LangChain

AI & ANALYTICS

Cristina García Pérez

VIEWNEXT
AN IBM SUBSIDIARY

Índice

- Procesamiento de Lenguaje Natural
- ¿Qué es LangChain?
  - ¿Cómo funciona LangChain?
- Introducción
  - ¿Qué podemos hacer para obtener modelos de lenguaje personalizados o específicos?
  - ¿Por qué necesitamos LangChain?
- Prompts: Gestión de entradas de LLM
  - Prompt Template
- Cadenas: Combinación de LLM con otros componentes
  - LLMChain y SimpleSequentialChain

Índice

- Agentes de LangChain
  - Componentes
- Casos de Uso
  - Summary
  - CSV

VIEWNEXT
AN IBM SUBSIDIARY

Procesamiento de Lenguaje Natural

- Rama de la IA encargada de dar a los ordenadores la capacidad de comprender textos y palabras habladas.
- Motor detrás de la inteligencia de la máquina en muchas aplicaciones modernas del mundo real.

Una de las herramientas más importantes en el NLP es **LangChain**

¿Qué es LangChain?

- Framework de código abierto para el desarrollo de aplicaciones que utilizan LLMs.
- Disponible en bibliotecas basadas en Python y Javascript.
- Diseñado para que sea fácil de utilizar y muy intuitivo.
- Posibilidad de provechar el poder de los LLM y combinarlos con otras fuentes de conocimiento.

¿Cómo funciona LangChain?

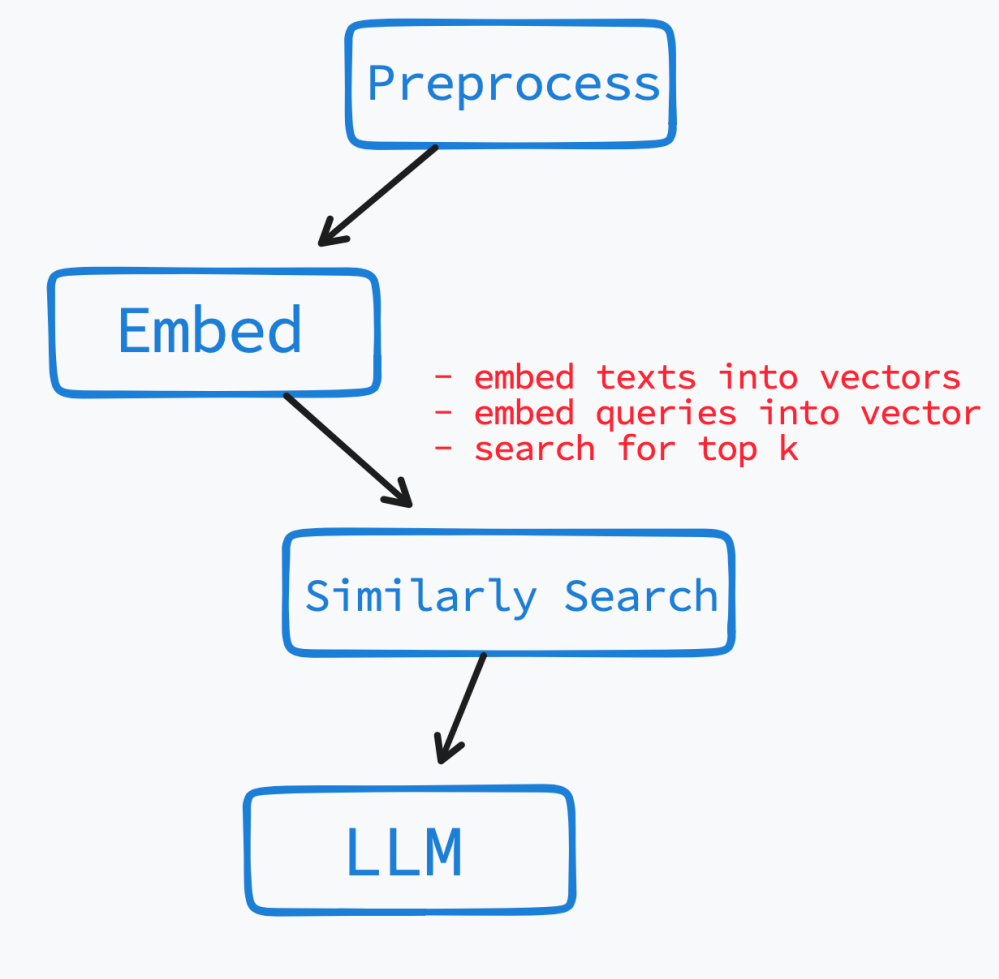
- Capacidad de enlazar diferentes componentes para crear casos de uso más avanzados utilizando LLMs.
- Componentes
  - Plantillas de Prompts: Plantillas del estilo chatbot, preguntas y respuestas…
  - Componente de LLM: Permite enlazar con diferentes modelos de lenguaje
  - Agentes: Utilizan los LLM para decidir qué acciones deben realizarse
  - Módulo de memoria: Permite dar a los LLM memoria a corto y a largo plazo

Introducción

- Los LLMs, como ChatGPT, son muy generalistas.
  - No pueden proporcionarnos respuestas específicas a determinadas preguntas.
  - No pueden realizar una serie de tareas que requieran un conocimiento profundo sobre un tema o experiencia en un campo particular.

¿Qué podemos hacer para obtener modelos de lenguaje personalizados o específicos?

¿Por qué necesitamos LangChain?



Prompts: Gestión de entradas de LLM

- Prompt Engineering: Marcar la diferencia entre obtener resultados regulares u obtener resultados buenos.
- Instrucciones aconsejables: Concisas, relevantes y que aporten contexto al modelo de lenguaje.
- Prompt Templates: Nos ayudan a construir indicaciones a partir de múltiples componentes.

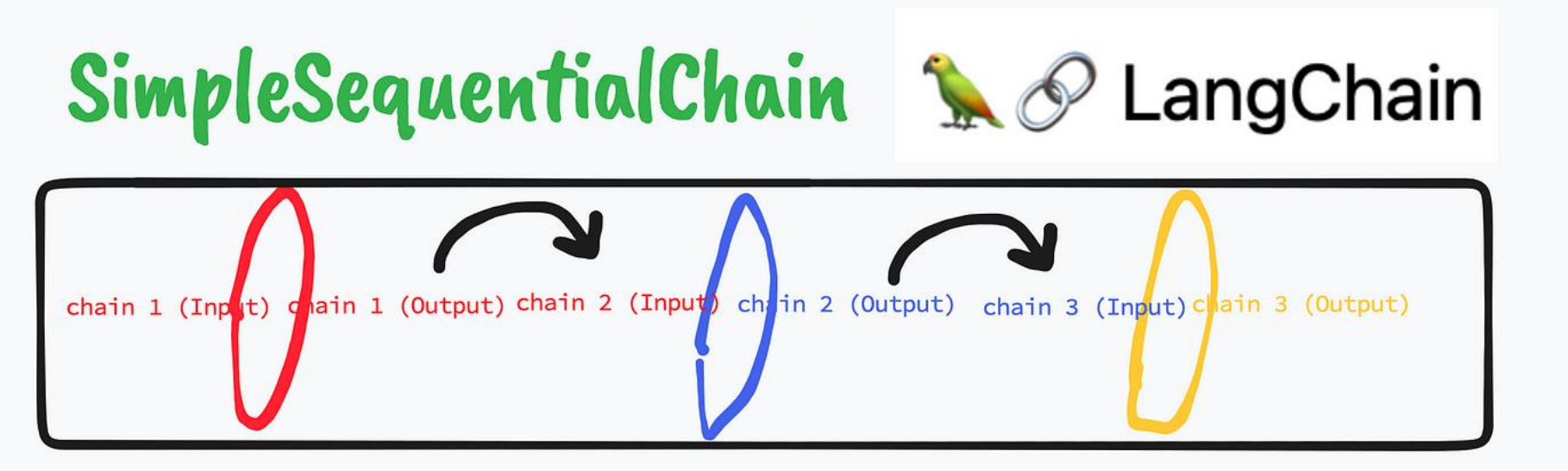
Prompts: Gestión de entradas de LLM

```
template = "¿Cuál es el mejor nombre para una empresa que fabrica {producto}?"

prompt = PromptTemplate(
    input_variables=["producto"],
    template=template
)

response = config.model.invoke(prompt.format(producto="Chocolatinas con sabor a brocoli"))
```

Cadenas: Combinación LLM con otros componentes

- Encadenamiento en LangChain: Proceso de combinar los LLM con otros componentes para crear una aplicación única.
  - Encadenamiento simple.
  - Encadenamiento secuencial: Proceso donde se encadenan más componentes.

Cadenas: Combinación LLM con otros componentes



Agentes de LangChain

- Limitación de los LLMs: Carecen de información contextual.
  - No se puedan aplicar a contextos específicos donde se requiere información actualizada o muy específica sobre determinado tema.

¿Cómo superar las limitaciones?

Dar acceso a herramientas complementarias que permitan, por ejemplo: **la búsqueda en internet**, **utilizar métodos de cálculo** o, **realizar de manera autónoma consultas en internet**.

Agentes de LangChain

- Herramienta muy útil para dotar a los LLMs de capacidades adicionales.
- Objetivo: Poder superar gran parte de sus limitaciones.

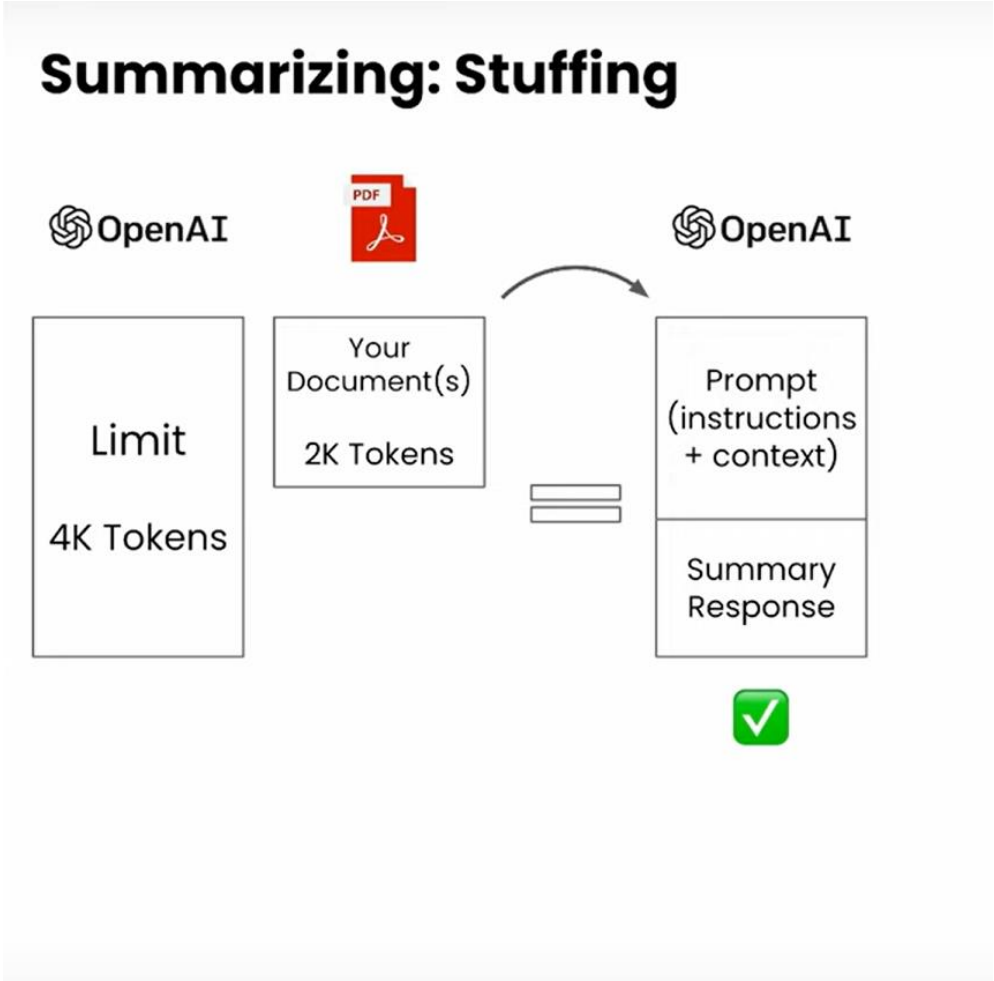
Componentes

- **LLM**: Modelo que permite el razonamiento y toma de decisiones.
- **Conjunto de herramientas** que utilizan los LLMs y, ejecutarán para realizar los pasos necesarios para lograr la tarea.

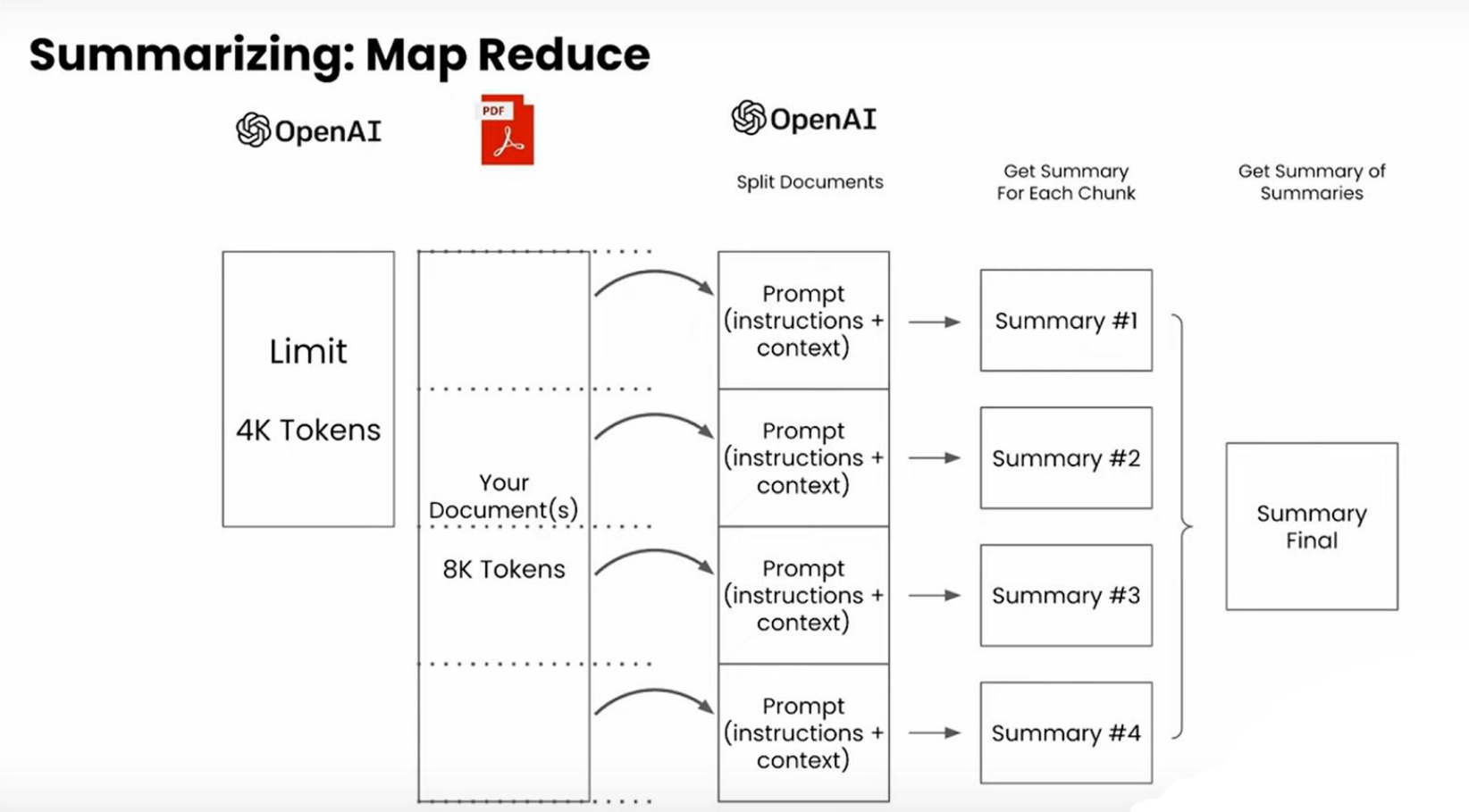
Casos de Uso: Summarization

- LLM: Gran herramienta para resumir contenido, dada su habilidad para **comprender y sintetizar texto**.
- Pero… ¿Cómo pasar los documentos a la ventana de contexto del LLM?
  - Stuff: La idea es que "empaqueta" todos los documentos en un único prompt.
  - Map-Reduce: Resume cada documento por separado en un paso de "mapeo" y luego "reduce" los resúmenes en un resumen final.
  - Refine: Construye una respuesta haciendo un bucle sobre los documentos de entrada y actualizando iterativamente su respuesta.

Casos de Uso: Summarization

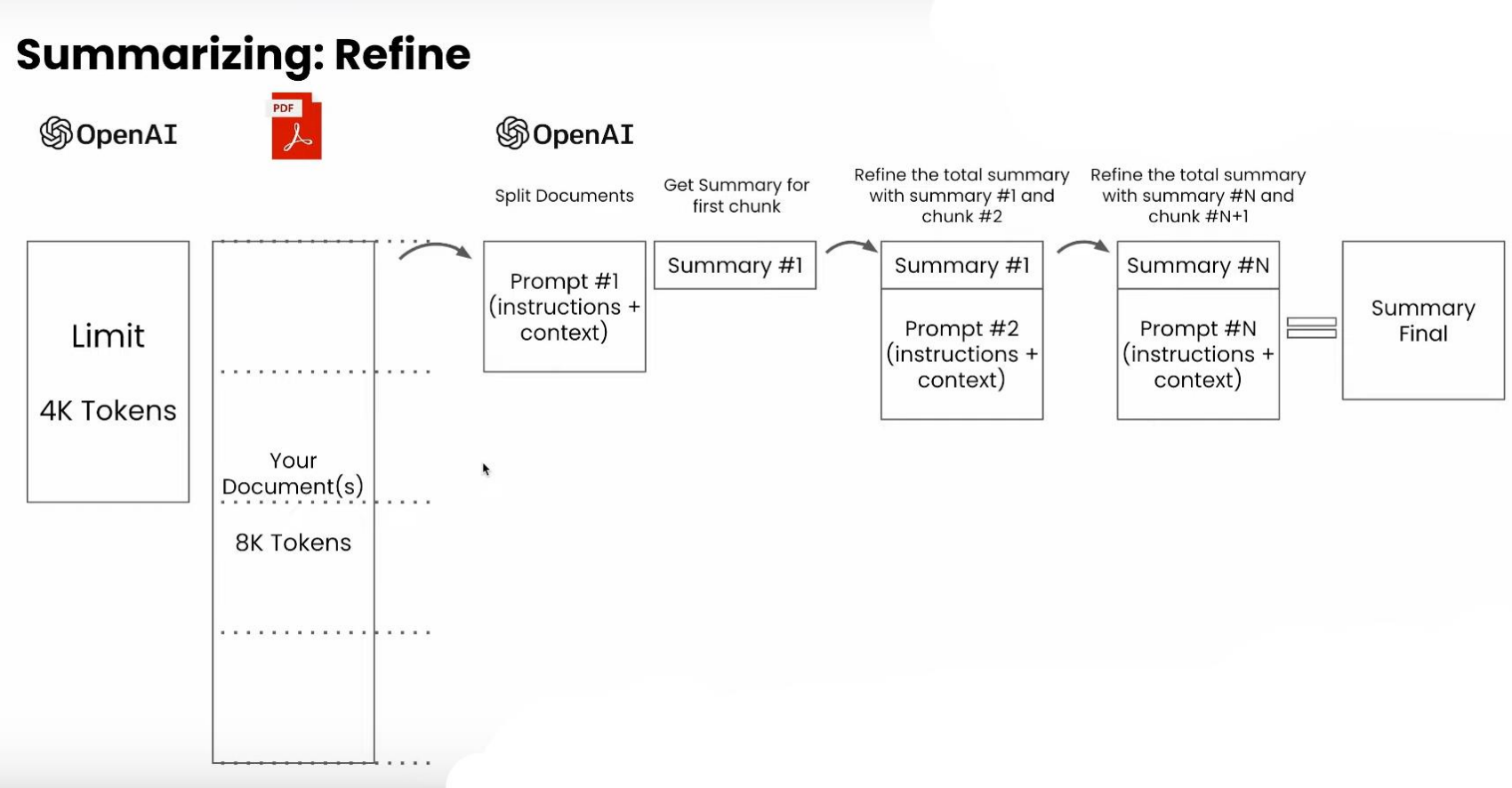


Casos de Uso: Summarization



VIEWNEXT
AN IBM SUBSIDIARY

Casos de Uso: Summarization



VIEWNEXT
AN IBM SUBSIDIARY

Casos de Uso: CSV

- Utilizar agentes para interactuar con datos en formato CSV.
- Optimizado principalmente para responder preguntas.

Crear una aplicación LLM con LangChain

Streamlit

¡MUCHAS GRACIAS!

LangChain

AI & ANALYTICS

Cristina García Pérez