Desarrollo de un Asistente Lector Inteligente Basado en GPT-2 y RAG

Rodrigo Zeferino y Sebastian Merino

Abril 2025

Resumen

Este proyecto consiste en el desarrollo de un asistente capaz de responder preguntas técnicas basadas en un documento, combinando el poder de un modelo de lenguaje (GPT-2) con recuperación semántica mediante embeddings (RAG). Se busca que el sistema pueda adaptarse a textos privados o especializados sin necesidad de reentrenamiento completo.

1 Introducción

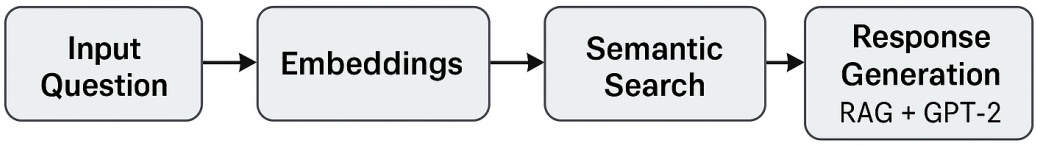
El objetivo principal del proyecto fue crear una herramienta que pudiera responder preguntas relacionadas con marketing, comportamiento del consumidor o temas similares, sin depender de conexión externa ni modelos propietarios.

Para lograrlo, se utilizó el modelo GPT-2, entrenado con una base textual generada desde Wikipedia, y complementado con una estructura de recuperación semántica (RAG) usando embeddings generados con `sentence-transformers` y búsqueda por similitud utilizando `faiss`.

2 Arquitectura del sistema

El sistema está compuesto por los siguientes módulos:

- **Generación de base lógica:** Se obtienen artículos de Wikipedia relacionados con los temas de interés y se guardan como base de conocimiento.
- **Fine-tuning de GPT-2:** Se entrena el modelo sobre la base generada para especializarlo.
- **Embeddings y recuperación:** Se generan vectores semánticos de cada fragmento y se indexan para búsqueda rápida.
- **Generación de respuestas:** Se forma un *prompt* con los fragmentos más relevantes y se genera una respuesta usando el modelo GPT-2.



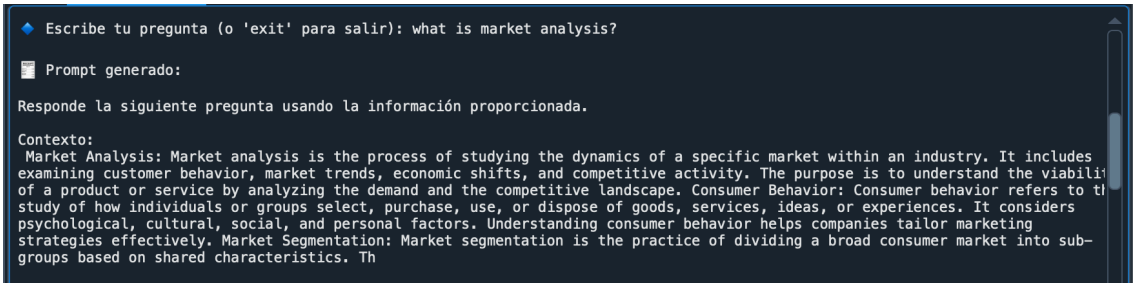
3 Ejecución y funcionamiento

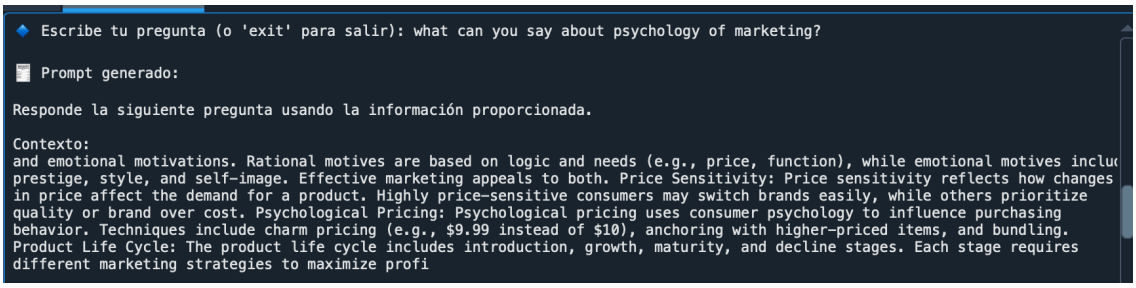
El sistema se ejecuta desde terminal, solicitando preguntas al usuario. En cada interacción:

1. Se convierte la pregunta en un vector.
2. Se recuperan fragmentos relevantes del documento cargado y/o la base general.
3. Se construye un prompt con ambos contextos.
4. GPT-2 genera una respuesta.

Evidencia de ejecución

A continuación se muestran capturas de pantalla del sistema corriendo localmente:





4 Conclusiones

El sistema fue exitoso en responder preguntas técnicas siempre y cuando la información estuviera presente en la base o el archivo cargado. Se demostró que GPT-2, combinado con recuperación semántica, puede ofrecer respuestas útiles y adaptadas a contenido propio.

Este enfoque permite flexibilidad, privacidad y escalabilidad para asistentes personalizados en distintas industrias.

Tecnologías utilizadas

- **Lenguaje:** Python 3.10
- **Modelos:** GPT-2 (via `transformers`), Sentence-Transformers, FAISS
- **Dataset:** Wikipedia (vía API) + documentos del usuario
- **Entorno:** MacBook Air M1 con entorno conda

5 Análisis Crítico Final

Limitaciones observadas

A pesar de que el sistema demuestra un desempeño sólido en la generación de respuestas basadas en contexto, se identificaron algunas limitaciones importantes:

- **Dependencia del contenido:** El modelo genera respuestas únicamente basadas en el contenido proporcionado. Si el texto cargado no contiene información relevante o explícita, la respuesta puede ser vaga o incorrecta.
- **Falta de razonamiento complejo:** GPT-2 no fue diseñado para inferir, argumentar o realizar razonamientos profundos. Opera mediante predicción de texto basada en patrones.
- **Tamaño del modelo:** El modelo seleccionado (GPT-2 pequeño) tiene limitaciones de vocabulario y profundidad comparado con versiones más recientes como GPT-3 o GPT-4.

- **Conversaciones independientes:** Cada pregunta es tratada de forma aislada. No existe memoria de conversación o seguimiento de contexto a lo largo de múltiples turnos.

Oportunidades de mejora

Basado en las observaciones, se proponen futuras mejoras:

- **Implementar modelos más robustos:** Considerar el uso de modelos de mayor tamaño o especializados en conversación, como GPT-J o LLaMA.
- **Agregar verificación de contexto:** Mejorar el sistema de recuperación semántica para validar la relevancia del fragmento recuperado antes de generar la respuesta.
- **Optimización de respuestas:** Ajustar la estructura de los prompts para guiar al modelo a producir respuestas más directas, breves o formales según necesidad.
- **Integrar memoria conversacional:** Para futuros desarrollos, considerar sistemas que mantengan el historial de conversación y adapten las respuestas dinámicamente.

Este análisis permite dimensionar el potencial del sistema actual y trazar una ruta clara hacia futuras iteraciones que permitan incrementar su desempeño y adaptabilidad.