Proyecto: Construcción de una Aplicación Multimodal con OCR y LLMs

Curso: Inteligencia Artificial

Universidad: EAFIT

Profesor: Jorge Padilla

Modalidad: Taller práctico de desarrollo.

---

1. Introducción y Objetivos

En este taller, construiremos una aplicación web interactiva de principio a fin. El objetivo es integrar dos dominios de la IA: la **Visión Artificial** para "leer" texto de imágenes y el **Procesamiento de Lenguaje Natural (NLP)** para "entender" y "transformar" ese texto. Usaremos Python, la librería Streamlit para la interfaz, y nos conectaremos a APIs externas de vanguardia como GROQ y Hugging Face para acceder a modelos de lenguaje potentes.

Al finalizar, cada estudiante deberá tener una aplicación funcional que:

- Permita al usuario subir una imagen.
- Extraiga automáticamente el texto de esa imagen usando un modelo OCR.
- Envíe el texto a un Modelo de Lenguaje Grande (LLM) para realizar tareas como resumir, traducir o analizar.
- Permita cambiar de proveedor de LLM (GROQ vs. Hugging Face) y ajustar parámetros clave para ver cómo cambia el resultado.

---

2. Configuración del Entorno (Actividad Previa)

Antes de empezar a programar, cada estudiante debe preparar su entorno de desarrollo.

Tareas:

1. **Obtener Claves de API:**
   - Crea una cuenta en Groq.com y genera una clave de API.

    - Crea una cuenta en HuggingFace.co y genera un *Access Token*.
2. **Estructura del Proyecto:**
    - Crea una carpeta para el proyecto.
    - Dentro, crea un archivo principal llamado app.py.
    - Crea un archivo llamado .env para almacenar tus claves de API de forma segura. El formato debe ser:
      `GROQ_API_KEY="tu_clave_de_groq"`
      `HUGGINGFACE_API_KEY="tu_clave_de_huggingface"`

3. **Instalación de Librerías:**
    - Identifica e instala todas las librerías de Python necesarias para el proyecto. Deberás investigar cuáles son requeridas para:
        - Crear la aplicación web.
        - Realizar el Reconocimiento Óptico de Caracteres (OCR).
        - Conectar con la API de GROQ.
        - Conectar con la API de Hugging Face.
        - Cargar las variables de entorno desde el archivo .env.

Módulo 1: El Lector de Imágenes (OCR)

Objetivo: Construir la primera parte de la aplicación: una interfaz que reciba una imagen y muestre el texto que contiene.

Actividades a desarrollar en app.py:

1. **Crear la Interfaz Básica:**
    - Importa Streamlit.
    - Configura un título para la página (ej: "Taller IA: OCR + LLM").
    - Añade un encabezado para esta sección.
2. **Implementar la Carga de Archivos:**
    - Utiliza el widget de Streamlit apropiado para permitir que el usuario suba un archivo de imagen (restringe los tipos a .png, .jpg, .jpeg).
3. **Cargar y Ejecutar el Modelo OCR:**
    - Instancia el lector de la librería easyocr.
    - **Desafío:** La carga del modelo puede ser lenta. Investiga cómo usar los decoradores de caché de Streamlit (ej. @st.cache_resource) para asegurar que el modelo se cargue en memoria solo una vez y no cada vez que el usuario interactúa con la app.
4. **Procesar y Mostrar Resultados:**
    - Si el usuario ha subido una imagen:
        - Muéstrala en la interfaz.
        - Convierte la imagen a un formato que el modelo OCR pueda procesar.

- Ejecuta el modelo sobre la imagen para extraer el texto.
- Muestra el texto extraído en un área de texto (st.text_area) para que el usuario pueda verlo y copiarlo si lo desea.

Módulo 2: Conexión con el Cerebro Lingüístico (GROQ API)

Objetivo: Tomar el texto extraído y enviarlo a la API ultrarrápida de GROQ para que un LLM realice una tarea específica.

Actividades a desarrollar:

1. **Cargar las Credenciales:**
   - Importa las librerías necesarias para cargar las variables del archivo .env.
   - Asegúrate de que tu clave de API de GROQ esté disponible en tu código.
2. **Expandir la Interfaz de Usuario:**
   - Añade un st.selectbox para que el usuario elija el modelo de GROQ que desea usar (ej: llama3-8b-8192, mixtral-8x7b-32768).
   - Añade otro st.selectbox que le permita al usuario elegir una tarea a realizar sobre el texto (ej: "Resumir en 3 puntos clave", "Identificar las entidades principales", "Traducir al inglés").
   - Añade un st.button con el texto "Analizar Texto".
3. **Implementar la Lógica de la API:**
   - Cuando el usuario presione el botón, y solo si hay texto extraído del paso anterior:
     - Instancia el cliente de la API de GROQ.
     - Construye la llamada a la API (chat.completions.create).
     - **Punto Clave:** Deberás estructurar correctamente el prompt dentro del parámetro messages, utilizando los roles system (para darle instrucciones al modelo) y user (para pasarle el texto a analizar).
     - Muestra la respuesta del modelo en la pantalla. Utiliza st.markdown para un formato de texto enriquecido.
4. **Desafío de Persistencia:**
   - ¿Qué sucede si interactúas con un widget y Streamlit recarga la página? El texto extraído podría perderse. Investiga cómo usar st.session_state para guardar el texto extraído y que persista entre interacciones.

Módulo 3: Flexibilidad y Experimentación

Objetivo: Hacer la aplicación más robusta y versátil, permitiendo cambiar de proveedor de

API y ajustar los parámetros del modelo.

Actividades a desarrollar:

1. **Control de Parámetros:**
    - Añade dos widgets st.slider a la interfaz para que el usuario pueda controlar de forma interactiva los parámetros temperature (creatividad) y max_tokens (longitud de la respuesta).
    - Asegúrate de que los valores de estos sliders se pasen correctamente en la llamada a la API de GROQ.
2. **Integración de un Segundo Proveedor (Hugging Face):**
    - Añade un widget st.radio al principio de la sección de NLP para que el usuario elija entre "GROQ" y "Hugging Face".
    - Implementa la lógica condicional (if/elif) para ejecutar el código correspondiente al proveedor seleccionado.
    - Para la opción de Hugging Face:
        - Usa la librería huggingface_hub para instanciar un InferenceClient con tu clave de API.
        - Investiga cómo llamar a una tarea de la API de Inferencia (ej. summarization).
        - **Nota:** La estructura de la llamada y los parámetros pueden ser diferentes a los de GROQ. ¡La lectura de documentación es clave aquí!
        - Muestra el resultado en la interfaz.

Puntos de Discusión y Reflexión Final

Una vez que la aplicación esté funcionando, el grupo deberá debatir:

- ¿Qué diferencias de velocidad notaron entre GROQ y Hugging Face? ¿A qué creen que se debe?
- ¿Cómo afecta el cambio de temperature a las respuestas del LLM? ¿Cuándo usarían un valor bajo vs. uno alto?
- ¿Qué tan importante fue la calidad del texto extraído por el OCR para la calidad del análisis del LLM?
- ¿Qué otros modelos o tareas se podrían integrar en esta aplicación? (Ej: Análisis de sentimientos, clasificación de texto, generación de preguntas y respuestas).