Informe TP2 NLP

Recolección de información:

Para empezar, descargue archivos PDF con información sobre el juego Viticulture, y los almacene en una carpeta de Google Drive. Luego, utilizando la ayuda de LLMs fabriqué 2 archivos CSV con información numérica y tabular del juego, y añadí esos dos archivos a la carpeta. Desde mi Colab extraje estos archivos para luego utilizarlos como fuente del RAG. Por otro lado, la BD de grafos la cree a mano utilizando las reglas y también la ayuda de distintos LLMs, y luego revisa cuidadosamente que las relaciones de la BD sean ciertas.

Base de datos vectorial:

Cree una BD vectorial con ChromaDB en la que guarde los embeddings de los textos PDF para luego acceder a esa información de forma conveniente utilizando una búsqueda vectorial.

LLM utilizado:

Para no gastar mucho tiempo ni recursos decidí usar un LLM gratuito que ofrece Hugging Face con solo mil millones de parámetros, el modelo se llama “bigscience/bloom-1b1”. Luego pude comprobar que este modelo tuvo un muy mal rendimiento en general lo que terminó afectando al resultado final del chatbot negativamente. Probablemente si se hubiera usado un LLM potente y de pago como los que ofrece OpenAI los resultados habrían sido muy superiores.

Chatbot:

Para seleccionar qué fuente de datos usar, elegí utilizar prompt engineering, de esta forma primero le paso al modelo un prompt ampliado con la técnica few-shot para que elija entre PDFs, CSVs y Grafos. Por lo que pude experimentar nunca elige grafos, lo cual es un problema que probablemente se podría solucionar utilizando un mejor modelo de lenguaje como backbone.
Luego de seleccionar la fuente de datos, dependiendo de la respuesta se toma el contexto desde la BD vectorial (en caso de PDF), se realiza una búsqueda sobre los dataframes que luego se pasa como contexto al LLM (en el caso de CSV) o se busca alguna relación en el grafo si esa es la respuesta del selector.

Pude observar que en el caso de CSVs el modelo funciona bastante bien y agarra la información correctamente, aunque a veces responde mal, probablemente porque el modelo es muy pequeño. Por otro lado, en el caso de seleccionar PDFs, primero ocupaba toda la ventana de contexto y tuve que achicar la respuesta, y luego pude comprobar que el LLM alucinaba respuestas absurdas luego de recibir el contexto.

Conclusión:

Este método para darle contexto necesario a un LLM parece ser muy útil en el caso de requerir un sistema experto en un tema específico del que tenes informacion pero el modelo no, ya sea porque la información es privada (un chatbot para una empresa), o porque el modelo simplemente no fue entrenado con esos datos. En este caso, no pude terminar de observar todo el valor que puede brindar esta técnica debido a las limitaciones técnicas del LLM utilizado.

Valentín Dalmau