ASISTENTE ACADÉMICO RAG

Sistema de Retrieval-Augmented Generation
para Consulta de Sílabos Académicos

Proyecto: Tarea RAG Personal

Fecha: 22 de October de 2025

Repositorio: https://github.com/ricdex/utec-ai/tree/main/tarea_rag_personal

Objetivo Principal:

Desarrollar un sistema inteligente basado en RAG que permita consultar información académica de sílabos en lenguaje natural. El sistema identifica cursos, ciclos académicos, metodologías, criterios de evaluación y bibliografía mediante búsqueda semántica combinada con coincidencia de palabras clave.

1. OBJETIVO DEL SISTEMA

Problema Resuelto: Las instituciones académicas almacenan información de cursos en archivos PDF (sílabos) desorganizados. Los estudiantes no pueden consultar esta información de forma natural y rápida. Un asistente RAG tradicional falla cuando las consultas son conversacionales o contienen términos genéricos.

Solución: Un asistente RAG inteligente que combina búsqueda semántica con recuperación híbrida, priorizando nombres exactos de cursos. El usuario puede hacer preguntas como:
• 'Dame detalle del curso de Investigación Operativa'
• '¿Cuál es la fórmula del promedio final?'
• '¿En qué ciclo se aborda gestión de proyectos?'

2. ARQUITECTURA DEL SISTEMA

2.1 Componentes Principales

Módulo de Carga (load_documents.py):
• Lectura de PDFs desde carpeta 'silabus'
• Chunking estructurado basado en secciones (I., II., III., etc.)
• Generación de embeddings (Ollama o OpenAI)
• Indexación en Chroma (BD vectorial)

Sistema RAG Principal (rag_system.py):
• Inicialización del recuperador híbrido
• Configuración de plantilla de prompt personalizado
• Cadena RAG: Recuperador → Prompt → LLM → Parser
• Visualización de chunks recuperados

Interfaz Interactiva (main.py):
• Loop de preguntas y respuestas
• Detección automática de BD (crear si no existe)
• Salida formateada con chunks visibles

3. RECUPERADOR HÍBRIDO (INNOVACIÓN PRINCIPAL)

El **RecuperadorHibrido** es la innovación clave que resuelve el problema de búsqueda semántica pura. Combina dos estrategias de recuperación:

Estrategia 1: Búsqueda Semántica (Base)

```
FUNCIÓN recuperar_documentos_semanticos(consulta):
  RETORNAR chroma.buscar_similitud(consulta, k=15)
  # Obtiene 15 documentos ordenados por similitud
```

Estrategia 2: Detección de Nombres de Cursos

```
FUNCIÓN extraer_palabras_clave(texto):
  PARA CADA palabra EN texto:
    SI longitud(palabra) > 2:
      agregar a lista_palabras_clave
  RETORNAR lista_palabras_clave

FUNCIÓN detectar_nombre_curso(palabras_clave):
  cursos_conocidos = ['investigación', 'operativa', 'ética', 'deontología', ...]
  SI alguna palabra_clave EN cursos_conocidos:
    RETORNAR verdadero
  RETORNAR falso
```

Estrategia 3: Re-ranking con Boost de Nombre

```
FUNCIÓN calcular_puntuacion(documento, palabras_clave, tiene_nombre_curso):
  SI tiene_nombre_curso:
    RETORNAR 1000.0 + puntuacion_palabras_clave
  SINO:
    posicion = indice_en_lista + 1
    RETORNAR (1.0 / posicion) + (puntuacion_palabras_clave * 0.5)

FUNCIÓN recuperar_hibrido(consulta):
  docs = recuperar_documentos_semanticos(consulta)
  palabras = extraer_palabras_clave(consulta)
  PARA CADA doc EN docs:
    tiene_nombre = detectar_nombre_curso(palabras)
    puntuacion = calcular_puntuacion(doc, palabras, tiene_nombre)
  ORDENAR docs POR puntuacion (descendente)
  RETORNAR docs[0:10]
```

Resultado: Consultas como 'dame detalle del curso de investigacion operativa' ahora retornan el sílabo correcto en posición 1, en lugar de otros cursos irrelevantes.

4. RESULTADOS Y VALIDACIÓN

Consulta	Posición Correcta	Estado
Investigación Operativa	1	✓
fórmula promedio final Inv. Operativa	1	✓
gestión de proyectos	Top 3	✓
evaluación en Ética y Deontología	1	✓
Redes de Comunicaciones ciclo	1	✓

Métricas de Rendimiento

• Tiempo de indexación: ~10 segundos (120 chunks)
• Tiempo de consulta: ~2-3 segundos (incluye LLM)
• Precisión en recuperación: 100% (5/5 consultas correctas)
• Tamaño del código: 191 líneas Python (sin test/diag)
• Modelos soportados: Ollama (local) y OpenAI API

Mejoras Implementadas

✓ Chunking estructurado por secciones del documento
✓ Recuperador híbrido con detección de nombres de cursos
✓ Prompt optimizado con instrucciones explícitas
✓ Soporte dual: Ollama (local) y OpenAI API
✓ Visualización de chunks recuperados para debugging
✓ Interfaz interactiva minimalista

5. CONCLUSIONES

Se desarrolló exitosamente un sistema RAG educativo que resuelve un problema real: la consulta inteligente de información académica. La innovación principal es el **Recuperador Híbrido**, que combina búsqueda semántica con priorización de nombres exactos de cursos. Esto permite al sistema responder consultas naturales como un asistente académico real.

Código disponible en: https://github.com/ricdex/utec-ai/tree/main/tarea_rag_personal