TUIA NLP 2025
Práctica Unidad 6 Chatbots y sistemas de diálogo

1. Levantar una instancia de LLM en el código mediante API, puede ser HuggingFace u otro proveedor.

   En el primer caso (usando HuggingFace), para poder acceder a estos modelos vamos a necesitar apoyarnos en la siguiente librería huggingface_hub, luego deberemos obtener un token de acceso para poder acceder mediante API a los modelos que nos provee HuggingFace, los cuales están en la siguiente lista.

   El siguiente código nos será de ayuda para poder contactarnos mediante API con algunos de los modelos disponibles:

```
import os
from huggingface_hub import InferenceClient

client = InferenceClient(
    provider="hf-inference",
    api_key=<HF_TOKEN>,
)

completion = client.chat.completions.create(
    model=<MODEL_ID>,
    messages=[
        {
            "role": "user",
            "content": "What is the capital of France?"
        }
    ],
)

print(completion.choices[0].message)
```

   En donde reemplazamos `<HF_TOKEN>` por el token de acceso y `<MODEL_ID>` por el modelo que vamos a utilizar.

   a. A partir de esto, le enviaremos un prompt al modelo para luego obtener esa respuesta e imprimirla. ¿Qué observamos en los datos devueltos?

2. Apoyándonos en el mismo prompt enviado en el ejercicio anterior:
   a. Realizar múltiples llamadas al LLM variando los siguientes parámetros:
      - temperature: Probar con un valor bajo (0.2) y uno alto (0.9)
      - max_new_tokens / max_length: Probar que genere respuestas largas.
      - top_k y top_p

   b. Tomando en cuenta las llamadas anteriores, ¿Como afecta la modificación de cada parámetro?

3. Utilizando la estructura de mensajes con roles (system, user, assistant), enviar un prompt donde el mensaje de system instruya al LLM a adoptar una personalidad o rol específico. Luego, enviar un mensaje de user preguntando algo relacionado con ese rol.

   Por último, evaluar si el LLM adoptó el rol indicado y cómo esto influyó en la respuesta.

4. Trabajar la gestión de historial de conversaciones:
   a. Iniciar una conversación con el LLM. El primer turno puede ser una pregunta simple del usuario.
   b. Almacenar el mensaje del usuario y la respuesta del asistente en una lista de diccionarios, donde cada diccionario representa un mensaje con su role y content.
   c. Para el segundo turno del usuario, formular una pregunta que dependa del contexto del primer turno.

   Evaluar:
   - ¿El LLM es capaz de "recordar" y utilizar la información de los turnos anteriores gracias al historial enviado?
   - ¿Qué ocurriría si no se enviara el historial completo? Justificar.

5. Diseñar el prompt para cumplir tareas específicas:
   a. Elegir una tarea simple como una clasificación, luego diseñar un prompt *few-shot*. Esto implica proporcionar al LLM un par de ejemplos (pregunta/respuesta o instrucción/ejemplo de salida) dentro del mismo prompt antes de la instrucción final.
   b. Enviar este prompt al LLM y analizar la calidad del eslogan generado.

   Evaluar:
   - Probar con diferentes números de ejemplos (uno, dos, tres). ¿Cómo afecta la cantidad de ejemplos a la calidad de la salida?
   - ¿Qué tan importante es la calidad y relevancia de los ejemplos proporcionados en el prompt few-shot?

6. A partir del ejercicio anterior, desarrollar un clasificador siguiendo las siguientes pautas:
   a. Definir un conjunto pequeño de intenciones que un usuario podría tener al interactuar con un hipotético chatbot experto en juegos de mesa.
   b. Para cada intención, crear 2-3 ejemplos de frases que un usuario podría escribir.
   c. Luego construir un prompt que instruya al LLM a clasificar una nueva consulta del usuario en una de las intenciones definidas. El prompt debe incluir ejemplos de cada intención.

   Explicar por qué la técnica de few-shot prompting es adecuada para esta tarea y cómo la estructura del prompt ayuda al LLM a entender la tarea.

7. Siguiendo el punto anterior, se tendrá que crear una función en Python que reciba una consulta de usuario como entrada y haga lo siguiente:
    a. Construir el prompt de clasificación (como se diseñó en el paso anterior), insertando la consulta del usuario en el lugar correspondiente.
    b. Enviar el prompt completo al LLM.
    c. Procesar la respuesta del LLM para extraer únicamente la etiqueta de la categoría predicha.
    d. Probar la función con al menos 10 consultas de usuario nuevas (diferentes a las usadas en los ejemplos del prompt), cubriendo todas las intenciones definidas.

    Evaluar lo siguiente:
    - ¿Qué tan preciso es el clasificador?
    - ¿Existen casos donde el LLM clasifica incorrectamente? Analizar posibles causas.
    - ¿Cómo se podría mejorar la precisión del clasificador?