

Trabajo Final – Modelos de Lenguajes

Juan Andres Gossweiler – Martin Levy

Introducción:

El Trabajo Final consiste en el desarrollo de una aplicación del estilo chatbot que permita realizar consultas sobre un tema específico, utilizando herramientas como Streamlit y Fast API. Las respuestas deben generarse usando un modelo de lenguaje (LLM) con una arquitectura RAG (Retrieval-Augmented Generation), sin necesidad de mantener un historial de conversación.

Tema seleccionado

El tema seleccionado es el uso del reglamento oficial del juego de cartas Magic: The Gathering, generando así una especie de juez experto en el juego, capaz de explicar el reglamento, o los conceptos más relevantes frente a una consulta.



Fuente y parseo de los datos

La fuente del reglamento se obtuvo en la página principal de Magic: The Gathering. Dentro de la sección “Reglamento”: <https://magic.wizards.com/en/rules>. Una ventaja que se presentó es que el documento se pudo descargar en formato pdf como txt.

Dado que contaba con una sola fuente de información, se optó por utilizar ChatGPT para parpear los datos y así poder alimentar el modelo. El formato pdf dio ciertos problemas, dado que el OCR contaba con numerosos errores y separaba las palabras de forma incorrecta. Finalmente logramos parpear el archivo txt,

generando así dos archivos json, uno con las reglas principales, separado por el tomo de la regla y su texto correspondiente. El segundo archivo json contaba con el glosario, separando así el término a referirse y su definición.

Armado de retriever:

Se cargó un modelo de embedding (intfloat/multilingual-e5-large) para convertir los documentos en vectores numéricos. Se construyó un índice FAISS a partir de los documentos y sus embeddings. Luego se configuró un retriever a partir del índice FAISS, que se utilizaría para encontrar documentos relevantes dada una consulta. La configuración `search_kwargs={"k": 4}` indica que se recuperarán los 4 documentos más similares.

Para seleccionar el modelo de embedding seleccionado, realizamos una comparativa entre distintos modelos existentes. En esta línea tanto teórica como prácticamente hacía sentido seleccionar este modelo.

Aspecto teórico:

- El modelo basado en e5 soporta más de 100 lenguajes, lo que es muy conveniente dado que, en nuestro caso, tenemos los documentos en inglés y las preguntas probablemente en sean todas en español.
- Consta de robustez semántica, para textos largos
- No es el más veloz de los multilingual-e5 pero esto es compensado por su alta precisión semántica.

Aspecto práctico:

- Comparado con los otros modelos (all-MiniLM-L6-v2 y all-mpnet-base-v3) más allá de la LLM seleccionada, era el modelo de embedding que brindaba resultados más acertados. Los otros modelos con los que se probó tendían a agregar información que no tenía que ver con la pregunta o generaba una respuesta que parecía que la información brindada por el usuario era otra (por ejemplo, hablar de “Emblem” como si fuera parte de la pregunta).

Sistema RAG

Se configuró la cadena RAG (rag_chain) utilizando LangChain Expression Language (LCEL). Esta cadena primero pasa la pregunta al retriever para obtener el contexto relevante, luego formatea el contexto junto con la pregunta en el prompt, y finalmente pasa el prompt al LLM para generar la respuesta.

Modelolo utilizado

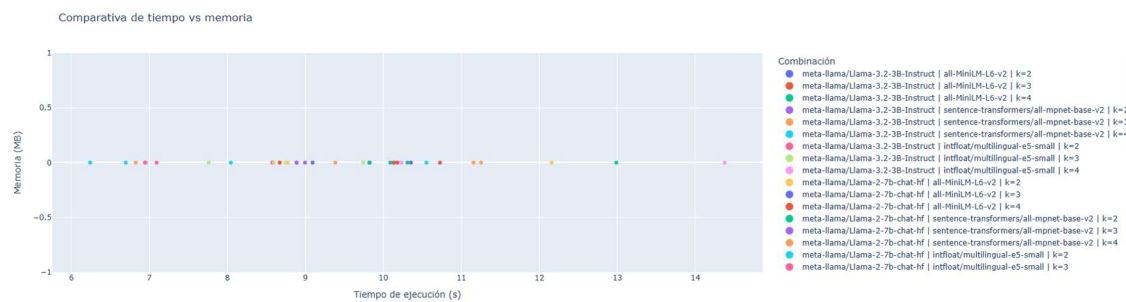
En esta sección se modificaron distintos parámetros del pipeline, la cantidad de embeddings retornados por el retriever y se seleccionaran distintos modelos de LLM para obtener la combinación óptima entre correctitud de la respuesta y tiempo de respuesta.

Se analizaron cinco modelos de lenguaje autoregresivos ampliamente utilizados en aplicaciones de generación de texto, asistencia conversacional y sistemas RAG (Retrieval-Augmented Generation). Estos modelos difieren en tamaño, arquitectura, capacidades de ajuste fino (instruct tuning), y restricciones de uso.

| Modelo | Parámetros | Instruc t-tuned | Propósito principal | Aplicaciones recomendadas | Licencia |
|--------------------------|------------|--------------------|---------------------------|------------------------------------|--------------------|
| Llama-3.2-3B-Instruct | 3.2B | Sí | Chat, instrucciones | RAG ligero, asistentes embebidos | Meta (restrictiva) |
| Llama-2-7b-hf | 7B | No | Modelo base | Fine-tuning personalizado | Meta (restrictiva) |
| Llama-2-7b-chat-hf | 7B | Sí (RLHF) | Asistencia conversacional | Chatbots, agentes conversacionales | Meta (restrictiva) |
| EleutherAI/gpt-j-6B | 6B | No | Generación general | Aplicaciones de texto genérico | Apache 2.0 |
| mosaicml/mpt-7b-instruct | 7B | Sí | Instrucciones, RAG | Producción sin restricciones | Permisiva |

Se realizó, además, una evaluación a base al tiempo de ejecución del procesamiento del prompt, para así determinar la combinación de parámetros

óptima. Hay que considerar que estas evaluaciones se hicieron en un entorno con capacidad gráfica de una tarjeta A100.



Además de esta evaluación, se realizó otra prueba, evaluando las respuestas a una misma pregunta, considerando qué tan acertadas eran las respuestas, el nivel de alucinación, o la capacidad conversacional del modelo.

Luego de ver los tiempos de respuesta de cada combinación, al igual que las respuestas de cada una, la combinación más eficiente es:

- embedding model: intfloat/multilingual-e5-large
- LLM: meta-llama/llama-3.2-3b-instruct
- k: 3

Esta combinación responde dentro del promedio de respuestas de todos los modelos evaluados. A su vez, es la combinación que responde más preciso manteniendo una respuesta corta.

Por otro lado, evaluando todas las respuestas, se ve que lo más relevante al momento de elegir es el modelo de embedding. Las respuestas entre LLM con mismo k e igual modelo de embedding tiene diferencias, pero terminan siendo muy similares. Luego, entre k=3 y k=4 no hay diferencias relevantes.

Arquitectura del sistema

A continuación, se brinda un diagrama de alto nivel con los distintos componentes del sistema.

El sistema puede separarse en 2 grandes partes: el backend y el frontend.

Frontend

Es la aplicación hecha con streamlit la cual genera y gestiona la interfaz de usuario por la cual los usuarios realizan preguntas. Está se comunica por llamada http con el backend para enviar la pregunta y esperar la respuesta de nuestro agente.

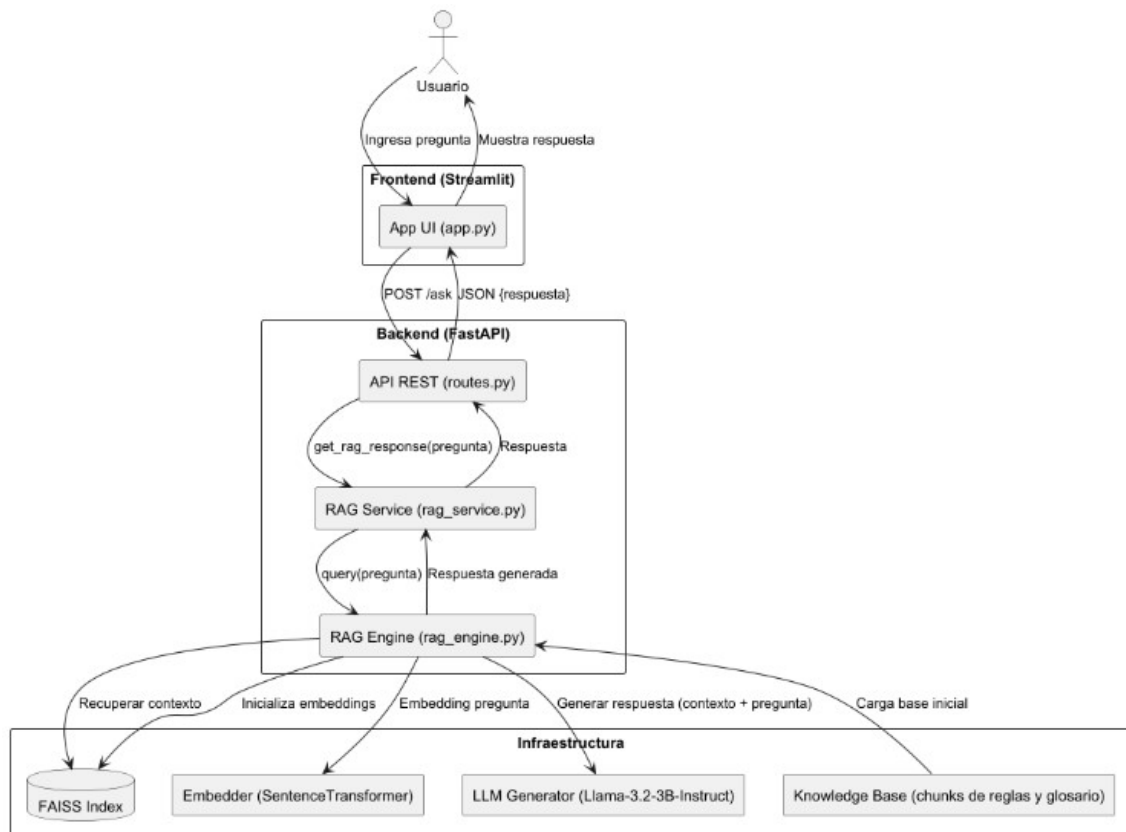
Backend

Es el encargado de obtener la LLM, generar el RAG y almacenar los chunks. A su vez, contiene la base de datos vectorial FAISS. Esta parte se crea con FastAPI.

El backend tiene 3 componentes:

1. API REST: componente que gestiona las rutas de nuestra aplicación. Gestionando como se exponen. En particular, tenemos una sola ruta que es */question* para realizar la consulta al agente.
2. RAG Service: Es el servicio encargado de generar los documentos a partir de los chunks. Define el pipeline que usara nuestra LLM y genera el indexado de FAISS. Por último, es quien crea el último componente que es el RAG Engine.
3. RAG Engine: recibe el pipeline de la LLM, el indexado y el modelo de embedding. Este componente se encarga de armar el retriever y es quien expone el método *question*. Este método se encarga de generar el rag chain con el que se crea la respuesta final al usuario. Para hacer esto también define el template del prompt a utilizar.

Por otro lado el backend tiene una carpeta “data” que almacena los chunks de la información del proyecto, en nuestro caso las reglas y el glosario de Magic the gathering.



Conclusiones

A partir de este ejercicio práctico logramos entender las complejidades e importancias de las decisiones de qué modelos utilizar. En un ambiente experimental como un notebook de colab, donde el cómputo es de parte de un servicio cloud, se pierde que tan pesados son realmente los modelos y el costo computacional asociado. En este sentido, al tener que armar una aplicación local y probarla, pudimos apreciar cómo aumentaban los tiempos de respuesta del agente, pero sobre todo los tiempos en generar el RAG y disponibilizarlo.

Por otro lado, más allá de que las respuestas no fueran del todo exactas, aprendimos cómo rápidamente se puede generar un agente y validar conceptos de qué tan sencillo puede ser generar un chatbot especializado en cierta temática.

Para futuros pasos, se podría hostear la aplicación en AWS o GCP para acceder a procesadores y GPUs más potentes y así poder realizar (y utilizar) modelos más complejos. Asimismo, para el caso de nuestro agente, se podrían mejorar las respuestas si se agregaran chunks con información de todas las cartas disponibles.

del juego, al igual que nomenclaturas de estructuras de decks utilizados por jugadores.


De la mano de lo anterior, se podría mejorar el template para sumar alguna estrategia de prompt engineering y así obtener respuestas más precisas y mejorar la resolución a preguntas complejas.

Capturas Interfaz

Pregunta 1: Describe la habilidad de flying en una criatura




Welcome! Ask any question related to our knowledge base and get an AI-generated answer using Retrieval-Augmented Generation.

 Enter your question:

Describe la habilidad de flying en una criatura

Ask

 Here's what I found:

La habilidad de flying en una criatura significa que esa criatura puede volar sobre cualquier bloqueo, es decir, puede atacar a cualquier criatura o jugador sin tener que superar un bloqueo. Esto se indica en el reglamento como una habilidad de evasión (evasion ability) y se describe en la sección 702.9, "Flying".

En otras palabras, cuando una criatura tiene flying, no necesita superar el bl

Es importante destacar que esta habilidad solo afecta cómo se bloquean la

Si tienes alguna duda sobre cómo funciona flying en un determinado escena

Directa apelación correcta al Glosario en la pregunta sobre una cualidad de una criatura.

Pregunta 2: ¿Cómo funciona el stack y cómo se resuelve?




Welcome! Ask any question related to our knowledge base and get an AI-generated answer using Retrieval-Augmented Generation.

 Enter your question:

¿Cómo funciona el stack y cómo se resuelve?

Ask

 Here's what I found:

El stack es un concepto fundamental en Magic: The Gathering que se refiere a la secuencia en la que se resuelven las habilidades y hechizos en un turno de juego. Cuando un hechizo o habilidad se coloca en el stack, significa que su efecto se va a resolver en un orden específico.

Según la regla 608, "Resolving Spells and Abilities", cuando un hechizo o habi

Ahora bien, hay un aspecto importante a considerar: algunas habilidades

Por ejemplo, si un hechizo dice "Puedo ser lanzado desde mi mano o de mi

Es importante tener en cuenta que el stack se vacía de abajo hacia arrib

Si tienes alguna duda sobre cómo se está resolviendo un hechizo o habili

Recuerda que la regla 609, "Stacking", también es relevante en este cont

Pregunta difícil para el propio jugador de Magic. Concepto se explica de forma correcta, haciendo referencia a las secciones correctas del reglamento.

Pregunta 3: ¿Cuáles son las distintas formas de perder la partida?



RAG Chatbot

Welcome! Ask any question related to our knowledge base and get an AI-generated answer using Retrieval-Augmented Generation.

Enter your question:

¿Cuáles son las distintas formas de perder la partida?

Ask

Here's what I found:

En Magic: The Gathering, hay varias formas de perder una partida. Según el reglamento, una de ellas es cuando un jugador concede la partida en cualquier momento (rule 104, "Ending the Game"). Al hacerlo, ese jugador deja la partida inmediatamente y pierde la partida.

Otra forma de perder la partida es cuando un efecto establece explícitamente que un jugador pierde la partida (rule 104, "Ending the Game").

Además, según el reglamento, existen reglas adicionales específicas para ciertos tipos de juegos, como dos cabezas gigantes (rule 810.8) y emperadores (rule 809.5), así como una regla adicional para partidas de comandante (rule 903.10). Sin embargo, estas reglas no cambian la forma general de perder la partida descrita anteriormente.

Siempre es recomendable consultar con un juez en caso de duda o controversia durante un torneo, ya que pueden proporcionar orientación y asegurarse de que se apliquen las reglas correctas.

Respuesta correcta, pero incompleta, faltan otras formas en la que un jugador puede perder la partida, como por ejemplo cuando no hay cartas en el mazo, o cartas que lo indican de forma directa.

Pregunta 4: ¿Cuál es la distancia del sol a la luna?



Welcome! Ask any question related to our knowledge base and get an AI-generated answer using Retrieval-Augmented Generation.

 Enter your question:

¿Cuál es la distancia del sol a la luna?

Ask

 Here's what I found:

Lo siento, pero parece que hay un error en tu pregunta. El texto proporcionado no tiene relación con el reglamento del juego de cartas Magic: The Gathering.

Si deseas saber algo sobre el juego, estaré encantado de ayudarte. Por favor, proporciona una pregunta relacionada con el juego y estaré encantado de responder.

Si necesitas recordatorios o ayuda para encontrar una sección específica del reglamento, puedo ofrecerte esa información.

¿En qué puedo ayudarte?

Respuesta coherente ante pregunta que no tienen nada que ver con el juego.