Social Science Search Demo

Sheikh Mastura Farzana

Requirements

- Demo search application using 10 documents from SSOAR repository.
- Documents associated with democracy, migration and Germany.
- Full-text search capacity with metadata search option.
- Usage of open standard software.

Data Acquisition

- Single request: static search URL on SSOAR that already contains keyword query (democracy+germany+migrant).
- 10 links max and URL deduplication.
- Results are not stored rather sent to the next module and processed within the pipeline.
- Custom header (SSOAR-Scraper/1.0) so the server knows who’s calling.

- Pagination: for more results, navigation to subsequent result pages is needed.
- Polite crawling: Right now, it is one request. In production, introduction of small random delays and retry if the server times out or blocks the crawler.
- Reusable class: This scraper is tightly coupled to SSOAR. Making it generic would let us plug in new sources.
- Save to disk: If the crawl session is saved, re-run of enrichment or indexing later without re-scraping is possible.

Enrichment: metadata extraction

- Parsed each document page using BeautifulSoup.
- Extracted metadata: title, authors, year, abstract, language, keywords, DOI, pdf link and source URL.
- Assembled into consistent JSON files for each document.

- Using document URLs for extracting metadata - fallback option. Initial idea was to use PDF content for metadata.
- Several PDF parsers were tested but none performed well at extracting metadata from PDF or clean main content. (Try out Apache Tika in future).
- LLMs could have been used to extract metadata from the PDF, skipped for now.
- Raw plain text has been extracted from PDFs and saved, was not used for the indexing as substantial effort was required to clean the data.

Enrichment: Geo-location Extraction (LLM)

- Extracted country information (name, coordinates, country code) from each document (abstract) using LLM.
- Additional metadata that has been used as a filter for search results.
- Location information for content is an interesting aspect to showcase.

- Use open source LLM such as Mistral using LM Studio in future (current laptop not good enough to handle local LLM).
- Project location data on a map to visualize content location information.

Enrichment: Summarizer

- Summarize the abstract in ≤ 2 sentences, ~40 words.
- Output is short, concise and displayed in search result preview.
- Follows primary language (English) of the search engine, German document summaries are also in English.

- Use open source LLM models.
- Detect and remove redundant lead-in phrases (“This paper discusses…”)
- Tune summarizer for multilingual abstracts, maybe translate to search engine language when user opts for it.

Indexing

- Created an Elasticsearch index called app_demo
- Defined the structure (mapping):
  - title, abstract, summary: searchable text
  - year, language, authors, country: exact match fields for filters
- Abstract has the highest importance followed by title and then summary (during query).

- Add full_text from PDFs to enable complete document search
- Add dense vector (embedding) to support semantic “similar meaning” queries
- Add analyzers (eg: stemmers) make search smarter (e.g. match “migration” and “migrating”)
- Move boosting into mapping, avoid repeating it in queries
- Use ingest pipelines - clean/normalize text at index-time

Frontend

- Built a clean search UI using Streamlit
- User enters a query → searches abstract^3, title^1.5, summary
- Top 5 results shown with:
    - Title (clickable → takes to PDF link)
    - 2-line summary
    - Keywords
- Sidebar filters based on the search results: Year, Language, Author, Country

- Replace Streamlit with React + Elastic UI for more control and design polish
- Add map view to visualize country mentions
- Add language switch or auto-translate query (e.g. DE → EN)
- Highlight search terms in results (abstract or summary)
- Add full-document view when link is clicked (title + abstract + summary + PDF text)

Knowledge Graph with RAG

- Build a Knowledge Graph from enriched metadata
  - Nodes: Documents, Authors, Countries, Keywords
  - Edges: “mentions”, “written by”, “published in”
- Use it to answer structured questions like:
  - “Which countries are most studied in 2023?”
  - “Find docs by Author X mentioning Germany”
- Use RAG (Retrieval-Augmented Generation) to answer free-text questions
  - Step 1: Retrieve relevant abstracts or summaries
  - Step 2: Use LLM to answer based on retrieved context
- Replace keyword search with semantic Q&A
  - e.g., “How do integration policies differ in France vs Germany?”

DEMO

Questions and Feedback