marvik.

Large Language Models (LLM)
Retrieval Augmented Generation (RAG)
Langchain

UM UNIVERSIDAD DE MONTEVIDEO

Clase - 30/04

Agenda

- Introducción general al flujo de trabajo LLM + RAG
- ¿Qué es un Large Language Model (LLM)?
- Parámetros relevantes LLM
- Embeddings
- ¿Qué es una base vectorial y cómo funciona FAISS?
- RAG (Retrieval-Augmented Generation)
- LangChain
- Flujo de trabajo paso a paso
- Beneficios y aplicaciones
- Demo / Notebook

marvik.

Introducción
Arquitectura LLM + RAG

Flujo de trabajo para sistema LLM + RAG

marvik.

Este flujo representa el proceso completo que implementamos para transformar documentos PDF en respuestas contextuales utilizando modelos de lenguaje.

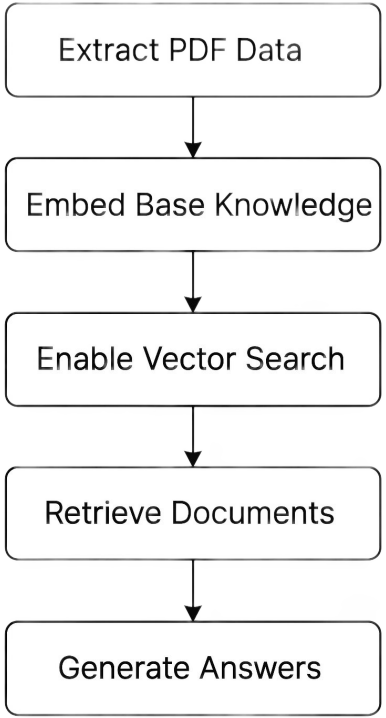
Extract PDF Data: Extraemos el texto y metadatos relevantes desde documentos.

Embed Base Knowledge: Convertimos ese texto en vectores numéricos mediante un modelo de embeddings que captura el significado semántico del contenido.

Enable Vector Search: Almacenamos los vectores para búsquedas por similitud, permitiendo recuperar información relevante

Retrieve Documents: A partir de una consulta, realizamos búsqueda vectorial para recuperar los fragmentos más relevantes.

Generate Answers: generamos respuestas contextualizadas utilizando tanto la consulta como el contenido recuperado.



marvik.

Large Language Model (LLM)

¿Qué es un Large Language Model (LLM)?

marvik.

Los Large Language Models (LLMs) son modelos de aprendizaje profundo entrenados con enormes corpus de texto. Permiten interpretar, generar y razonar sobre lenguaje humano.

Capacidades clave:

- Comprender preguntas y generar respuestas coherentes.
- Traducir, resumir o reformular texto.
- Codificar conocimiento general y de tareas específicas.

Ejemplos populares:

- GPT-4 (OpenAI)
- LLaMA 3 (Meta)
- Mistral, Claude, entre otros.

En esta clase utilizaremos meta-llama/Llama-3-8B-Instruct

Parámetros relevantes LLM

marvik.

Parámetros del modelo de generación:

- Temperatura: controla qué tan determinístico o aleatorio es el modelo.
- Muestreo: habilita la selección aleatoria de tokens, promoviendo mayor variedad en las respuestas.
- Penalización de repetición: reduce la probabilidad de repetir tokens ya utilizados.
- Límite de longitud: establece la cantidad máxima de tokens en la respuesta.
- Top-k sampling: restringe la selección al top k tokens más probables en cada paso de generación.
- Top-p sampling (nucleus sampling): permite seleccionar dinámicamente el conjunto más pequeño de tokens cuya probabilidad acumulada alcanza un umbral p.
- no_repeat_ngram_size: evita repetir secuencias de n palabras, mejorando la diversidad.
- early_stopping: permite finalizar la generación anticipadamente si se detecta un cierre natural.
- length_penalty: ajusta la preferencia por respuestas más largas o más concisas.

Estas configuraciones nos permiten balancear creatividad, coherencia y longitud.

marvik.

Embeddings y Bases de datos Vectoriales

Embeddings

marvik.

Los embeddings son representaciones vectoriales de texto que capturan el significado semántico de palabras, frases o párrafos. Son fundamentales para el funcionamiento tanto del sistema de recuperación como del modelo generativo.

En el flujo de trabajo propuesto:

- Utilizamos un modelo de embeddings como e5-base, optimizado para tareas de recuperación semántica. Este modelo convierte el texto extraído de los PDFs en vectores que se almacenan en FAISS.

- Cuando un usuario realiza una consulta, se utiliza el mismo modelo de embeddings para vectorizar la pregunta, lo cual permite hacer una comparación coherente y efectiva con los vectores previamente almacenados.

¿Qué es una base vectorial y cómo funciona FAISS?

marvik.

¿Qué es FAISS y una base vectorial?

Una base vectorial almacena información en forma de vectores numéricos que representan significado semántico.

FAISS (Facebook AI Similarity Search) es una librería optimizada para búsqueda por similitud entre vectores:

- Ideal para entornos locales o soluciones a medida.
- Permite búsquedas tipo "¿qué contenido se parece a esta pregunta?"
- Rápido y eficiente tanto en CPU como en GPU.

FAISS es útil en el flujo para encontrar los fragmentos más relevantes ante una consulta textual.

marvik.

Retrieval Augmented Generation Langchain

RAG: Retrieval Augmented Generation

marvik.

¿Qué es RAG?

RAG (Retrieval-Augmented Generation) es un paradigma híbrido que mejora los LLMs al proporcionarles información externa contextual.

¿Cómo funciona?

1) Se recupera contenido relevante desde una base vectorial (vectorDB) usando embeddings.
2) Este contenido se introduce como contexto en el LLM, que lo usa para generar respuestas más precisas y específicas.

Ventajas:

- Mejora la factualidad.
- Supera la “alucinación” de los modelos.
- Permite incorporar datos privados o actualizados sin reentrenar el modelo.

¿Qué es LangChain?

LangChain es un framework para construir aplicaciones basadas en modelos de lenguaje.
Permite componer "cadenas" de pasos, conectando:

- Modelos (LLM, chat models).
- Memoria, contexto, agentes y herramientas externas.
- Integraciones con bases vectoriales, SQL, APIs o archivos.

En esta clase veremos como LangChain nos permite ejecutar el paso de recuperación (retrieve) y generación (generate) de forma orquestada.

Sitio oficial LangChain

Beneficios del enfoque LLM + RAG

marvik.

El enfoque LLM + RAG nos brinda:

- Mayor precisión: las respuestas se basan en contenido real indexado previamente.
- Adaptabilidad: se puede aplicar a múltiples dominios (salud, legal, educación, etc.).
- Escalabilidad: el sistema puede indexar y consultar miles de documentos eficientemente.
- Flexibilidad: permite cambiar el modelo generativo o el vectorDB sin rehacer todo el flujo.

Algunas aplicaciones prácticas:

- Sistemas de soporte interno en empresas.
- Automatización de respuestas legales o regulatorias.
- Búsqueda de información técnica en manuales extensos.

¡Gracias!

¿Preguntas?

santiago@marvik.ai

marvik.