# Smart History Tutor

Bruno Jesús Pire Ricardo

Grupo C311, Facultad de Matemáticas y Computación, Universidad de La Habana, Cuba

**Abstract.** El proyecto *Smart History Tutor* se desarrolló con el objetivo de diseñar e implementar un sistema inteligente orientado a responder preguntas sobre Historia Universal. Para ello, se empleó una arquitectura multiagente alineada con el modelo *Retrieve-Augmented Generation* (RAG), que combina técnicas de recuperación semántica con generación de texto. El sistema integra agentes especializados para la recuperación de información mediante índices FAISS, la generación de respuestas utilizando la API Google Gemini y la mejora de consultas mediante prompts adaptados. Asimismo, se implementó un algoritmo de chunking basado en metaheurísticas para optimizar la segmentación de documentos históricos obtenidos mediante un crawler de Wikipedia en español. La solución propuesta permite refinar preguntas, evaluar la suficiencia del contexto y generar respuestas claras y contextualizadas. Los resultados alcanzados muestran la viabilidad de la arquitectura planteada para gestionar grandes corpus históricos y ofrecer asistencia educativa.

**Keywords:** RAG, multiagente, recuperación de información, metaheurísticas, FAISS, API, Google Gemini, crawler, Wikipedia

## 1 Introducción

El proyecto *Smart History Tutor* se concibió con el propósito de implementar un sistema inteligente capaz de responder preguntas relacionadas con Historia Universal. La motivación principal que impulsó el desarrollo de esta solución fue la posibilidad de obtener de manera eficiente un corpus amplio y especializado en dicha temática, superando uno de los desafíos más comunes en este tipo de trabajos: la dificultad de reunir, estructurar y gestionar grandes volúmenes de información histórica fiable.

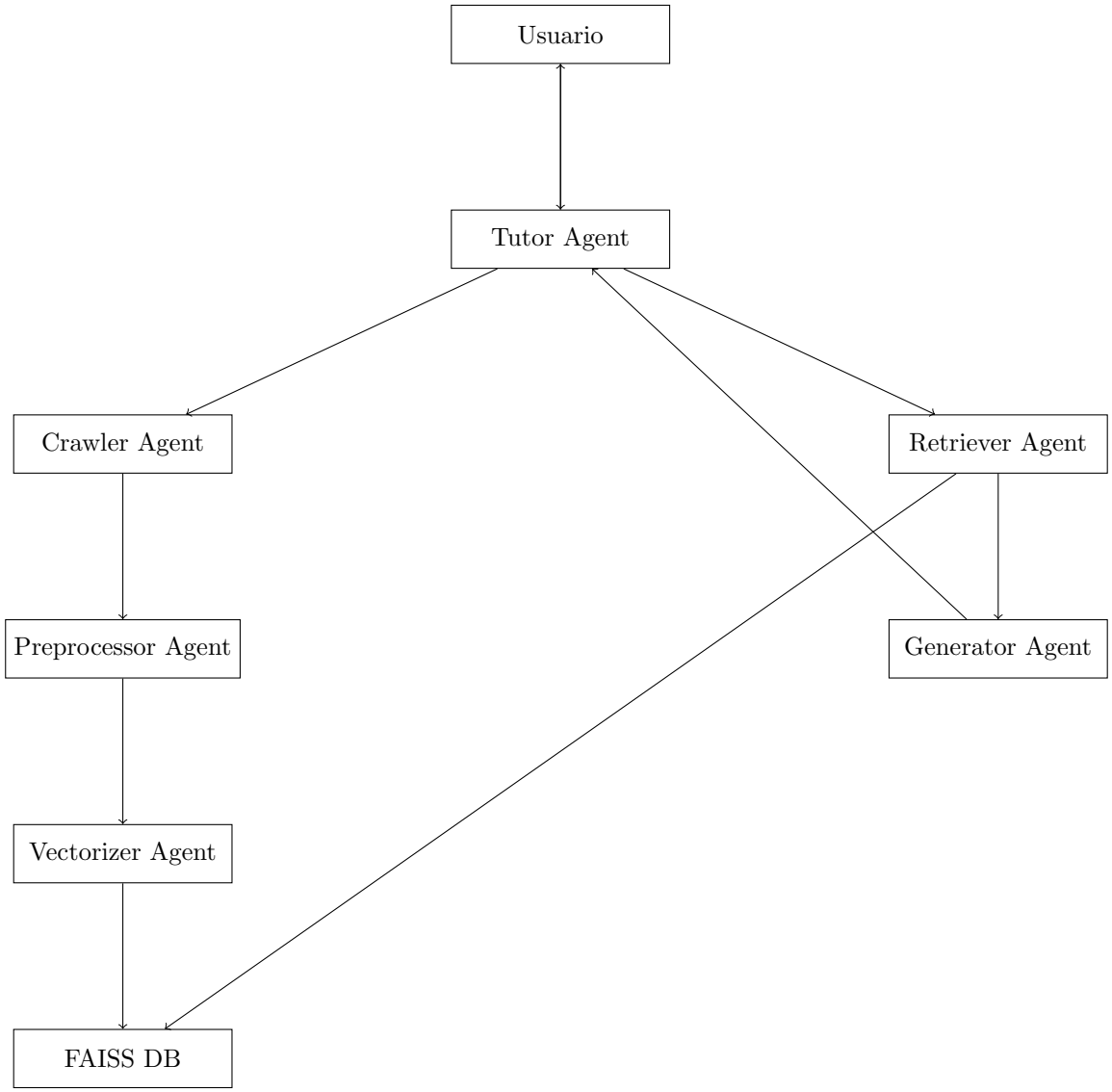
Para construir el sistema, se recurrió a Wikipedia en español como fuente primaria del corpus, aprovechando su amplia cobertura de temas históricos y su disponibilidad abierta. El contenido fue procesado y segmentado mediante técnicas de *chunking*, lo que permitió alcanzar un total aproximado de 9000 fragmentos con información contextualizada.

El alcance de esta propuesta se centró en establecer las bases de una herramienta que pudiera servir como soporte en el aprendizaje de Historia Universal para estudiantes de enseñanza media superior. Se planteó un diseño que pudiera integrarse en futuros sistemas educativos y plataformas de tutoría virtual, garantizando un acceso ágil a información histórica de calidad.

## 2 Soluciones implementadas

El sistema adoptó una arquitectura multiagente alineada con el modelo *Retrieve-Augmented Generation* (RAG), diseñada para integrar procesos de adquisición, procesamiento, recuperación y generación de información. El componente central, el *Tutor Agent*, coordinó a los módulos especializados:

- *Crawler Agent*: recopilación de artículos desde Wikipedia.
- *Preprocessor Agent*: limpieza, tokenización y generación de chunks optimizados por metaheurística.
- *Vectorizer Agent*: extracción de embeddings y almacenamiento en FAISS.
- *Retriever Agent*: recuperación semántica usando FAISS.
- *Generator Agent*: verificación de contexto y generación de respuesta con Gemini.




## 3 Consideraciones y justificación teórico-práctica

Se eligió un modelo de embeddings de complejidad moderada para facilitar su ejecución en hardware limitado, aunque esto implicó mayores tiempos de procesamiento para lotes grandes. La primera aproximación con modelos QA extraían citas textuales sin suficiente argumentación, y los modelos generativos locales (Tiny LLaMA) no fueron adecuados para el contexto complejo requerido. La API de Google Gemini se seleccionó por su capacidad de manejar volúmenes grandes de contexto y flexibilidad, manteniendo costos bajos durante la fase

experimental. El uso de metaheurísticas para el chunking permitió abordar el problema como uno de satisfacción de restricciones, demostrando conocimientos en IA y simulación.

## 4 Evaluación cuantitativa y cualitativa

El sistema gestionó un corpus de 9000 fragmentos procesados. Los tiempos de recuperación y generación fueron adecuados para pruebas académicas, aunque la vectorización y almacenamiento presentaron demoras con lotes extensos debido al modelo de embeddings elegido. Cualitativamente, el flujo robusto entre agentes permitió generar respuestas históricas argumentadas, mejorando respecto a versiones basadas en QA tradicionales y subrayando la modularidad del diseño.

## 5 Autocrítica: logros, insuficiencias y propuestas de mejora

El sistema obtuvo respuestas correctas y fundamentadas en preguntas de tipo factual y biográfico, demostrando la capacidad de recuperar y contextualizar información. Se consideró como fortalezas el flujo multiagente, la gestión del corpus fragmentado y la calidad argumentativa de las respuestas generadas con la API de Google Gemini. Además, la arquitectura modular facilitó la integración de componentes y su posible ampliación.

Entre las limitaciones identificadas se encuentran el rendimiento en preguntas complejas que demandan un contexto más amplio y específico, así como la necesidad de gestionar los límites de uso de la API gratuita para evitar interrupciones. Asimismo, el uso de Wikipedia como fuente principal del corpus, si bien resultó práctico para obtener información de forma rápida y masiva, representa una limitación en cuanto a la credibilidad y rigor de los datos históricos empleados. Como mejora, se sugiere la extracción futura de información desde documentos históricos especializados o bases de datos respaldadas por profesionales en la materia, con el objetivo de incrementar la confiabilidad de las respuestas generadas.

Las propuestas de mejora incluyen la optimización de las estrategias de recuperación, la reducción de la dependencia de servicios externos mediante el empleo de modelos generativos locales, y la implementación de mecanismos más eficientes de control sobre los tokens y la selección de fragmentos en consultas complejas.

## References


1. Lewis, P. et al.: Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. NeurIPS 2020. arXiv:2005.11401
2. Jégou, H., Douze, M., Johnson, J.: Billion-scale similarity search with GPUs. arXiv:1702.08734

3. Douze, M. et al.: The FAISS Library. arXiv:2401.08281
4. Gao, Y. et al.: Retrieval-Augmented Generation for Large Language Models: A Survey. arXiv:2312.10997