Procesamiento del Lenguaje Natural - TRABAJO PRÁCTICO FINAL

Alumna: Menescaldi, Brisa

• Ejercicio 1
Nuestro objetivo principal es crear un chatbot experto en un tema a elección, usando la técnica RAG. Siendo elegido como tema la Historia Argentina, haciendo hincapié en presidencias y los acontecimientos que han ocurrido en ellas.
Utilizando cómo fuente:
- Documentos de texto
- Datos numéricos en formato tabular (en mi caso un CSV)
- Base de datos de grafos (en mi caso, utilicé Wikidata)

Primero aparte de instalar e importar las librerías necesarias que luego serán utilizadas, descargamos las fuentes de datos que se encuentran en una carpeta de google drive.
Luego hacemos referencia a lo que son los documentos de texto, que nos encontramos con libros sobre Historia Argentina, tomando los solamente los archivos que son PDF dónde se itera por cada página extrayendo el texto; a su vez también se le hace una limpieza cómo convertir todo el texto a minúscula, eliminar caracteres no deseados, mantengo los acentos y teniendo en cuenta que después voy a hacer split de textos usando Langchain, entonces elimino los saltos de línea simples que estén seguidos por una palabra para que estos sean de referencia para dividir el texto.
Después si realizamos los split de textos, que serán convertidos en embeddings dichos fragmentos usando el modelo 'sentence-transformers/paraphrase-multilingual-mpnet-base-v2' de Hugging-Face. Dónde almaceno los embeddings de los fragmentos de texto en una base de datos vectorial ChromaDB convirtiendo a string los documentos obtenidos de los splits de textos y dándole un valor id único.

Siguiendo con los datos tabulares, que nos encontramos con un CSV que nos brinda información sobre las presidencias en Argentina. Y en referencia a la base de grafos, hacemos uso de Wikidata dónde iremos en busca de la Bibliografía de los presidentes argentinos.

Como nos pide el enunciado, debemos hacer dos clasificadores:
> basado en un modelo entrenado con ejemplos y embeddings (Unidad 3).
En el cuál, usamos el modelo 'sentence-transformers/all-mpnet-base-v2' de Hugging-Face que utiliza el clasificador con regresión logística. Y vimos que los resultados no son tan malos, pero nuestras métricas con este modelo no son las mejores.

> basado en LLM (Unidad 6).

<u>Para implementar nuestro sistema RAG:</u>
Mediante la plantilla Jinja2 le damos formato de conversación de forma estructurada que nos servirá para la generación de texto. Buscamos clasificar las preguntas en categorías usando el modelo de Hugging Face.

Siendo las categorías:
Historia Argentina: Preguntas sobre eventos históricos.
Presidencias: Preguntas sobre el período o motivos de asunción de los presidentes.
Bibliografía presidentes: Preguntas sobre detalles biográficos de los presidentes.

Antes de pasar al contexto, quiero que el contexto de los datos tabulares no sea todo el dataframe sino que solamente sobre el presidente que estoy consultando.
Después si busco el contexto en base a la pregunta ingresada y la clasificación que esta obtenga para saber a qué fuente de datos tengo que ir a buscar la información para dar la respuesta.

Para el clasificador, en el caso de las categorías Presidencias y Bibliografia presidentes hago una identificación sobre el presidente entre [corchetes], siendo que fue la manera que encontré para que lo identifique. No logré poder hacerlo por una función en particular.

Se hace la llamada al modelo para generar la respuesta. Y por último, implementamos el RAG con todas las funciones que venimos definiendo en el desarrollo del código.

• Ejercicio 2
Rerank en el contexto del Retrieval Augmented Generation (RAG) es un proceso que utiliza un modelo de aprendizaje automático avanzado donde se hace una re-evaluación de los documentos relevantes que inicialmente el sistema RAG trae con respecto a una consulta específica y así mejorar su relevancia. Luego se genera una respuesta utilizando los documentos mejor clasificados.
Al aplicar obtenemos una mejora en la precisión ya que al utilizar modelos avanzados para re-evaluar la relevancia de los documentos, se mejora la precisión de las respuestas proporcionadas; reducción de errores donde se minimiza la "alucinación" de los modelos generativos al asegurar que se utilicen los datos más relevantes y correctos; y eficiencia ya que prioriza documentos más relevantes, mejorando la eficiencia del procesamiento y reduciendo el tiempo de respuesta.

RAG con Rerank

CONSULTA → RECUPERACIÓN DE DOCUMENTOS → Rerank → GENERACIÓN DE RESPUESTAS

La información fue sacada del artículo Association of Data Scientists (Association of Data Scientists) dónde encontramos la manera de cómo implementar reranking en sistemas RAG, la importancia y los beneficios para mejorar la relevancia y precisión de las respuestas generadas por estos sistemas.