udp

Tarea 1: Plataforma de análisis de preguntas y respuestas en Internet

Profesor: Nicolás Hidalgo
Ayudantes: Isidora González, César Muñoz Rivera, Natalia Ortega y Joaquín Villegas

Introducción y Contexto

Yahoo! Answers, conocido en el ámbito hispanohablante como Yahoo! Respuestas, fue una destacada plataforma comunitaria de preguntas y respuestas. Su funcionamiento se centraba en la interacción de los usuarios, quienes podían plantear interrogantes sobre variados temas y recibir diversas soluciones aportadas por otros miembros de la comunidad. Un sistema de votación y selección permitía que las respuestas más valoradas ganaran importancia, estableciendo así un mecanismo de validación colectiva del conocimiento.
En el contexto académico actual, y tras el cese de operaciones de dicha plataforma durante la pandemia, se ha identificado una oportunidad para revisitar su valioso repositorio de preguntas. El presente proyecto surge de la propuesta de replicar una parte de su funcionalidad a través de la aplicación de tecnologías de inteligencia artificial, específicamente mediante el uso de un **LLM** *(Large Language Model)*.
Nuestro interés consiste en crear una plataforma escalable que permita realizar el análisis de las preguntas/respuestas generadas por humanos y las generadas por un LLM. Dicho análisis debe apuntar a la capacidad del modelo LLM de replicar respuestas humanas.

Metodología

El desarrollo del proyecto se estructurará de manera modular a lo largo del semestre, articulándose en tres entregas principales. Cada una de estas entregas se centrará en un objetivo específico, aunque no serán mutuamente excluyentes, se espera que los módulos o servicios desarrollados en fases anteriores puedan ser reutilizados y adaptados según las necesidades de las entregas posteriores.

La evaluación de cada etapa se fundamentará en la presentación de un **informe técnico** exhaustivo. Dicho documento deberá detallar el proceso de diseño e implementación, incluyendo las métricas de rendimiento, la comparación de resultados, un análisis crítico de la solución propuesta y una sección de preguntas y respuestas. Es un requisito fundamental que todas las decisiones de diseño y arquitectura estén debidamente justificadas y que el comportamiento del sistema esté respaldado por datos empíricos obtenidos durante la fase de evaluación. Adicionalmente, se deberá entregar un **video de demostración** que muestre el funcionamiento de los servicios implementados en cada hito.

Las tres entregas programadas abordan los hitos clave del ciclo de vida del proyecto, definidos de la siguiente manera:

1. **Datos y arquitectura base:** Abocados en la recuperación, gestión y almacenamiento eficiente de los datos y eventos que alimentan el sistema.
2. **Procesamiento y Fallback:** Centrado en la preparación y transformación de los datos para su posterior análisis. Esta etapa incluye la implementación de colas de procesamiento y mecanismos de *fallback* para gestionar solicitudes que presenten errores.
3. **Visualización:** Abocado a la creación de una interfaz o *dashboard* que provea una vista agregada y comprensible de las métricas más relevantes del sistema.

Para garantizar la portabilidad y reproducibilidad del entorno de desarrollo, se utilizará **Docker**[1] como tecnología de virtualización de recursos para el despliegue de los distintos módulos. No se impondrá un *stack* tecnológico específico. El equipo de desarrollo tendrá la libertad de seleccionar las herramientas, lenguajes y metodologías que considere más apropiados, siempre y cuando cada elección esté justificada en el informe correspondiente.

[1] https://www.docker.com/

Datasets y LLM

Conjunto de Datos (Dataset)

Para la implementación de este proyecto, se utilizará el conjunto de datos de Yahoo! Respuestas disponible en la plataforma Kaggle[2]. Este dataset se distribuye en archivos de valores separados por comas (CSV), los cuales contienen cuatro columnas: un índice de clase (del 1 al 10), el título de la pregunta, el contenido de la misma y la mejor respuesta seleccionada por la comunidad. Dado que el alcance de este proyecto no contempla el entrenamiento de un modelo de aprendizaje automático (Machile Learning), se podrá utilizar indistintamente el conjunto de entrenamiento (`train.csv`) o el de prueba (`test.csv`) como fuente principal de datos.

Large Language Model (LLM)

Se han evaluado diversas alternativas para la selección del LLM que generará las respuestas. A continuación, se describen las opciones disponibles para el equipo de desarrollo:

- **Servicio Gestionado:** Se recomienda el uso de la API de Gemini, provista por Google[3]. Esta plataforma ofrece un nivel de uso gratuito (*free tier*) que resulta adecuado para el desarrollo del proyecto[4]. No obstante, es fundamental que el equipo revise la documentación oficial para comprender los *rate limits* y optimizar su implementación en consecuencia.
- **Modelos Locales:** Como alternativa, es posible ejecutar un modelo de lenguaje de código abierto de forma local utilizando herramientas como Ollama[5]. Esta opción elimina las restricciones de uso por tokens o número de solicitudes, pero su rendimiento dependerá directamente de la capacidad del hardware local donde se ejecute.
- **Servicios de Pago:** Finalmente, existe la posibilidad de utilizar APIs de modelos comerciales de pago. Sin embargo, esta opción no es recomendada, ya que cualquier costo monetario asociado deberá ser asumido íntegramente por los miembros del equipo.

[2] `https://www.kaggle.com/datasets/jarupula/yahoo-answers-dataset`
[3] `https://aistudio.google.com/welcome`
[4] `https://ai.google.dev/gemini-api/docs/rate-limits`
[5] Guía de implementación de Ollama: `https://medium.com/@gabrielrodewald/running-models-with-ollama-step-by-step-60b6f6125807`

Entregable 1: Datos, Cache, y Calidad de respuestas

El objetivo principal de esta primera entrega es el diseño y desarrollo de un sistema que nos permita comparar y analizar la calidad de las respuestas generadas por un LLM a preguntas históricas planteadas por usuarios de Internet. Para ello, haremos uso de un LLM a elección, y el conjunto de preguntas y respuestas de la plataforma Yahoo! Answers. El sistema definido para esta entrega consiste de 4 módulos claves: generador de tráfico, sistema de cache, score/calidad, y sistema de almacenamiento.
La interacción entre los componentes se presenta en la Figura 1. El funcionamiento de la plataforma puede resumirse en 4 simples pasos. 1) el módulo generador de tráfico debe seleccionar aleatoriamente del conjunto de datos de referencia una pregunta e inyectarla a modo de consulta al sistema siguiendo una tasa de consulta generada a partir de una distribución a elección. 2) La consulta generada debe ser enviada al sistema de cache para validar si esta fue realizada previamente. En caso de estar presente en cache, se actualiza el almacenamiento contabilizando un nuevo acceso. En caso de no estar presente en el cache, la consulta debe ser entregada al módulo Score/calidad. 3) El módulo Score/Calidad debe consultar al LLM y obtener una respuesta la cual debe ser manejada y comparada usando una métrica a definir para cuantificar la calidad de la respuesta con respecto a la mejor respuesta del conjunto de datos de Yahoo!. 4) Finalmente, la pregunta, ambas respuestas, el valor de la métrica y las veces que se ha realizado la pregunta deben ser guardados en el sistema de almacenamiento.

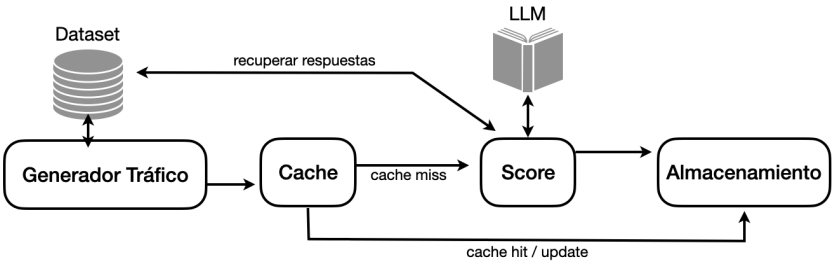

Figura 1: Enter Caption

Para la correcta evaluación de esta entrega, el sistema deberá cumplir con los siguientes objetivos específicos:

- **Selección y Referencia de Datos:** Se debe utilizar un conjunto de datos público de Yahoo! Answers, las cuales deberán ser debidamente documentados y referenciados en el informe.
- **Generación de Tráfico Sintético:** Se debe desarrollar un componente para la generación automática de consultas. Este módulo deberá extraer preguntas del sistema de almacenamiento y simular tráfico hacia el pipeline descrito, siguiendo al menos dos distribuciones de tasa de arribo distintas, las cuales deberán ser definidas y justificadas por el equipo de desarrollo.
- **Implementación de Caché:** Se requiere la implementación de un sistema de caché que intercepte los eventos del generador de tráfico. Este sistema deberá identificar y almacenar en memoria las respuestas a consultas repetitivas para optimizar los tiempos de entrega. Los parámetros de configuración de la caché (ej. tamaño, política de evicción) deberán ser definidos y justificados a partir de un análisis experimental.
- **Métrica de calidad de respuesta**: Se requiere la definición y justificación de una métrica de calidad de las respuestas, cuyo objetivo sea comparar la cercanía de las respuestas entregadas por el LLM con las respondidas por los usuarios de la plataforma de Yahoo!.
- **Modelo de Persistencia:** Es necesario implementar un sistema de almacenamiento de datos capaz de guardar la información: la pregunta original, la respuesta original del dataset (mejor respuesta), la respuesta generada por el LLM, el puntaje entregado por la métrica de calidad, y el conteo de veces que se ha repetido la pregunta. El sistema de almacenamiento deberá contener, como mínimo, 10.000 de registros.

Requerimientos y Organización del Sistema

El sistema a desarrollar deberá cumplir con los siguientes requisitos funcionales y de arquitectura, diseñados para garantizar su modularidad, eficiencia y escalabilidad.

- **Diseño Modular**
  - La arquitectura del sistema deberá estar estructurada en módulos independientes y cohesivos, facilitando su mantenimiento, escalabilidad y la futura integración con componentes externos.

- Se definen cuatro servicios principales que se comunicarán de manera secuencial:
  1. *Generador de Tráfico:* Simula las solicitudes de los usuarios al sistema, iniciando el flujo de consulta.
  2. *Caché:* Captura las solicitudes entrantes. Si la respuesta a una pregunta ya se encuentra almacenada en memoria, actualiza el sistema de almacenamiento. De lo contrario, delega la solicitud al siguiente módulo del pipeline.
  3. *Generador de Respuestas:* Recibe las preguntas que no fueron resueltas por la caché, las procesa y las envía al LLM para obtener una respuesta actualizada.
  4. *Score:* evalúa cercanía de las respuestas obtenidas desde el LLM con la mejor respuesta del dataset de Yahoo!
  5. *Almacenamiento:* Persiste de forma permanente la información relevante, incluyendo la pregunta original, su respuesta del dataset y la nueva respuesta generada por el LLM.

- **Conjunto de Datos y Consultas**
  - El sistema deberá operar sobre un conjunto de datos base de, como mínimo, 10.000 pares de preguntas y respuestas.
  - El volumen y la naturaleza de las consultas simuladas por el generador de tráfico deberán ser congruentes con la cantidad y variedad de los datos disponibles para consultar.

- **Sistema de Caché**
  - Se debe diseñar e implementar un sistema de caché optimizado para consultas frecuentes. La evaluación de su rendimiento es un requisito clave.
  - Dicha evaluación deberá incluir un análisis comparativo de, al menos, dos políticas de remoción (ej. LRU, LFU, FIFO), el efecto de variar el tamaño de la caché, TTL, etc.
  - Asimismo, se analizará el comportamiento del sistema bajo las diferentes distribuciones de tráfico implementadas. Todas las decisiones de parametrización final deberán estar justificadas con base en los resultados experimentales obtenidos.

- **Distribución y Despliegue**
  - El sistema completo deberá estar en contenedores utilizando Docker. Esto asegura la portabilidad del entorno y simplifica el proceso de despliegue y escalabilidad de los servicios.

- **Documentación y Buenas Prácticas**
  - Se deberá entregar una documentación técnica exhaustiva que abarque tanto la funcionalidad implementada como el código fuente[6]. Esta incluirá una justificación rigurosa de las decisiones de diseño, la elección de tecnologías y las metodologías de desarrollo aplicadas.

[6]Se requiere que el código esté alojado y documentado en un sistema de control de versiones, como GitHub o GitLab, siguiendo las buenas prácticas de la industria (ej. README.md detallado, comentarios en el código, etc.).

Análisis y Discusión

En esta sección, se debe realizar un análisis crítico y una discusión fundamentada sobre las decisiones de diseño y los resultados obtenidos. El análisis debe estar respaldado por datos empíricos, gráficos y tablas comparativas. A continuación, se detallan los puntos que deben abordar obligatoriamente:

Análisis del Comportamiento de la Caché

Impacto de la Distribución de Tráfico

Deben analizar y comparar el rendimiento del sistema de caché bajo las distintas distribuciones de tráfico que implementaron. Es necesario que respondan preguntas como:

- ¿Cómo varía la tasa de aciertos (*hit rate*) y de fallos (*miss rate*) entre una distribución y otra?
- ¿Qué distribución de tráfico genera un mayor estrés o beneficio para la caché y por qué?
- ¿Qué implicaciones tienen estos resultados para un escenario de aplicación real?

Efecto de los Parámetros: Tamaño y Política de Remoción

Deben presentar un análisis experimental sobre la configuración de la caché. Para ello, tienen que:

- Evaluar y comparar el rendimiento de, al menos, dos políticas de remoción (ej. LRU, LFU, FIFO).
- Analizar cómo el tamaño de la caché afecta su rendimiento.
- Justificar, con base en los datos obtenidos, la elección final de la política y el tamaño de la caché para su sistema, discutiendo los pros y contras de su decisión.

Análisis y Discusión de una Función de Puntuación (Score)

Para esta sección, deben seleccionar e implementar al menos una función o métrica de puntuación (*score*) que permita evaluar y comparar la calidad de las respuestas generadas por el LLM frente a las respuestas originales del dataset. Una vez seleccionada, deben aplicarla a un subconjunto de sus datos y realizar un análisis crítico de los resultados.
La discusión debe estar centrada en los siguientes puntos:

- **Justificación de la Métrica Seleccionada:** Deben justificar por qué eligieron esa función de score específica (ej. similitud de coseno sobre embeddings, ROUGE, BERTScore, etc.). Expliquen qué aspecto de la "calidad" de una respuesta mide (ej. superposición léxica, equivalencia semántica, etc.) y por qué es relevante para este problema.
- **Utilidad y Limitaciones de la Métrica:** Con base en sus resultados, discutan la utilidad práctica de la métrica implementada. ¿Consideran que el score es lo suficientemente fiable como para ser usado en el sistema global para tareas como filtrar respuestas de baja calidad o comparar el rendimiento de diferentes LLMs? ¿Cuáles son las limitaciones evidentes de la métrica que seleccionaron?

Análisis del Sistema de Almacenamiento

Deben realizar un análisis detallado de la tecnología de almacenamiento de datos que eligieron. Es fundamental que:

- Justifiquen rigurosamente por qué seleccionaron esa base de datos (ej. SQL vs. NoSQL, y el motor específico como PostgreSQL, MongoDB, etc.).
- Relacionen su elección con los requisitos específicos del proyecto, como el modelo de datos, la escalabilidad esperada, los patrones de consulta y la facilidad de integración en una arquitectura de contenedores (Docker).

Requisitos de la Entrega

La evaluación de este hito se basará en los siguientes entregables:

- **Informe Técnico:** Un documento en formato LaTeX (entregado como PDF) que describa de manera concisa y rigurosa la arquitectura y el enfoque del sistema. El informe debe centrarse en el **análisis crítico** que justifique las decisiones tomadas y los resultados obtenidos. El contenido debe incluir:
  - Una descripción detallada de los componentes y las funcionalidades implementadas en cada módulo del sistema.
  - Una justificación fundamentada de las decisiones de diseño, así como de las tecnologías y metodologías empleadas.
  - Un análisis exhaustivo de los resultados, respaldado por datos empíricos. Se deben utilizar gráficos, tablas comparativas y otros recursos visuales para facilitar la interpretación y evaluación de las métricas definidas.
- **Video de Demostración:** Un video con una duración de 10 minutos donde se muestre el sistema en funcionamiento, explicando sus componentes y el flujo de datos.
- **Código Fuente:** El código fuente completo del proyecto, el cual deberá estar alojado en un repositorio público en GitHub o GitLab. El enlace al repositorio deberá incluirse tanto en la sección de comentarios de la plataforma CANVAS como en el informe técnico.
- **Archivos de Despliegue:** Un archivo `Dockerfile` y/o `docker-compose.yml` que permita la construcción y ejecución de todos los servicios implementados. El archivo `README.md` del repositorio deberá contener instrucciones claras y precisas para el despliegue y la ejecución del sistema.

Reglas y consideraciones de la entrega

- **Fecha de Entrega:** La fecha límite para la entrega de esta tarea es el día 01/10/2025 hasta las 23:59 hrs. Se recomienda gestionar el tiempo de forma efectiva. La entrega es vía Canvas del curso.
- **Integrantes:** La tarea debe ser realizada en grupos de hasta **2** alumnos/as.
- **Ética y Autoría:** Se debe respetar el reglamento de la universidad en cuanto a plagio y autoría. Cualquier evidencia de copia o falta de autoría conllevará sanciones según lo establecido.