

# RAG

(Retrieval Augmented Generation)

# 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.

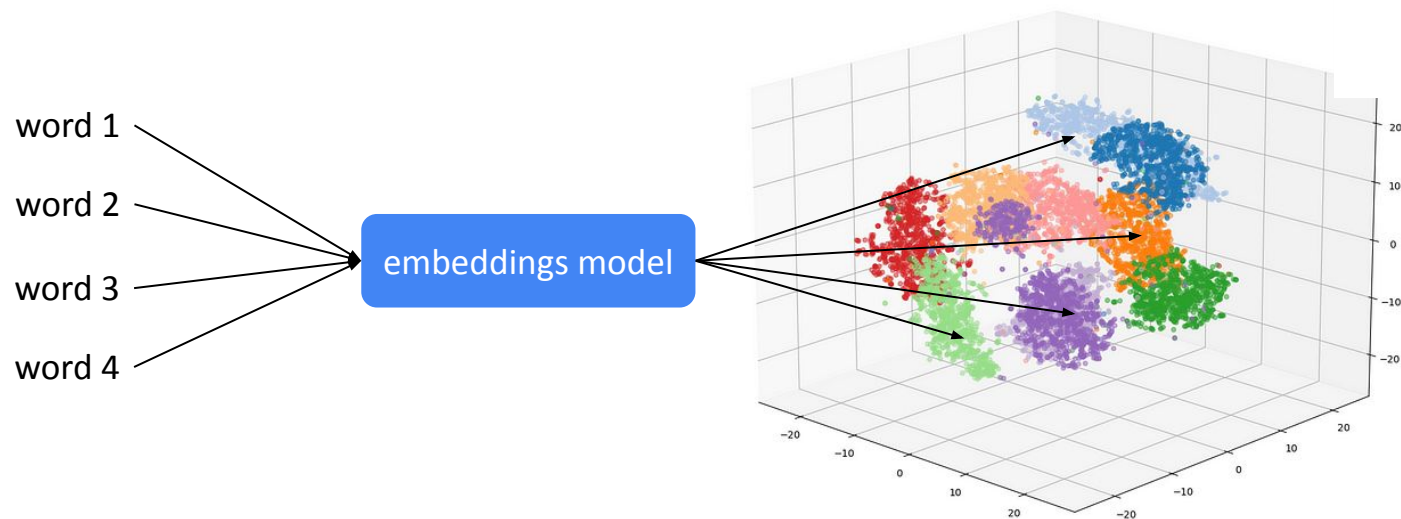


El proceso RAG se basa en 3 partes:

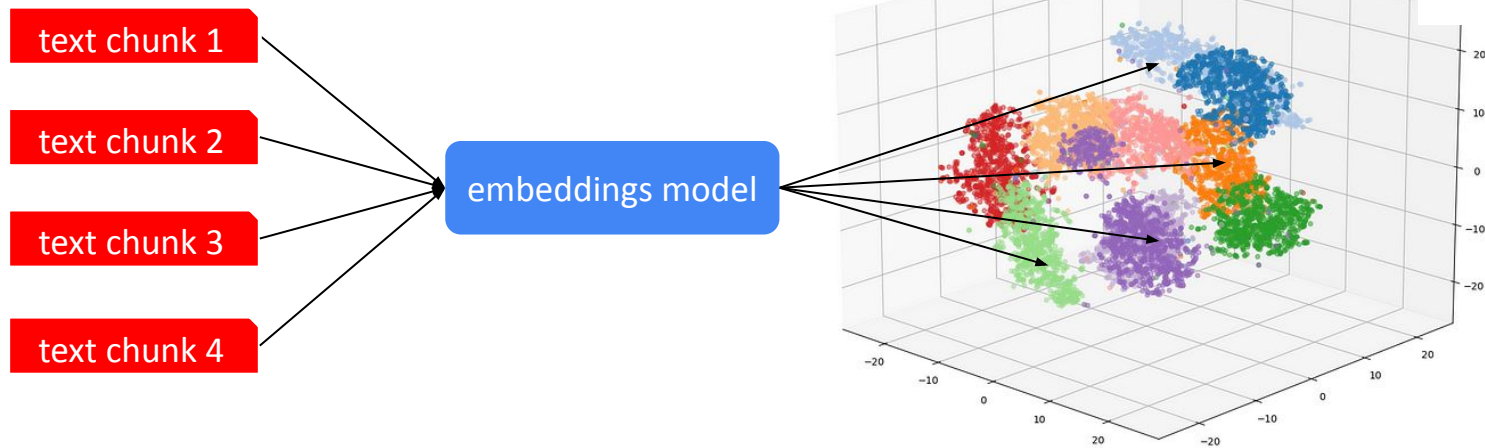
- **Retrieval**: En función de la **query**, se recuperan datos de un repositorio de documentos usando técnicas IR.
- **Augmentation**: Combina la información recuperada junto con un **prompt**.
- **Generation**: Esta información aumentada con el prompt es enviada a un LLM para **generar texto de salida**.



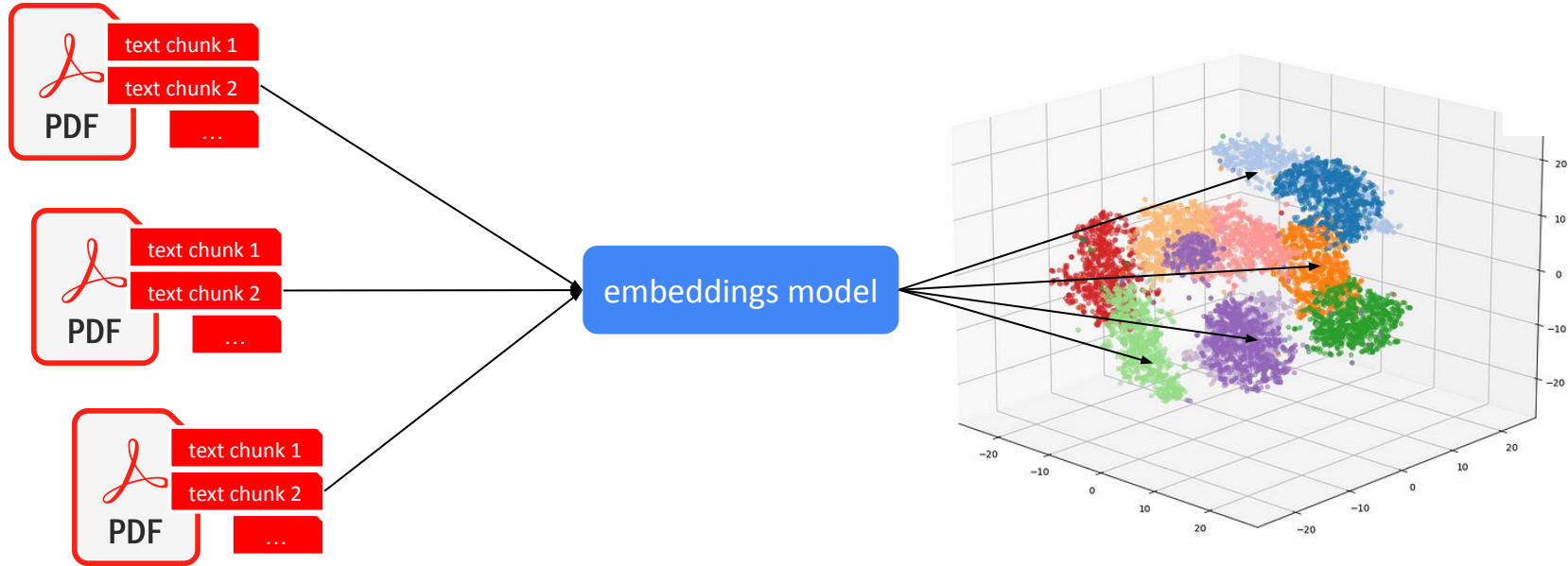
# Word Embeddings



# Text Embeddings



# Document Embeddings

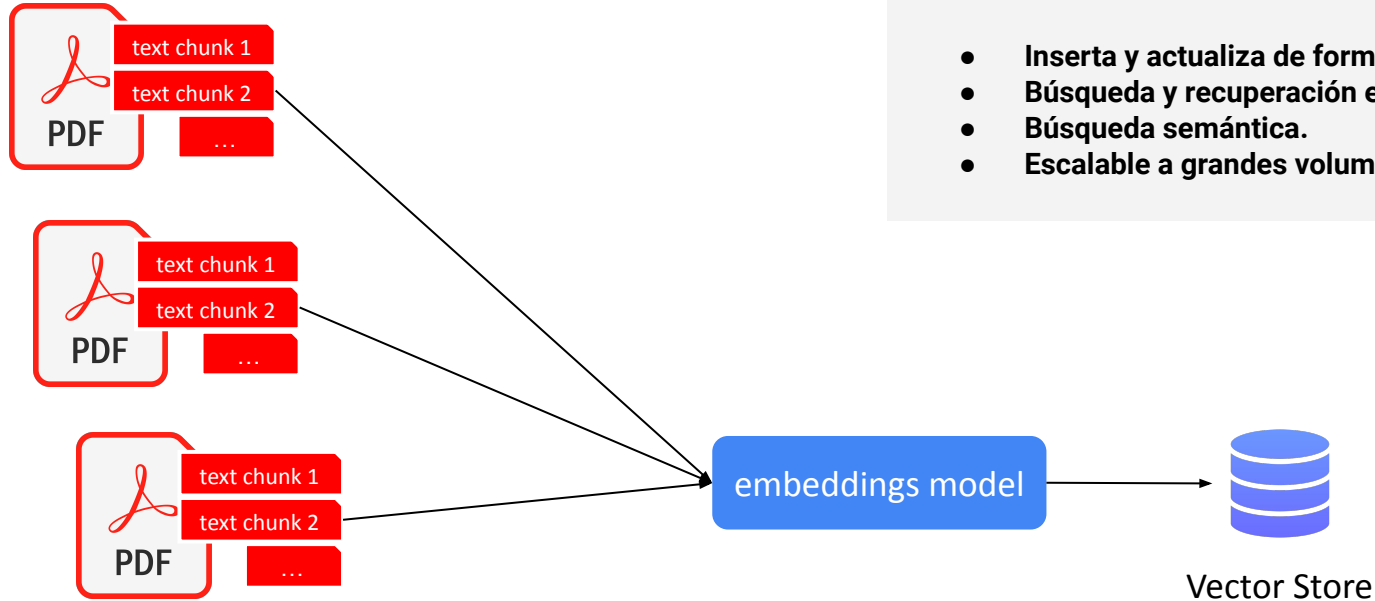


# Vector Store

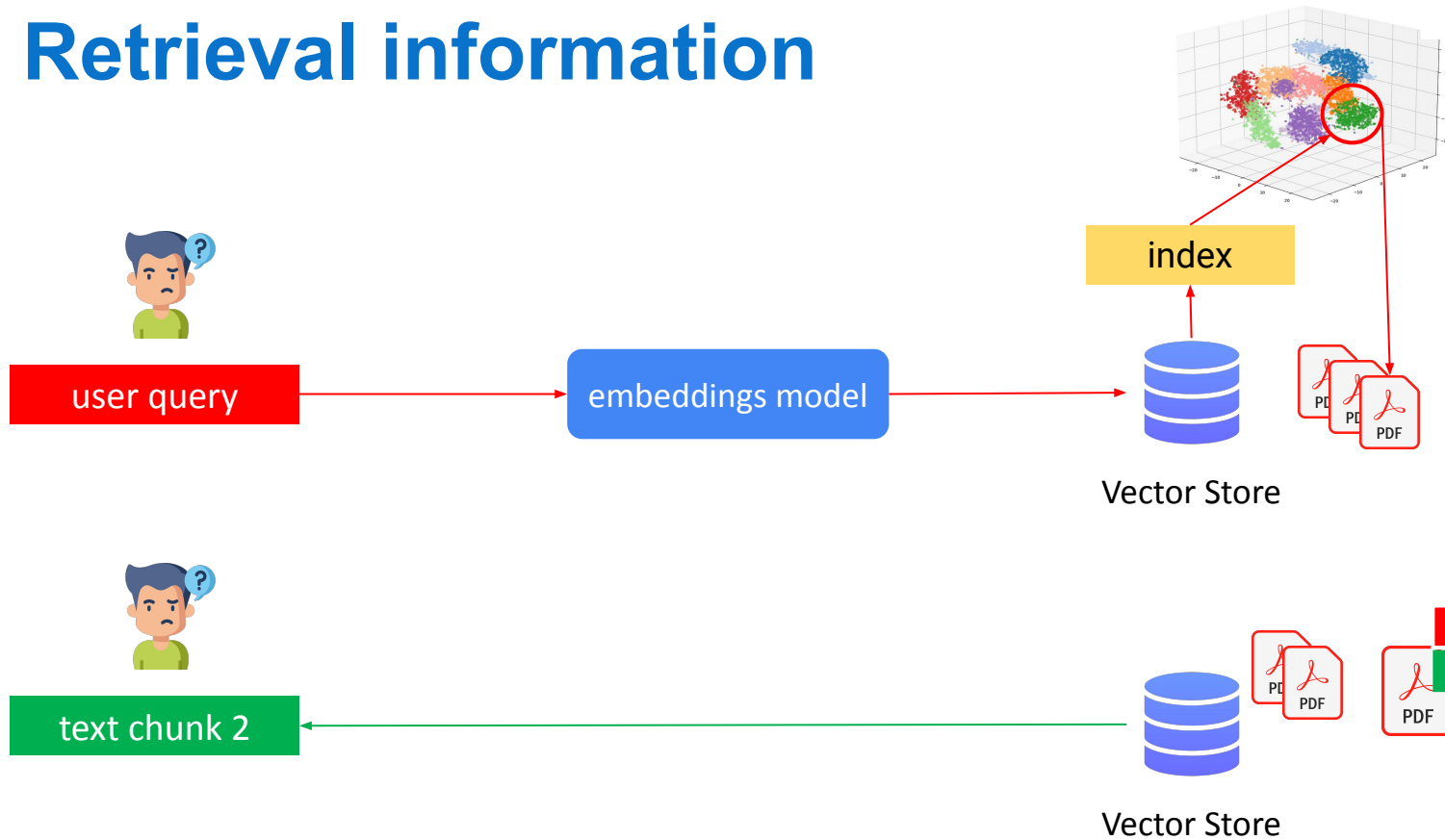
Un VS es una estructura de almacenamiento diseñada para guardar y gestionar vectores de embeddings.

Características:

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



# Retrieval information





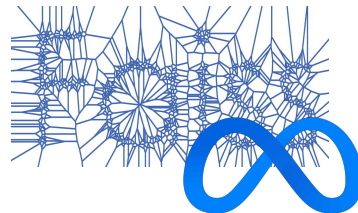
# Similarity Search

## 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:

$$j = \operatorname{argmin}_i \|x - x_i\|$$

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

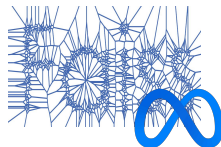


## Algoritmos de búsqueda semántica

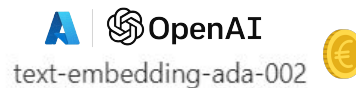
- **Índice de búsqueda exhaustiva (Flat):** Es el método más simple y directo. Utiliza la búsqueda lineal para encontrar los vecinos más cercanos y, aunque no es el más rápido para grandes conjuntos de datos, es muy utilizado por su simplicidad y precisión completa.
- **Índice IVF (Inverted File Index):** 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



## Embeddings Models



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



# LLM

## Models



### Overview

The OpenAI API is powered by a diverse set of models with different capabilities and price points. You can also make customizations to our models for your specific use case with [fine-tuning](#).

MODEL	DESCRIPTION
<a href="#">GPT-4 Turbo and GPT-4</a>	A set of models that improve on GPT-3.5 and can understand as well as generate natural language or code
<a href="#">GPT-3.5 Turbo</a>	A set of models that improve on GPT-3.5 and can understand as well as generate natural language or code
<a href="#">DALL-E</a>	A model that can generate and edit images given a natural language prompt
<a href="#">TTS</a>	A set of models that can convert text into natural sounding spoken audio
<a href="#">Whisper</a>	A model that can convert audio into text
<a href="#">Embeddings</a>	A set of models that can convert text into a numerical form
<a href="#">Moderation</a>	A fine-tuned model that can detect whether text may be sensitive or unsafe
<a href="#">GPT base</a>	A set of models without instruction following that can understand as well as

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

## Creación de cuenta

### Crédito

#### Billing

[Overview](#) [Payment methods](#) [Billing history](#) [Preferences](#)

#### Pay as you go

Credit balance ⓘ

\$6.79

- ☐ **Auto recharge is off**  
When your credit balance reaches \$0, your API requests will credit balance topped up.  
[Enable auto recharge](#)

### Límite de uso

#### Usage limits

Manage your API spend by configuring monthly spend limits. Notification emails will be sent to members of your organization with the "Owner" role. Note that there may be a delay in enforcing limits, and you are still responsible for any overage incurred.

#### Usage limit

The maximum usage OpenAI allows for your organization each month. [View current usage](#)

\$120.00

#### Set a monthly budget

If your organization exceeds this budget in a given calendar month (UTC), subsequent API requests will be rejected.

\$20.00

#### Set an email notification threshold

If your organization exceeds this threshold in a given calendar month (UTC), an email notification will be sent.

\$10.00

# OpenAI query

```
user_query = tt + '\n\n' + texts[63][:500]
print(user_query)
```

¿hay algún canon de tratamiento de aguas nocivas?

v) Ampliación y mejora de tratamiento EDAR Sant Lluís (término municipal de Sant Lluís):  
Descripción de la actuación:  
Estación depuradora de aguas residuales (EDAR).  
Tuberías generales de saneamiento de 250, 250, 250 y 350 mm de diámetro.  
Emisario terrestre de 400/355 mm de diámetro.  
Instalaciones y elementos complementarios asociados (arquetas o cámaras auxiliares, acometidas o instalaciones eléctricas consistentes en un prisma de 0,4x0,4 m con dos conductos de 160 mm de diámetro, etc.).  
Polí

```
chat_completion = client.chat.completions.create(
    model="gpt-3.5-turbo",
    messages=[{
        "role": "user",
        "content": user_query
    }]
)
```

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

## api-key (\*borrar después del curso)

nlp-course1

sk-...GiBr

Enabled

+ Create new secret key

## pricing

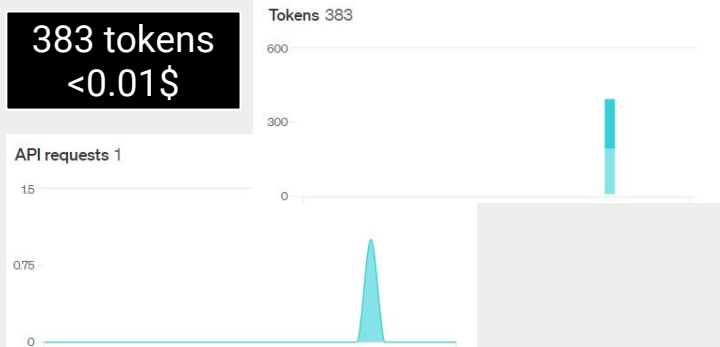
GPT-3.5 Turbo models are capable and cost-effective.

`gpt-3.5-turbo-0125` is the flagship model of this family, supports a 16K context window and is optimized for dialog.

`gpt-3.5-turbo-instruct` is an Instruct model and only supports a 4K context window.

[Learn about GPT-3.5 Turbo ↗](#)

Model	Input	Output
gpt-3.5-turbo-0125	\$0.50 / 1M tokens	\$1.50 / 1M tokens
gpt-3.5-turbo-instruct	\$1.50 / 1M tokens	\$2.00 / 1M tokens



# 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. Es útil para tareas complejas de razonamiento.

MIT  
Technology  
Review

Featured Topics Newsletters Events

ARTIFICIAL INTELLIGENCE

### Job titles of the future: AI prompt engineer

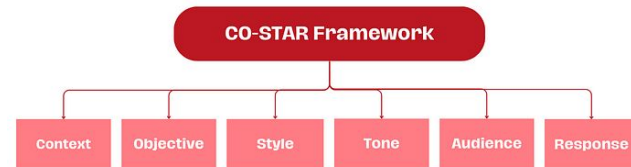
A new—and increasingly common—role helps guide generative AI.



# 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://howarddatascience.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.

delimiters (<<<, ###)

few-shot examples

Classify the sentiment of each conversation in <<<CONVERSATIONS>>> as 'Positive' or 'Negative'. Give the sentiment classifications without any other preamble text.

###

EXAMPLE CONVERSATIONS

[Agent]: Good morning, how can I assist you today? [Customer]: This product is terrible, nothing like what was advertised!

[Customer]: I'm extremely disappointed and expect a full refund.

[Agent]: Good morning, how can I help you today? [Customer]: Hi, I just wanted to say that I'm really impressed with your product. It exceeded my expectations!

###

EXAMPLE OUTPUTS

Negative

Positive

###

<<< [Agent]: Hello! Welcome to our support. How can I help you today? [Customer]: Hi there! I just wanted to let you know I received my order, and it's fantastic! [Agent]: That's great to hear! We're thrilled you're happy with your purchase. Is there anything else I can assist you with? [Customer]: No, that's it. Just wanted to give some positive feedback. Thanks for your excellent service! [Agent]: Hello, thank you for reaching out. How can I assist you today? [Customer]: I'm very disappointed with my recent purchase. It's not what I expected at all. [Agent]: I'm sorry to hear that. Could you please provide more details so I can help? [Customer]: The product is of poor quality and it arrived late. I'm really unhappy with this experience. >>>





# LLM Guardrails

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

Qué deberían incluir?

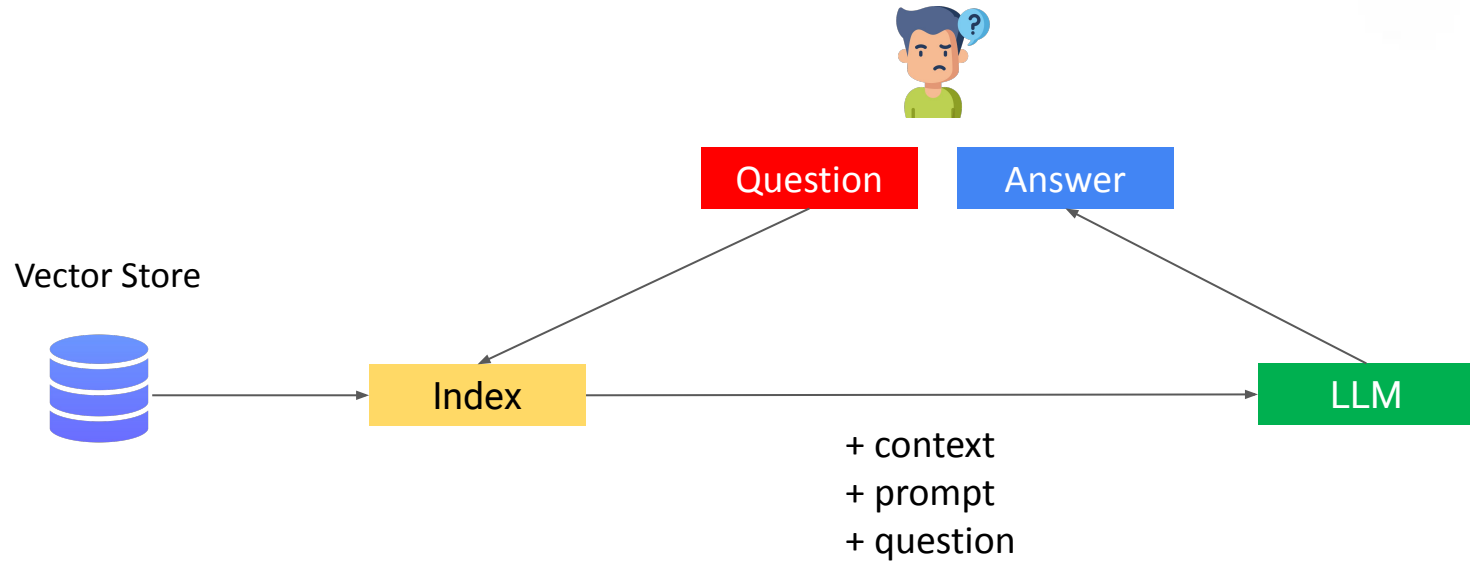
- Definición de la tarea, para que el LLM siempre recuerde lo que tiene que hacer a lo largo del chat.
- Formato de salida, para que el LLM siempre recuerde **cómo debe responder**.
- Guardrails, para que el LLM siempre recuerde **cómo no debe responder**. Este punto es especialmente importante para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

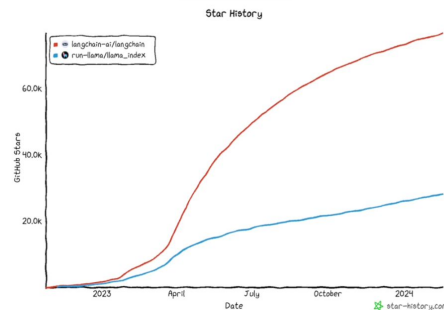
Son bibliotecas de Python diseñada para facilitar la **construcción de aplicaciones avanzadas de lenguaje natural** mediante la integración de **LLMs** con lógicas programáticas y bases de datos.

## Langchain

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

## Llama Index

Nación 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.



# 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.



# Tools



Las **Tools** son las interfaces para que un LLM pueda “interactuar con el mundo”

Ejemplo:



¿me puedes describir esta imagen?  
<https://icdn.football-espana.net/wp-content/uploads/2021/08/1002614356.jpg>

¿tiene el LLM Tools?

no

si

Lo siento como IA de lenguaje no puedo acceder a ninguna página web externa ni entender imágenes

Se revisan una a una todas las descripciones de las Tools a ver si alguna cuadra con la pregunta del usuario

Tool 1



Tool 2



Tool Image Caption

¿me puedes describir esta imagen?  
<https://icdn.football-espana.net/wp-content/uploads/2021/08/1002614356.jpg>

"use this tool when given the URL of an image that you'd like to be "described. It will return a simple caption describing the image."



Si cuadran, **el propio LLM extrae los argumentos que necesita la tool para ser ejecutada** ☐ url


```
class ImageCaptionTool(BaseTool):  
    name = "Image captioner"  
    description = "use this tool when given the URL of an image that you'd like to be "  
        "described. It will return a simple caption describing the image."  
  
    def _run(self, url: str):  
        # download the image and convert to PIL object  
        image = Image.open(requests.get(img_url, stream=True).raw).convert('RGB')  
        # preprocess the image  
        inputs = processor(image, return_tensors="pt").to(device)  
        # generate the caption  
        out = model.generate(**inputs, max_new_tokens=20)  
        # get the caption  
        caption = processor.decode(out[0], skip_special_tokens=True)  
        return caption
```

# Ejemplo



Y aquí un ejemplo de cómo ChatGPT4 acaba llamando a esas herramientas externas.


 ChatGPT 4 ▾



 **You**  
dibujame un botón rojo en fondo verde

 **ChatGPT**  
 Creating image



 ChatGPT 4 ▾

 **You**  
mira en esta web si hay una tool para discord: <https://llamahub.ai/>

 **ChatGPT**  
 Doing research with Bing



# Hub de tools



google\_docs

Verified Loader

jerryjliu

☆ 17 • ☁ 714 • 2 months ago

slack

Verified Loader

jerryjliu

☆ 13 • ☁ 152 • 2 months ago

discord

Verified Loader

jerryjliu

☆ 32 • ☁ 113 • 2 months ago

s3

Verified Loader

thejessezhang

☆ 2 • ☁ 486 • 10 days ago

mongo

Verified Loader

jerryjliu

☆ 11 • ☁ 412 • 1 month ago

file/pandas\_excel

Loader

maccarini

☆ 2 • ☁ 294 • 2 days ago

neo4j\_query\_engine

LlamaPack

wenqiglantz

☆ 1 • ☁ 24 • 7 days ago

gmail

Tool

ajhofmann

☆ 3 • ☁ 1 • 2 months ago

reddit

Loader

vanessahlyan

☆ 1 • ☁ 5 • 5 months ago

<https://llamahub.ai/>





# Agentes



Un **agente** es un motor de razonamiento y toma de decisiones automatizado basado en lenguaje natural.

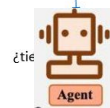
- En realidad, el LLM no es la pieza que llama a las Tools. Son los “agentes” o “routers” o “semantic functions”.

El agente utiliza un LLM como “cerebro” para exhibir capacidades más allá de la generación de texto, incluida la realización de conversaciones, la realización de tareas. Es decir, recibe una entrada o consulta del usuario y toma **decisiones internas** para ejecutar esa consulta con el fin de devolver el resultado correcto.

```
sys_msg = """Assistant is a large language model trained by OpenAI.  
  
Assistant is designed to be able to assist with a wide range of tasks, from answering simple ques  
  
Assistant is constantly learning and improving, and its capabilities are constantly evolving. It  
  
Overall, Assistant is a powerful system that can help with a wide range of tasks and provide valu  
"""  
  
new_prompt = agent.agent.create_prompt(  
    system_message=sys_msg,  
    tools=tools  
)
```



¿me puedes describir esta imagen?  
<https://icdn.football-espana.net/wp-content/uploads/2021/08/1002614356.jpg>



no

si

Lo siento como IA de lenguaje no puedo acceder a ninguna página web externa ni entender imágenes

Se revisan una a una todas las descripciones de las Tools a ver si alguna cuadra con la pregunta del usuario

Nota, la “jerga” de langchain puede variar con otros orquestadores y parece estar cambiando. Pero el potencial de lo que se puede llegar a hacer se mantiene.

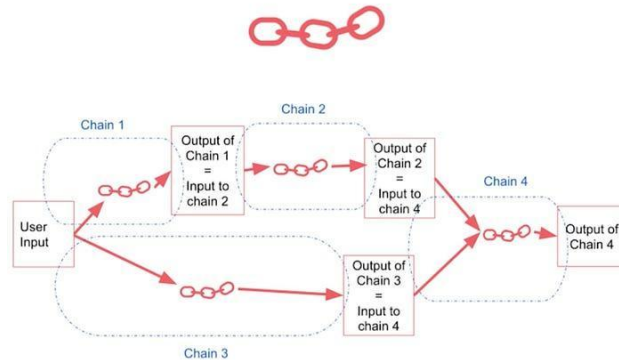


# Arquitectura de Agentes

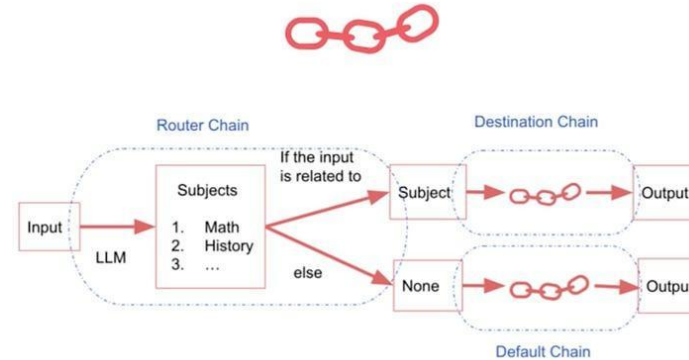


La **arquitectura de agentes** puede ser mucho más compleja. Los agentes pueden “hablar” con otros agentes de tal manera que se crean “cadenas” de razonamiento basadas en LLM.

Sequential Chain



Router Chain



Nota, la “jerga” de langchain puede variar con otros orquestadores y parece estar cambiando. Pero el potencial de lo que se puede llegar a hacer se mantiene.



