Asignatura	Datos del alumno	Fecha
Herramientas para la Computación en la Nube Dirigidas a Inteligencia Artificial	Apellidos: Ortega de Mues	25/09/2024
	Nombre: Mariano	

Actividad 2. Ingeniería de *prompts* con modelos en la nube a través de Mistral AI

1. El modulo ctransformers esta optimizado para trabajar sobre CPU. Desarrollado en C/C++ sobre GGML, una libreria de tensores acelerada. Es especialmente adecuado para entornos de LLM donde no disponemos de GPU y nuestra intención es la inferencia, no la modificación o el entrenamiento del modelo LLM, como es nuestro caso explotando Mistral.

2. AutoModelForCausalLM puede ser iniciado partiendo de un modelo ya existente y entrenado (como en nuestro caso) mediante from_pretained(), o de una configuración que describe la arquitectura del modelo, pero sin los pesos del entrenamiento, mediante from_config().

3. Los LLM generan la respuesta de forma secuencial token por token. El método llm() genera esta respuesta hasta alcanzar las condiciones de parada indicadas, controlando parámetros en el proceso de generación. Cada vez que un token es generado, es agregado a la respuesta y la secuencia total incluida dentro del contexto para la generación de la siguiente. Se emplea un for para consumir este proceso. El parámetro max_new_tokens se emplea para controlar el número total de tokens generados. Si se llega al número indicado si alcanzar un fin de respuesta el modelo deja de generar. Los tokens son cada una de las palabras generadas por el LLM ante una consulta.

4. Empleamos la función para poder trabajar con plantillas de promts. El uso de plantillas es una práctica muy recomendable ya que permite estructurar modelos

de prompt para consultas recurrentes y organizarlas. Son muy flexibles y reutilizables. Gracias a los prompt la integración de LLM en nuestras aplicaciones es más sencilla que si es necesario modificar la consulta, modificamos generalmente la plantilla.

En algunos modelos LLM ,es posible utilizar los tokens [INST] e [INST] para realizar fine tunnig con instrucciones. Estas instrucciones permiten estructurar el contexto o establecer condiones a la respuesta esperada por parte del LLM.

5. Gracias a la técnica de fer-shot-prompts, es posible influenciar al LLM con ejemplos de lo consideramos como formato correcto para el resultado de una la consulta. Podemos implementarlo usando FewShotPromptTemplate().

6. El LLM indica que no lo sabe y que no se deriva del contexto proporcionado. Celda 15.

7. Las alucinaciones son respuestas inventadas o afirmaciones falsas generadas por los modelos LLM. Las alucinaciones no se basan en la realidad o el contexto proporcionado al LLM. En el notebook se presentan dos posibles soluciones, emplear [INST] directamente en el prompt o mediante templates para indicarle que si la confianza en la respuesta es baja simplemente indique que no lo sabe.

8. La variable ‘resp’ contiene la última respuesta generada por el LLM. El uso de esta variable para ir manteniendo el contexto en el dialogo del chat bot es muy útil ya que habilita una conversación fluida.

9. Para evitar que Mistral incluya en sus respuestas información sensible podemos emplear el string PII (Personal Identifiable Information) y explícitamente indicarle

que sustituya la información sensible en su respuesta por la cadena de '###'. Pese al comentario del

10. De forma general, esta cadena se emplea para el reconocimiento de fin de sentencia. Mistral no emplea directamente esta cadena, otros LLM si la usando como GPT-3 y GPT-4. La cadena equivalente en Mistral es </s>. Esta cadena se puede incluir por compatibilidad en el ejemplo y los transformes usados.