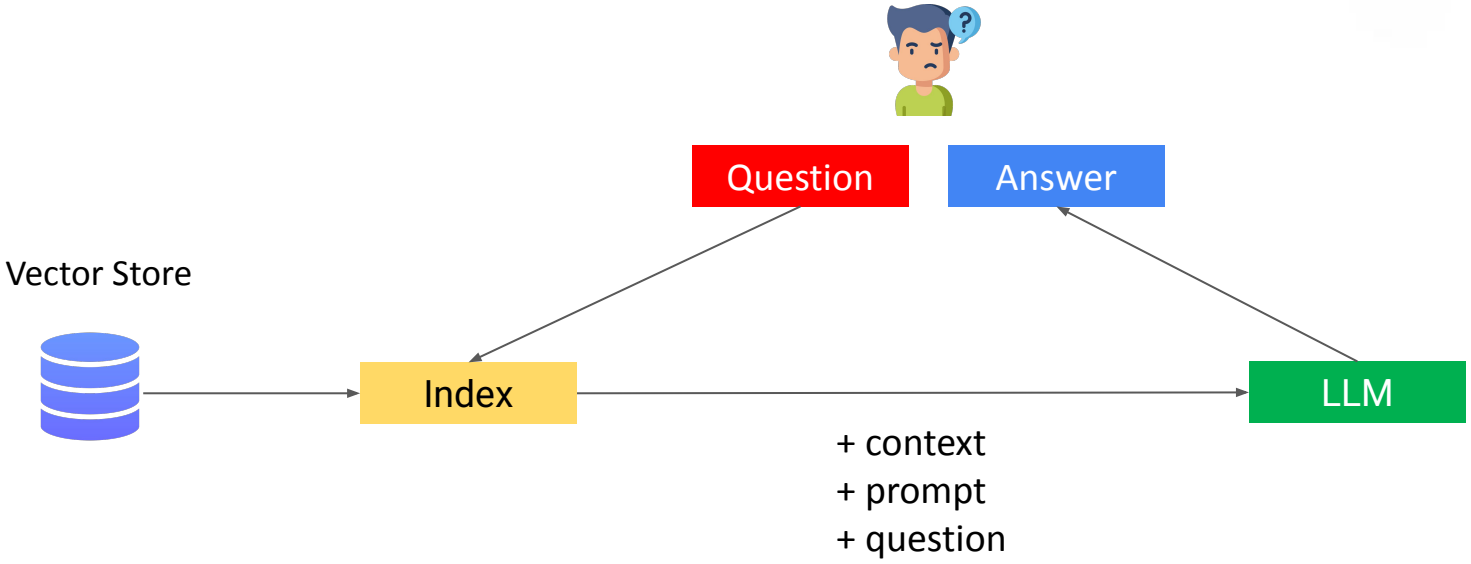
Qué deberían incluir?

- Definición de la tarea, para que el LLM siempre recuerde lo que tiene que hacer a lo largo del chat.
- Formato de salida, para que el LLM siempre recuerde **cómo debe responder**.
- Guardrails, para que el LLM siempre recuerde **cómo no debe responder**. Este punto es especialmente importante para evitar comentarios ofensivos, data leaking, trolling, etc.

Usa los "**System Prompts**" para proporcionar instrucciones que quieres que el LLM recuerde al responder durante todo el chat.

Eres un chatbot que interactúa con clientes. Responder siempre de manera informativa y educativa, enfocándose en proporcionar datos verificados y evitando especulaciones o información no confirmada. Mantener un tono neutral y respetuoso en todas las respuestas, sin expresar opiniones personales o hacer juicios de valor. No proporcionar asesoramiento legal, médico ni financiero específico. Evitar responder a preguntas que involucren datos personales sensibles o confidenciales.

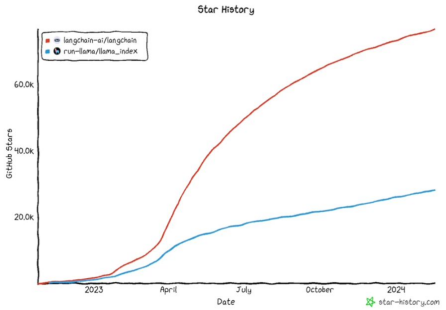
RAG workflow



retrieval | augmentation | generation

Orquestadores de soluciones LLMs

Son bibliotecas de Python diseñada para facilitar la **construcción de aplicaciones avanzadas de lenguaje natural** mediante la integración de **LLMs** con lógicas programáticas y bases de datos.



Langchain

Nació para la construcción de Arquitecturas LLM basada principalmente en el uso de cadenas de lenguaje y agentes (tools).

LlamaIndex

Nació para optimizar el acceso a datos para la construcción de RAGs de forma eficiente.

No obstante, ambas librerías tienen soluciones muy similares

LlamaIndex

LlamaIndex es un marco para construir aplicaciones de LLM con contexto aumentado. La aumentación de contexto se refiere a cualquier caso de uso que aplique LLMs sobre tus datos privados o específicos de un dominio. Algunos casos de uso populares incluyen los siguientes:

- Chatbots de Preguntas y Respuestas (RAG)
- Entendimiento y Extracción de Documentos
- Agentes Autónomos que pueden realizar investigaciones y tomar acciones

Use Cases

Prompting

Question-Answering (RAG)

Chatbots

Structured Data Extraction

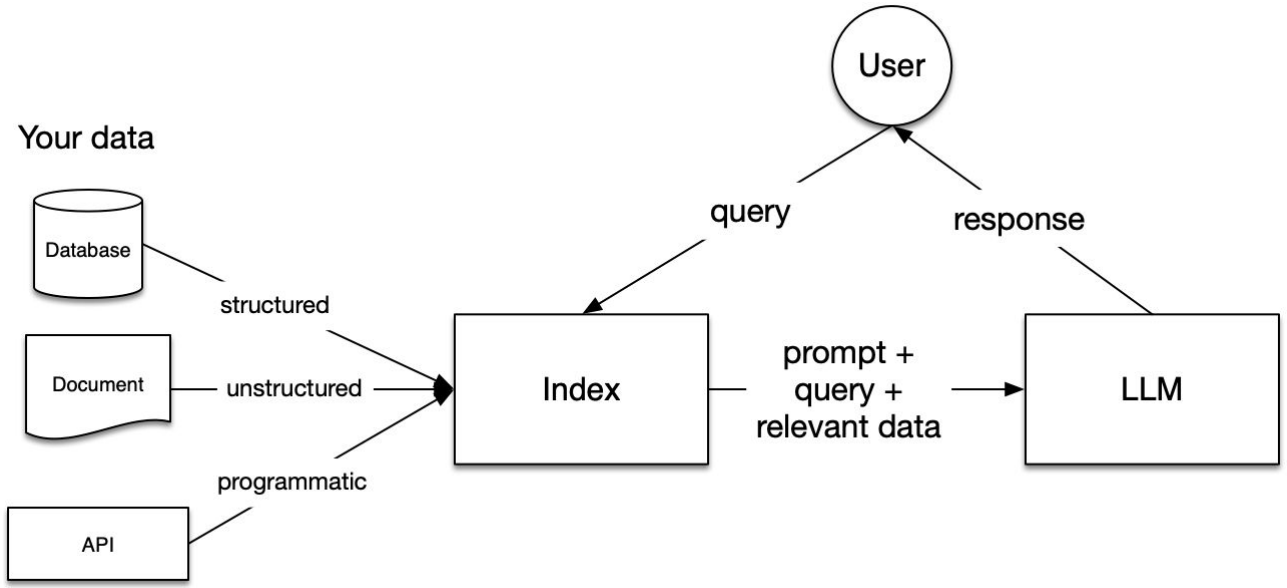
Agents

Multi-Modal Applications

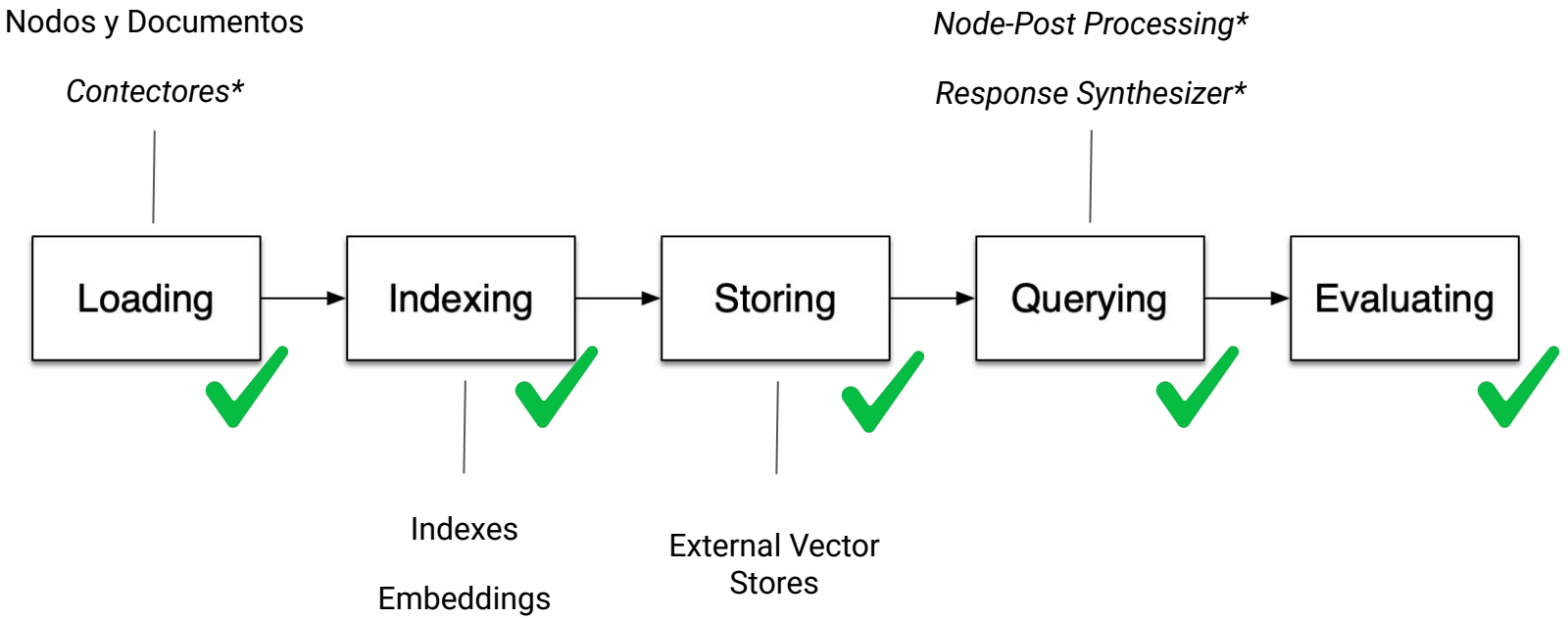
Fine-Tuning

Nota: **LlamaIndex NO es Llama**, Llama2, Llama3… Sin embargo, el nombre de LlamaIndex proviene de que, en sus comienzos, se desarrolló primero un motor optimizado para indexar datos y elaborar un motor de respuestas basadas en el LLM de Llama.

Arquitectura básica RAG



Stages



*se recomienda investigar estos conceptos en la documentación oficial de LlamaIndex

Conceptos básicos

- Un **Documento** es un contenedor genérico de cualquier fuente de datos, por ejemplo, un PDF, una salida de API o datos recuperados de una base de datos*.
- Un **Nodo** representa un "fragmento" de un Documento fuente, ya sea un fragmento de texto, una imagen u otro. Al igual que los Documentos, contienen **metadatos** e **información de relación con otros nodos**.
- Un **Index** es una estructura de datos compuesta por objetos de tipo nodo, diseñada para permitir consultas por parte de un LLM.
- Un **Retriever** es una herramienta que define la estrategia de recuperación de manera eficiente el contexto relevante de un índice cuando se le realiza una consulta.
- Un **Query Engine** es una interfaz genérica que te permite hacer preguntas a tus datos. Un Query Engine toma una consulta en lenguaje natural y devuelve una respuesta alimentada por la información de los datos.

*Un Documento siempre es un Nodo. Pero un Nodo no siempre es un Documento.

LLM-as-a-judge (evaluation)

1

2

q1, q2, …

RAG

r1, r2, …

User questions	User answers
q1	a1
q2	a2
…	…

groundtruth
question-answer

3

a1, a2, …

r1, r2, …

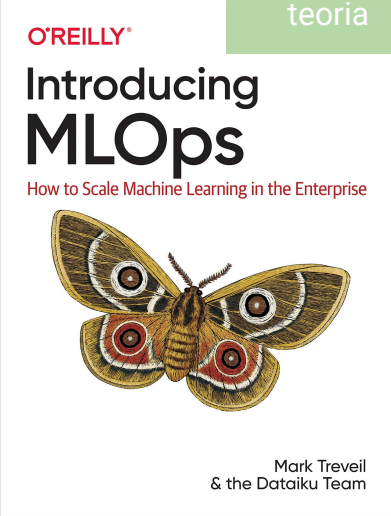
LLM Judge

Score	feedback
3	bla
5	bla
…	…

```
grading_prompt = """Evalúa la calidad de la respuesta generada por el modelo LLM en comparación con la respuesta de referencia (ground truth). A continuación se presentan los criterios para la evaluación: - **Score 1:** La respuesta del LLM es irrelevante o incorrecta en comparación con la verdad de referencia. - **Score 2:** La respuesta del LLM es algo relacionada pero tiene errores significativos o está incompleta en comparación con la verdad de referencia. - **Score 3:** La respuesta del LLM es aceptable, pero carece de precisión o detalles importantes en comparación con la verdad de referencia. - **Score 4:** La respuesta del LLM es buena y responde adecuadamente, aunque hay detalles que podrían mejorarse para alinearse más con la verdad de referencia. - **Score 5:** La respuesta del LLM es excelente, clara, completa y tan precisa como la verdad de referencia o incluso aporta detalles adicionales útiles. Proporciona un puntaje del 1 al 5 basado en los criterios anteriores y explica brevemente tu razonamiento. **Ground truth:** {ground_truth} **Respuesta del LLM:** {llm_response} **Evaluación:** """
```

LLMOps

teoria



- Es evolucionar MLOps a los LLM:
  - **Auditoría**:

Consiste en mantener un registro completo de todas las decisiones, modificaciones y procesos llevados a cabo durante el ciclo de vida del modelo, incluyendo los datos utilizados, los cambios en el código y las configuraciones del entorno. Esto asegura transparencia y facilita la trazabilidad en caso de que sea necesario revisar decisiones o resultados del modelo en el futuro.

  - **Reproducibilidad**:

La capacidad de replicar los resultados de un modelo bajo las mismas condiciones en cualquier momento.

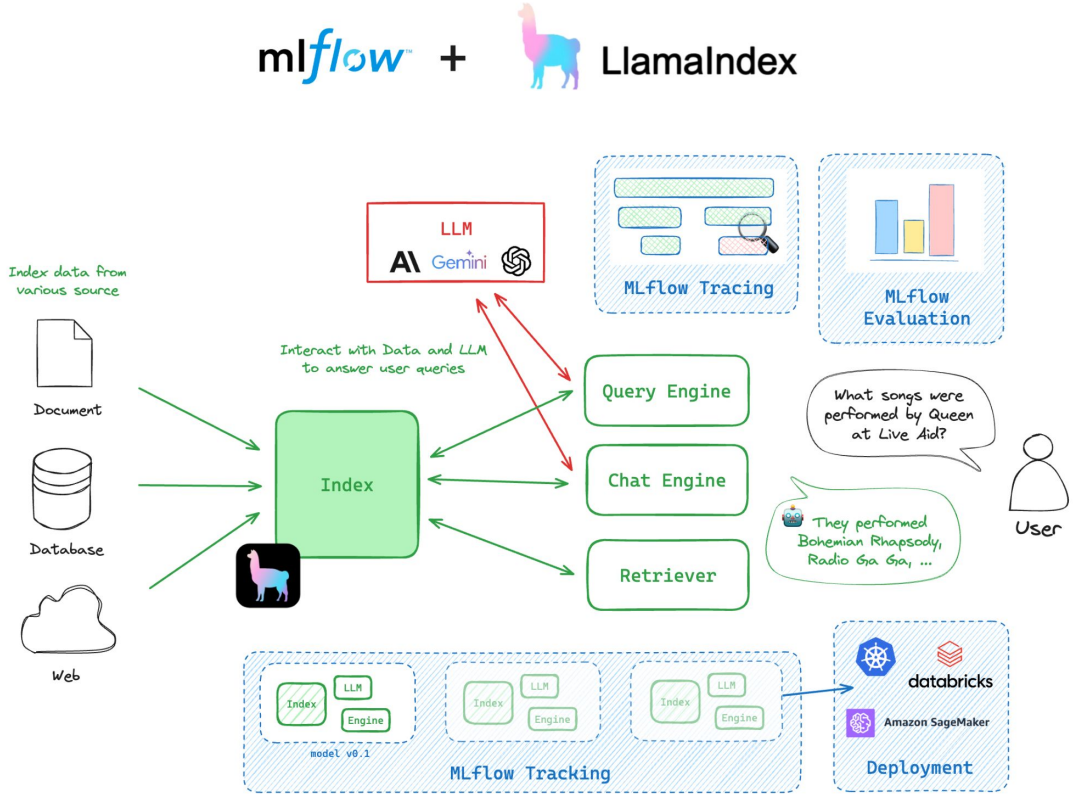
práctica

mlflow™

El **versionado** de código, datos y modelos es clave para el control de cambios y la colaboración.

No es viable **escalar** una aplicativo basado en LLM **a producción** sin una solución de **LLMOps**.

LLMOps



- LLMs
  - Tutorials and Use Case Guides for GenAI applications in MLflow
  - MLflow Tracing
  - MLflow AI Gateway for LLMs
  - LLM Evaluation
  - Prompt Engineering UI
  - **Native MLflow Flavors for LLMs**
    - MLflow Transformers Flavor
    - MLflow OpenAI Flavor
    - MLflow Sentence-Transformers Flavor
    - MLflow LangChain Flavor
    - MLflow LlamaIndex Flavor
    - MLflow DSPy Flavor
    - Explore the Native LLM Flavors
  - LLM Tracking in MLflow

https://mlflow.org/docs/latest/llms/llama-index/index.html

LLM-as-a-judge (evaluation)

groundtruth (caso de uso)

"tengo alguna garantía que cubra los humos?"	"Sí, las modalidades de los seguros Hogar Completo, Hogar Eficaz y Hogar Premium incluyen la cobertura de daños ocasionados por humo, ya sea por escapes repentinos en cocinas, sistemas de calefacción u otros aparatos eléctricos."
"qué seguros de hogar cubren las placas solares?"	"La modalidad Hogar Completo cubre el robo de placas solares en cubiertas de viviendas unifamiliares hasta 3.000 € por incidente. En la modalidad Hogar Premium, la cobertura para el robo de placas solares se amplía hasta 5.000 € por incidente."
"la vitrocerámica está incluida en alguna modalidad del seguro?"	"Sí, las modalidades Hogar Completo y Hogar Premium cubren la rotura accidental de elementos vitrocerámicos de cocina, incluyendo la reposición de la placa o cristal."

ntic master
revolucionamos la comunicación

UNIVERSIDAD COMPLUTENSE MADRID