Aplicación de modelos extensos de lenguaje en la ambientación narrativa

Autor

Mario Gómez Alonso

Director del trabajo

Bach. Josselyn Sofía Ordóñez Olazábal

Este plan de trabajo ha sido realizado en el marco de la asignatura Gestión de Proyectos entre octubre y diciembre de 2023.

Tabla de contenido

IMPORTANTE: No borrar las consignas en cada una de las cuatro secciones de este documento, de forma tal que el jurado tenga claro qué es lo solicitado en cada caso, así como el significado de los símbolos y colores utilizados.

Revisión	Cambios realizados	Fecha
1.0	Creación del documento	23/05/2025

1. Breve resumen del trabajo realizado hasta la fecha

Elabore un detalle del estado del proyecto a la fecha. **Utilicé como mínimo dos páginas completas y como máximo tres páginas**. Explique muy brevemente en qué consiste su Trabajo Final, **aunque esa información esté más detallada en el Plan de Trabajo al cual su Jurado también tiene acceso**. Incluya imágenes y tablas según considere apropiado. **Indique con claridad por qué estima que podrá completar todos los faltantes** (o al menos la gran mayoría) **antes del inicio del Taller de Trabajo Final.**

El trabajo final está enfocado en ampliar los servicios que ofrece la aplicación de la empresa Critical Match con el uso de modelos extensos de lenguaje. Esta app de móvil permite a sus usuarios la creación y búsqueda de grupos para participar en partidas de rol y el trabajo se enfoca en ofrecer a los usuarios la capacidad para enriquecer su ambientación con más contenido narrativo. Más concretamente, se trata de la implementación de un prototipo que sirva diferentes servicios web para que el usuario, a partir del contexto narrativo que proporcione a través de un fichero de texto en formato PDF/TXT/DOCX, reciba un listado de ideas relacionadas con las siguientes temáticas: Personajes, localizaciones y eventos históricos.

A pesar de que la arquitectura del sistema del proyecto y el trabajo no ha cambiado mucho, las tareas que se planificaron inicialmente no fueron exactamente las mismas que las que se realizaron. Concretamente todas aquellas relacionadas con el conjunto de datos y el proceso de *fine-tuning*, por una parte debido a la complejidad que supone realizar este procesamiento en modelos extensos de lenguaje y por otra parte gracias a la gran cantidad de modelos entrenados con las características deseadas que se encuentran accesibles en páginas como la de *HuggingFace*. Por lo tanto, el tiempo requerido para las tareas relacionadas se redujo considerablemente en comparación con la estimación.

Durante la realización del trabajo, se llevó a cabo un proceso de aprendizaje sobre los modelos de lenguaje grandes (LLM). Este aprendizaje fue tanto autodidacta como a través de varias asignaturas finales de la especialización. Esto retrasó significativamente el avance del proyecto. No obstante, el conocimiento adquirido permitió identificar correctamente la arquitectura de sistema en dos componentes diferenciados y facilitó enormemente la progresión. Tal y como se muestra en la figura 1, el prototipo consta de un componente que aloja los modelos extensos de lenguaje y un servidor web que aporta la interfaz gráfica al usuario y procesa sus peticiones hacia el modelo de inteligencia artificial.

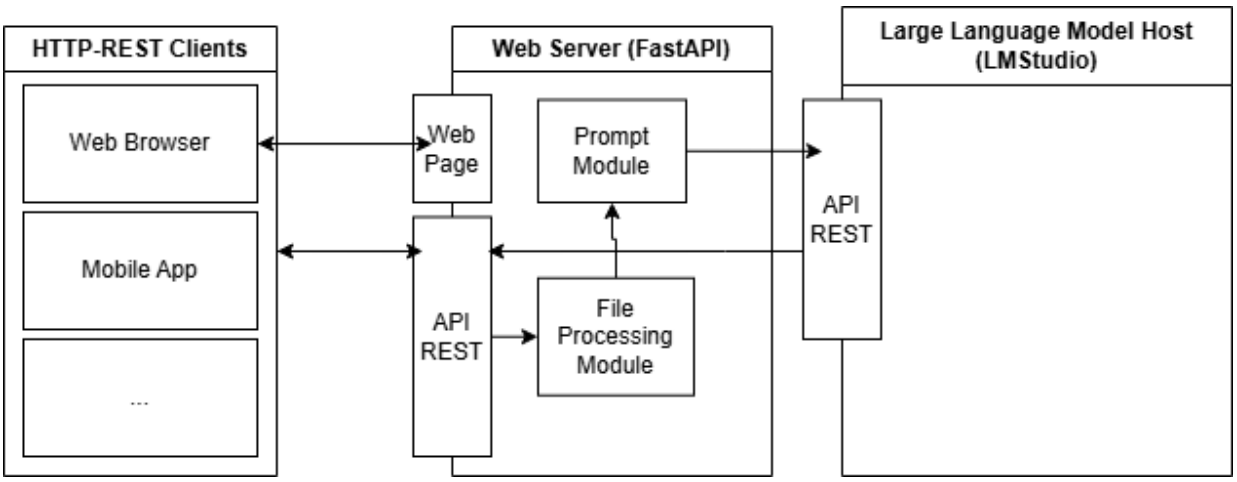

Figura 1. Arquitectura del sistema.

Al momento de redactar este informe, ambos componentes del sistema han logrado un avance considerable y todo el flujo de datos ya se encuentra completamente integrado.

El servidor web proporciona una página HTML simple en la que el usuario puede enviar un comando a través de un cuadro de texto y puede adjuntar archivos en los formatos previamente mencionados. La petición primero se convierte a texto plano mediante un procesador de archivos. Luego, esta información se fusiona con un prompt diseñado específicamente para el servicio solicitado. La instrucción se envía entonces al host de modelos extensos de lenguaje y devuelve la respuesta al usuario. Actualmente, estamos trabajando en la especificación de prompts definitivos que garanticen una mayor completitud y coherencia en las respuestas.

A su vez, se ha utilizado el programa LM Studio como herramienta para trabajar con los modelos extensos de lenguaje. Este programa permite administrar los modelos locales de la computadora, añadir nuevos a través de una interfaz de descarga y hacer uso del hardware disponible para arrancarlos de forma transparente para el usuario. Además, cuenta con varias ventanas de configuración de parámetros de lanzamiento del modelo, prompts personalizados, salida estructurada a través de un esquema JSON, opciones relacionadas con la aleatoriedad y temperatura de la inferencia y otras características experimentales. Aunque LM Studio ofrece un SDK en Python para interactuar mediante programación, se optó por utilizar su interfaz REST. La razón es que así el servidor web que usa LM Studio no depende exclusivamente de él. Al emplear un protocolo de comunicación estándar como REST, el servidor web puede comunicarse e interactuar fácilmente con cualquier otro servicio que aloje modelos de lenguaje grandes, lo que le da mayor flexibilidad.

Con la fase de desarrollo casi terminada, las tareas restantes se centran en la configuración, la mejora de los prompts, la selección de los modelos de datos y la definición del banco de pruebas para evaluar los resultados. Se calcula que esta fase durará entre cuatro y seis semanas, lo que encaja con el plazo de los talleres finales A y B. De este modo, todo el trabajo estará listo para la presentación final una vez finalicen ambos talleres.

2. Avance en las tareas

a) Indicar a continuación para cada una de las tareas su estado de situación según su criterio, utilizando verde si considera que es satisfactorio, amarillo si considera que es insatisfactorio por sobrecostos y/o demoras, y rojo si lo considera muy insatisfactorio por sobrecostos y/o demoras.
Si a la fecha de completar este informe no está previsto que la tarea haya comenzado entonces deje la celda correspondiente en blanco, sin pintarla con ningún color.

En subcelda inferior izquierda colocar:

- ** si los recursos u horas utilizadas fueron o están siendo muy inferior a lo planificado.
- * si los recursos u horas utilizadas fueron o están siendo inferior a lo planificado.
- $ si los recursos u horas utilizadas fueron o están siendo de acuerdo a lo planificado.
- $$ si los recursos u horas utilizadas fueron o están siendo superior a lo planificado.
- $$$ si los recursos u horas utilizadas fueron o están siendo muy superior a lo planificado.

En subcelda inferior derecha colocar:

- -- si la tarea se ejecutó o se está ejecutando mucho más rápido de lo previsto
- - si la tarea se ejecutó o se está ejecutando más rápido de lo previsto
- = si la tarea se ejecutó o se está ejecutando en el tiempo previsto.
- + si la tarea se ejecutó o se está ejecutando con demoras.
- ++ si la tarea se ejecutó o se está ejecutando con demoras muy significativas.

IMPORTANTE: Indicar con borde grueso las tareas que forman parte del camino crítico

1.1 Desarrollo inicial del servidor web y página principal		1.2 Implementación del modelo de datos y ORM		1.3 Implementación de pruebas automáticas		1.4 Validación de pruebas automáticas	
$	=	*	-	**	--	**	--
1.5 Implementación del comando para reentrenar el módulo LLM		1.6 Implementación de los servicios API REST del paquete MVP		1.7 Implementación de servicios REST adicionales			
*	-	$	=				
2.1 Obtención y análisis inicial del dataset		2.2 Desarrollo del software necesario para el tratamiento de los datos		2.3 Implementar pipeline de procesamiento del dataset		2.4 Fine-tuning del módulo LLM inicial	
$	++	**	--	**	--	**	--

2.5 Implementación de los métodos del módulo LLM para el paquete MVP		2.6 Presentación al cliente y análisis inicial de los resultados		2.7 Mejora del pipeline de procesamiento del dataset		2.8 Fine-tuning final del módulo LLM	
$$	=	$	++	$	=	$	=
2.9 Presentación al cliente y análisis final de los resultados		2.10 Implementación de métodos del módulo LLM adicionales					
3.1 Plan de proyecto		3.2 Presentación del plan de proyecto		3.3 Manual de administrador		3.4 Documento de análisis del dataset	
$	=	$	=				
3.5 Recopilación de informes de seguimiento del módulo LLM		3.6 Memoria técnica		3.7 Presentación y defensa del trabajo final			
		$	++				

Como se aprecia en la tabla, el proyecto experimentó retrasos significativos al principio, causados principalmente por la falta de conocimientos necesarios para el desarrollo. Esto impactó el camino crítico desde una fase muy temprana, generando demoras en la mayoría de tareas. Sin embargo, una vez superada esta barrera el resto progresó dentro de lo esperado, especialmente aquellas que se resolvieron de forma natural tras identificar LM Studio como la solución para el módulo LLM. Es importante mencionar también que los requisitos de hardware resultaron ser mayores de lo estimado, lo que implicó una inversión económica considerable, tal como se resalta en la tarea 2.5.

Las tareas de fine-tuning (ajuste fino) se llevaron a cabo finalmente mediante ingeniería de prompts. Esta técnica, aprendida recientemente, resultó ser más eficiente y cumplió mejor con los requisitos y limitaciones del proyecto.

Las tareas 1.7 y 2.10 no se realizarán, ya que son opcionales y exceden los límites de tiempo establecidos para el proyecto.

3. Cumplimiento de los requerimientos

a) Indicar a continuación para cada uno de los requerimientos el estado de situación según su criterio, utilizando verde si considera que ya se ha cumplido, amarillo si considera que aún no se a cumplido pero se podrá cumplir, y rojo si considera que aún no se ha cumplido y tiene dudas si se podrá cumplir.

Si considera que es necesario modificar los requerimientos respecto a los indicados en la planificación inicial entonces incluya acá los requerimientos actualizados, **marcando en negrita** aquellos que son nuevos o se han modificado.

Req #1.1: El servidor debe alojar y administrar la información relativa al dataset y la configuración del modelo LLM
Req #1.2: El servidor debe contar con los **prompts** necesarios para especializar **la respuesta de** la Inteligencia Artificial.
Req #1.3: Al ser desplegado, el servidor deberá **acceder al** módulo LLM y utilizarlo en el procesamiento de peticiones entrantes
Req #1.4: El servidor debe dar acceso a clientes externos a través del protocolo API REST.
Req #1.5: Los servicios REST deben aceptar entrada de texto en varios formatos: pdf, txt, word o texto plano en el cuerpo de la petición.
Req #1.6: Los servicios REST devolverán la respuesta en formato simple HTML para su cómoda visualización en un navegador.
Req #1.7: El prototipo del servidor debe de tener una disponibilidad del 100\% durante las pruebas y aceptar la carga de trabajo de peticiones individuales.
Req #2.1: El módulo LLM debe aceptar entrada de texto y generar texto como salida.

Req #2.2: El módulo LLM, si el texto de entrada es legible, debe aportar una respuesta con un detalle y profundidad razonables, además de ser coherente con las instrucciones recibidas.
Req #2.3: El módulo **se ajustará a los prompts recibidos para que, con el mismo contexto**, devuelva información enfocada en un aspecto específico de la narrativa.
Req #2.4: El tiempo de respuesta del módulo LLM debe estar, teniendo en cuenta el hardware con el que se dispone, la extensión del texto de entrada y que no se garantiza un rendimiento cercano a tiempo real, en un rango de tiempo razonable para un servicio REST (no más de 5 minutos).
Req #3.1: El proyecto contará con test automáticos que probarán el correcto funcionamiento de los módulos no relacionados con el modelo LLM.
Req #3.2: El cliente podrá validar los test automáticos a través de un análisis del código.
Req #3.3: Se validará con el cliente tanto el módulo LLM como el servidor.
Req #4.1: Se entregará un documento de administrador que explicará brevemente el sistema, qué comandos de ejecución tiene y los requisitos hardware necesarios para su correcto funcionamiento.
Req #4.2: Se entregará un documento con un breve análisis de los resultados del módulo LLM que se hayan expuesto al cliente durante las reuniones de seguimiento.
Req #5.1: Cualquier información del cliente que se utilice en el desarrollo del proyecto estará amparada por derechos de privacidad y propiedad intelectual
Req #5.2: El set de datos utilizado para el entrenamiento del módulo LLM debe tener una licencia libre o que se ajuste a uso académico.
Req #5.3: Todas las librerías de código empleadas en el desarrollo del proyecto deben tener una licencia de código abierto.

Los requisitos 1.2, 1.3 y 2.3 se han modificado para adaptarse a la nueva configuración de los prompts y a la independencia entre el módulo LLM y el módulo del servidor web.

4. Gestión de riesgos

a) Indicar a continuación para cada uno de los riesgos el estado de situación según su criterio, utilizando verde si considera que el riesgo ya no se manifestará o es muy improbable que se manifieste, amarillo si considera que es posible que es improbable que el riesgo se manifieste o si se manifiesta estima que será fácilmente controlado, y rojo si considera que es muy probable que el riesgo se manifieste y que no pueda ser controlado fácilmente.

Si considera que es necesario modificar los riesgos respecto a los presentados en la planificación inicial entonces incluya acá los riesgos actualizados, **marcando en negrita** aquellos que son nuevos o se han modificado, e indicando para ellos los valores de S, O y RPN, junto con su respectiva justificación.

Riesgo #1: No disponer de un modelo de tipo LLM pre entrenado que sea adecuado para el proyecto
Riesgo #2: Dataset inadecuado para el fine-tuning del modelo.
Riesgo #3: Capacidad hardware insuficiente
Riesgo #4: Retrasos en el proyectod
Riesgo #5: Resultado final insuficiente

Es importante señalar que los riesgos 3 y 4 se materializaron con la severidad prevista. Aunque no comprometieron el éxito del proyecto, sí afectaron de forma significativa el plazo para finalizar el trabajo de la carrera de especialización.

Actualmente no se estima que ningún riesgo más se materialice.