CS5481: Data Engineering - Projects

Instructions

1. Due at 11:59 pm, November 26, Tuesday, 2025.
2. This is the group project. Each group has up to 8 members. Please set up your group by 11:59pm, October 5, 2025 on Canvas-People-Groups.
3. You are required to submit the project report and source code via Canvas and give a 15-min presentation for your project in class. The project report should at least consist of following parts, including introduction, methodology, experiments and discussions. The report may contain up to 6 pages of main content, plus unlimited pages of references and appendix. The source code can be submitted by Jupyter Notebook or Python files.
4. Please attach your presentation slides at the end of the report.
5. Please state the individual contributions and ratios (%) in the report.
6. This project is very open, and you are highly encouraged to come up with fantastic ideas and designs!
7. If you have any questions, please post your questions on the Canvas-Discussion forum or contact our TAs.

Topic 1 LLM-Enhanced Content Filtering System

Traditional recommender systems rely on collaborative or content-based filtering. This project focuses on building a robust content-based filtering pipeline where item features are enriched using a Large Language Model (LLM). Instead of modeling user-user interactions, you will model item-item similarity based on rich, descriptive metadata.

Your task is to crawl detailed data for a specific domain (e.g., products, articles, video games). You will then use an LLM API to summarize descriptions, extract key features, and generate embeddings for each item. The core challenge is to design an efficient system to store this data and the generated vectors, and to retrieve similar items based on a user's selection quickly.

Key tasks:

1. **Data Ingestion:** Crawl and clean a dataset of at least 10,000 items with rich descriptive text.
2. **Feature Enrichment:** Use a pre-trained LLM API to generate structured tags, summaries, and vector embeddings for each item's description.
3. **Data Management:** Store the raw data, structured metadata, and vector embeddings in an appropriate database system (e.g., PostgreSQL with pgvector, or a dedicated vector database).
4. **Retrieval:** Implement a function that, given an item ID, retrieves the top-N most similar items using efficient vector similarity search.

Reference

1. Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.
2. Pinecone Documentation. (2024). *Vector Databases for Recommender Systems*.

Topic 2 Automated Topic Summary Page Generation

When a major event unfolds, information is often scattered across hundreds of articles and posts. This project challenges you to build a data pipeline that automatically crawls information about a specific event or topic and generates a consolidated, structured summary webpage. This goes beyond a simple timeline by creating a rich, multi-faceted overview.

You will choose an event, crawl data from various news sources over a period, and process this data to identify key sub-themes, involved entities (people, organizations), and a chronological summary. The final output should be a static HTML page generated by your pipeline.

Note that:

1. The final page should include: a main summary, a list of key entities, a timeline of major developments, and links to the original source articles.

2. Your pipeline must handle duplicate information from different sources, merging and synthesizing the content. An LLM API can be used for summarization and entity extraction.

Reference

1. Nenkova, A., & McKeown, K. (2012). A survey of text summarization techniques. In *Mining Text Data* (pp. 43-76). Springer.

2. Beautiful Soup Documentation. https://www.crummy.com/software/BeautifulSoup/bs4/doc/

Topic 3 Specialized Search Engine for Academic Papers

General-purpose search engines are powerful, but a specialized search engine can provide more relevant results for a specific domain. Your goal is to build a search engine exclusively for academic papers from a pre-print server like arXiv or a similar source.

This project requires a full-stack data engineering approach: crawling the data, creating a robust storage and indexing system, and building a simple query interface. The focus should be on implementing effective information retrieval techniques rather than complex UI.

You should consider three aspects:

1. **Massive Data Collection.** You must design a reliable crawler to download the metadata (title, authors, abstract) and preferably the full text of at least 100,000 academic papers.
2. **Efficient Indexing.** Store the collected data and build an inverted index to support fast keyword-based queries. You should implement a classic retrieval model like TF-IDF or BM25.
3. **Fast Data Query.** Develop a system that takes a text query and returns a ranked list of relevant papers from your index, prioritizing speed and accuracy. Direct SQL ‘LIKE‘ queries are not sufficient.

Reference

1. Apache Lucene Project. https://lucene.apache.org/
2. Robertson, S., & Zaragoza, H. (2009). The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval, 3*(4), 333-389.

Topic 4 Real-time Trend Analysis Dashboard

Keywords and topics can reveal societal trends, but their importance often changes rapidly. This project involves building a data pipeline to crawl social media or news data in near real-time, analyze it for emerging trends, and display the results on a live dashboard.

You will choose a specific domain (e.g., technology, finance, politics) and a data source (e.g., Twitter API, RSS feeds from news sites). Your system should continuously ingest data, process it to calculate term frequencies, identify sudden spikes or ”bursty” keywords, and push these statistics to a web-based dashboard that updates automatically.

Your pipeline should:

1. Continuously crawl and ingest data from a streaming or frequently updated source.
2. Process text data to remove stop words and perform normalization.
3. Implement an algorithm to detect trending terms based on frequency changes over time (e.g., comparing the last hour to the previous 24 hours).
4. Visualize the current trends and their intensity on a simple, auto-refreshing webpage or dashboard.

Examples of Visualization Tools

1. Plotly Dash. `https://dash.plotly.com/`
2. Streamlit. `https://streamlit.io/`

Topic 5 Knowledge Base Q&A System

LLMs are powerful, but they lack knowledge of specific, private, or recent documents. This project tasks you with building a Question-Answering system based on a custom knowledge base, using the Retrieval-Augmented Generation (RAG) pattern. This is a core data engineering task that enables LLMs to use external data.

The system should be able to answer questions based on a specific set of documents you provide. The main challenge is not the chatbot interface, but the backend data pipeline that prepares and serves the knowledge for retrieval.

Requirements:

1. **Data Preparation:** Collect a set of documents (e.g., course notes, product manuals, company reports). Write a script to load, chunk, and create vector embeddings for these documents using an open-source embedding model or API.
2. **Knowledge Base Storage:** Store the document chunks and their corresponding embeddings in a vector database.
3. **Retrieval Mechanism:** Given a user's question, first convert the question to an embedding, then retrieve the most relevant document chunks from the vector database based on semantic similarity.
4. **Answer Generation:** Pass the original question and the retrieved document chunks to an LLM API. Prompt the LLM to generate an answer based *only* on the provided context.

Reference

1. Lewis, P., et al. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems, 33*.
2. LangChain Documentation. (2024). *Question Answering over Documents*.

Topic 6 Open Your Mind

This is a very open-minded project that you can try anything you are interested and related to the course content. For this topic, the only requirement is that you should crawl massive data from the Internet, and then do something with the data. For example, you can predict the stock tendency based on the social news; you can build the social networks based on the interactions among users on social media platform. We highly recommend using ChatGPT to do some fun things. Just open your mind and take a try. Please contact TAs to verify your own selected topic before starting the group work.