How can we effectively and efficiently retrieve relevant information from historical newspapers?

Our task is to implement a search engine that is able of retrieving relevant historical newspaper passages from a large collection of digitized archives OCR. The project focuses on understanding how raw OCR text and its cleaned version affect the performance of an Information Retrieval System (IRS). To do this, we will explore two approaches:

1. Build the best IRS possible using the cleaned text.
2. Adapt and improve an IRS that must operate directly on the raw OCR text.

We will analyze how noise, transcription errors, and archaic language influence retrieval accuracy. We also aim to understand how much improvement text cleaning contributes when compared to the unprocessed OCR version. Intended users are people who work with or study historical material such as historians, librarians, journalists, researchers, students, and anyone wanting reliable access to historical facts and events.

Why the Task Is Challenging

The problem presents different challenges depending on the approach:

- **For approach 1 (clean text):**

  Analyzing the collection and identifying the best combination of retrieval models: lexical, neural, or hybrid; and applying advanced techniques such as query expansion or re-ranking to reach the highest possible effectiveness.

- **For approach 2 (OCR text):**

  Dealing with the imperfections of OCR data. Scans contain noise, spelling errors, and misrecognized characters. Cleaning or correcting this text is

essential to obtain retrieval performance closer to what we can achieve with cleaned data.

📌 Across both approaches, additional challenges include matching archaic vocabulary to modern queries, handling temporal references, and understanding how text quality impacts ranking behavior.

Why Solving This Task Is Important

- **Approach 1:**

  A strong retrieval system saves significant time by retrieving the most relevant documents quickly, allowing users to focus on analyzing information rather than searching for it.

- **Approach 2:**

  Looking for methods to clean the OCR provides insights into how noisy text influences system performance, and teaches us how to design retrieval pipelines that remain robust even when text quality is low.

📌 Overall, the task supports the broader goal of giving everyday users access to accurate, unfiltered historical information, which is essential for research, education, and understanding past events.

The starting point

We each came up with several ideas on how to approach this problem, but we all came togheter to define a final pipeline to experiment with in order to achieve the best possible retrieval system. We will always compare the results obtained on the clean documents to the ones obtained on the OCR text. If time permits we also had different ideas on how to approach the issue of cleaning the OCR text ourself:

- The simplest approach is to use a fine-tuned model specialized in cleaning OCR data.
- A less feasible option is to train a neural network on the available documents to generate clean text from OCR; however, this would likely require more time and computational resources that we have at our disposal.

Pipeline

Indexing Fields

- `year`: Extracted from publication date
- `extract_names`: Metadata containing only uppercase-starting names (for regex comparison with queries)
- `keywords`: Document expansion at indexing time
  - Keywords extraction via TF-IDF, KeyBERT, Part-of-speech filters
  - Synonym extraction (WordNet)
- `new_document_1` ***(to evaluate:** First document expansion)*
  - Hard swap of common "old parts of speech" extracted from training set using LLM
  - Append extracted keyword synonyms, replacing old terms with modern equivalents

---

Query Processing

- **Query Expansion** using:
  - **Pseudo-relevance feedback** (adding words to the query from “context”)
  - **Thesaurus-based expansion** ***(to evaluate:** if it is easy to get “old synonims” from e.g. wordnet, if not, discard)*
- **Weighted BM25 Scoring**:
  - 70% weight on context field
  - 30% weight on keywords field

    (Weights subject to evaluation)

First-Stage Retrieval

- **Method**: BM25 (Bi-encoder maybe too heavy)
- **Output**: Retrieve top 1000 documents

- **Fields**: Indexed fields with weighted scoring (primary focus on `context` and `keywords` fields, with later regex validation on the publication year, decide if hard match or higher score)

Second-Stage Retrieval

- **Reranking**: Cross-encoder *(or bi-encoder if too heavy)*
- **Output**: Reduce to top 100 documents

Optional Preprocessing for third-stage Retrieval *(to evaluate)*

- `new_document_2` *(Heavy computation, evaluate necessity)*
  - Second document expansion using LLM (e.g. rewrite the document with modern lexicon)
  - Followed by reranking

Final Reranking, third stage *(to evaluate)*

- **Output**: Select top 3 documents out of the previous 100 (maybe using cross-encoders)

Answer Generation

- **Option A**: Input top 3 documents to LLM for contextual answer generation
- **Option B**: Use DistilBERT to generate answer fine-tuned on training data