Q4: Intelligent Search for Document Corpus

Time: 2.5 hours

Problem Statement

Build a full-stack intelligent search system for **24K+ research papers/documents** that handles complex queries, finds connections across documents, and provides LLM-powered synthesis and insights.

Input Data

Download: https://www.kaggle.com/datasets/neelshah18/arxivdataset

Expected Output

Your full-stack application must provide:

1. Data Processing
✅ Ingest 24K+ papers metadata
✅ Extract abstracts and content
✅ Generate chunks and embeddings
✅ Build search index
✅ Extract citations
✅ Processing: "Indexed 24,000 papers"

2. Search Interface
Homepage:

- "Search across 24,000+ research papers..."
- Corpus: 24,000 papers indexed

Complex Query 1:
"What are the main criticisms of BERT?"

LLM Response:
"From 12 papers discussing BERT, main criticisms:

1. Computational Cost (8 papers)

- Papers: [Smith 2020], [Zhang 2021]
- Quote: "BERT requires substantial resources..." (Paper #23, page 5)

2. Context Window Limitation (7 papers)
    - 512 tokens insufficient for long docs
    - Solutions: Longformer, BigBird
3. Pre-training Bias (5 papers)
    - Inherits web data biases
    - Gender/racial bias observed

Citations: 12 papers cited"

Complex Query 2:
"Find papers proposing alternatives to attention"

LLM Response:
"Found 15 papers proposing alternatives:

Categorized:

1. Linear Attention (7 papers)
    - Key: "Transformers are RNNs" (Katharopoulos 2020)
    - Approach: $O(n)$ vs $O(n^2)$ complexity
2. State Space Models (5 papers)
    - Key: "S4" (Gu 2022)
    - Better for long sequences
3. Hybrid (3 papers)

Comparison table generated..."

Complex Query 3:
"What datasets are commonly used?"

LLM Response:
"Top datasets across corpus:

1. GLUE - 23 papers (23%)
2. SQuAD - 16 papers (16%)
3. ImageNet - 12 papers (12%)

Trend: Moving from GLUE to SuperGLUE"

3. Citation Analysis (Bonus)

- Citation network for papers
- Most cited papers in corpus
- Influential paper detection

4. Performance Requirements

- Search response: <3 seconds
- Cross-document synthesis works
- Handle 24K+ papers
- Accurate citations

What You Must Build

Full-Stack Application:

- **Frontend:** Search UI, results display
- **Backend:** PDF processing, search, LLM synthesis
- **Database/Vector Store:** For chunks and metadata
- **LLM:** Cross-document analysis
- **PDF Processing:** Text extraction

Tech Stack: Your Choice

- Language: Python (recommended), Node.js, etc.
- Frontend: React, Vue, HTML, etc.
- Vector DB: FAISS, Pinecone, Weaviate, Chroma
- LLM: OpenAI, Anthropic, Google, local
- PDF: PyPDF2, pdfplumber, PyMuPDF
- Architecture: Your design

Grading

- 40% - Handles 24K+ papers, complex synthesis works
- 30% - LLM provides cross-document insights
- 20% - System design and performance
- 10% - Creative features (citation analysis, etc.)

Show working system. Results matter.