Bridging the Semantic Gap: Leveraging Latent Semantic Analysis for Improved Information Retrieval

Alen Abdrakhmanov, Christabel Hammond, Kasey Purvor, Rasul Abdullayev
School of Electronic Engineering & Computer Science, Queen Mary University of London
25 February 2025

Abstract

This paper presents the design of a scalable search engine addressing synonymy and polysemy challenges in scholarly literature retrieval. Our hybrid approach combines field-weighted Latent Semantic Indexing(LSI) with BERT-based neural language models in a two-stage architecture: LSI-based candidate selection with enhanced weightings, followed by BERT-based semantic re-ranking. The system will be initially implemented using 21,841 papers from the CORE dataset, with an architecture designed to scale to the full corpus of over 300 million scholarly resources. Our evaluation employs JudgeBlender, an automated relevance assessment method, on 30 diverse academic queries across three categories, comparing six system configurations, using nDCG metric to quantify both overall performance and component-specific contributions. Our work delivers both research contributions in semantic information retrieval and a practical, scalable implementation for academic search that bridges the gap between statistical term matching and contextual language understanding.

Contents

1.0 Introduction

The exponential growth of scholarly literature has transformed the landscape of academic research. With over 300 million searchable scholarly resources available globally (Larsen & von Ins, 2010), researchers face significant challenges in discovering relevant literature. Academic search differs fundamentally from general web search, as it requires deeper semantic understanding, disciplinary context awareness, and the ability to track conceptual relationships across diverse terminology. Despite advances in information retrieval (IR), current academic search tools often fail to fully address these specialized needs.

At the core of academic search challenges lie two fundamental linguistic challenges:

- Synonymy: Where varied terminology is used to describe the same concepts.
- Polysemy: Where the same terms can carry different meanings in different contexts.

These challenges are especially pronounced in academic search where terminology varies significantly over time and across disciplines.

In this paper, we propose a hybrid approach combining Latent Semantic Indexing (LSI), with enhanced weighting on key terms, with BERT-based neural language models to address these two key challenges. LSI captures statistical term relationships through dimension reduction techniques, while BERT provides contextual understanding of language. Our research explores how field-weighted LSI and strategic BERT integrations can improve both recall and precision in scholarly search. We present our literature review, research goals, deliverables, system architecture, dataset, and evaluation framework.

2.0 Review of Related work

Information Retrieval (IR) systems have evolved significantly since Mooers (1950) first outlined their theoretical foundations. Traditional keyword-based retrieval models often struggle to return relevant results that accurately reflect user intent, necessitating approaches that capture semantic meaning beyond basic term matching. As we analyse the literature, several gaps emerge in effectively addressing academic search challenges.

Early IR approaches began with Luhn's (1957) automatic text analysis, which used word frequency to identify important terms in documents. This approach lacked a weighting mechanism to distinguish between semantically significant and common terms. Salton and Buckley (1988) addressed this limitation by introducing term frequency-inverse document frequency (tf-idf), balancing term frequency within documents against their rarity across the corpus. This weighting scheme significantly improved retrieval performance by reducing the influence of high-frequency but uninformative words like "the" and "of" (Robertson & Jones, 1976).

Building upon these foundations, Robertson et al. (1995) introduced the Okapi BM25 ranking function at TREC-3, incorporating document length normalization and a saturation function to prevent over-emphasizing frequent terms. Unlike tf-idf's linear relationship between term frequency and relevance, BM25 implements diminishing returns:

$$RSV^{BM25} = \sum_{i \in q} \log\left(\frac{N}{df_i}\right) \cdot \frac{(k_1 + 1)tf_i}{k_{1\left((1-b)+b\cdot\frac{dl}{avdl}\right)} + tf_i}$$

Where:

- RSV^BM25: Retrieval Status Value (relevance score)
- q: Query terms
- N: Total number of documents in collection
- df_i: Document frequency of term i
- tf_i: Term frequency of term i in the document
- k_1: Term frequency saturation parameter (typically 1.2-2.0)
- b: Document length normalization parameter (typically 0.75)
- dl: Document length
- avdl: Average document length in collection

Though BM25 offers superior performance through parameter tuning, it shares a fundamental limitation with tf-idf: reliance on exact term matching, failing to address synonymy and polysemy.

To overcome these semantic challenges, Deerwester et al. (1990) proposed Latent Semantic Indexing (LSI), which uses Singular Value Decomposition (SVD) to uncover latent semantic structures in term-document relationships. In LSI, a term-document matrix (X) is decomposed into three matrices as follows:

$$X = T^0 S^0 D^{0\mathrm{T}}$$

Where T_0 contains orthogonal term vectors, S_0 is a diagonal matrix of singular values, and D_0 contains orthogonal document vectors. This can be visualized as:

[Terms × Documents] = [Terms × Factors] × [Factors × Factors] × [Documents × Factors]^T

By selecting only the k largest singular values and their corresponding vectors, LSI creates a reduced-dimensional representation that captures significant semantic patterns while filtering noise:

$$\hat{X} = TS^k D^{\mathrm{T}} \ (where\ k << min(terms, documents))$$

This dimensionality reduction transforms both terms and documents into the same semantic space, where proximity indicates conceptual similarity rather than exact lexical matching. Queries are treated as "pseudo-documents" and positioned within this space based on their component terms, allowing retrieval based on semantic similarity.

Deerwester's tests demonstrated LSI's effectiveness compared to traditional methods, particularly at higher recall levels. However, LSI has notable limitations: each term is represented by a single point that averages all possible meanings, only partially addressing polysemy.

Additionally, updating LSI indexes when new documents are added presents computational challenges, requiring either full re-computation or approximation methods — a gap that remains underexplored in the literature.

Document structure plays a significant role in retrieval effectiveness. Robertson et al. (2004) demonstrated that field-weighted approaches, which assign different weights to document sections (title, abstract, body), significantly improve retrieval performance. Koopman and Zuccon (2014) further validated that accounting for document structure better captures the varying semantic importance of terms based on their location.

Recent advances in neural language models have transformed IR systems. Nogueira and Cho (2019) pioneered BERT for passage re-ranking, showing significant improvements over traditional methods by leveraging contextual embeddings. MacAvaney et al. (2019) explored strategies for efficiently incorporating BERT into document ranking systems. For keyword extraction, Grootendorst (2020) introduced KeyBERT, leveraging BERT embeddings to identify semantically important terms in documents. While these advances are promising, there remains a substantial gap in literature regarding how these neural approaches can be effectively integrated with statistical models like LSI for academic search applications.

The practical implementation of advanced IR techniques often requires a multi-stage retrieval architecture. Reed et al. (2020) proposed efficient passage extraction methods to accelerate BERT-based re-ranking while maintaining effectiveness. Johnson et al. (2017) introduced FAISS for approximate nearest-neighbour search in high-dimensional spaces, enabling scalable retrieval from large document collections.

Evaluation methodologies have evolved significantly, with recent work addressing the high cost and subjectivity of manual relevance assessments. Rahmani et al. (2023) developed JudgeBlender, an approach that ensembles multiple automated signals to generate relevance judgments without requiring human assessors. This methodology combines textual entailment, semantic similarity, and other features to approximate human judgments, enabling more extensive and cost-effective evaluation of retrieval systems. However, the application of such automated evaluation techniques specifically to academic search systems remains limited in the literature.

Despite these advances, a significant gap remains in effectively combining the statistical strengths of LSI with the contextual understanding of neural models in a computationally efficient architecture for academic search. The literature shows limited exploration of field-weighted LSI, and even less attention to how such approaches can be integrated with BERT-based enhancement optimized for academic search. Additionally, while individual components have been studied separately, their complementary integration in a computationally efficient architecture specifically designed for scholarly literature remains underexplored.

Our work addresses these gaps by developing a hybrid approach that leverages field-weighted LSI for efficient candidate selection and BERT for semantic re-ranking, specifically tailored to the unique challenges of academic search. By exploring both component-level contributions and overall system architecture, we aim to bridge the divide between statistical techniques and neural language understanding in the context of scholarly literature retrieval.

3.0 Research Goals

Our research aims to address fundamental challenges in academic search by investigating how semantic techniques can improve retrieval effectiveness. Building on the identified gaps in the literature, we formulate the following research questions that guide our system design and evaluation.

1. Baseline Comparison: How effectively does Latent Semantic Indexing (LSI) address synonymy and polysemy challenges in academic search compared BM25 when using the same dataset and queries?
2. Field Weighting Impact: To what extent does applying different weights to document fields (title, abstract, body) in the LSI model improve retrieval performance for academic papers compared to standard non-weighted LSI?
3. BERT Document Representation: How does enhancing document indexing with BERT-extracted key concepts improve the system's ability to capture the semantic core of academic papers compared to using only statistical LSI approaches?
4. BERT Query Re-ranking: What performance improvements can be achieved by implementing a two-stage retrieval process; where initial LSI results are re-ranked using BERT-based similarity measures between query and candidate documents?
5. Architecture Efficiency: What is the optimal architecture for combining LSI and BERT components in terms of computational efficiency, query response time, and index maintenance, particularly when scaling to large academic paper collections?

4.0 Deliverables

Our project will deliver a fully functional academic search engine prototype with the following components:

1. **Core Search Functionality**: A complete implementation of our field-weighted LSI and BERT-enhanced retrieval pipeline.
2. **User Interface**: An intuitive search interface designed for academic researchers with filtering capabilities and results visualization.
3. **Indexed Dataset**: Initial implementation using a representative subset of the CORE dataset (approximately 1 million papers).

4. **Scalability Design**: An architecture designed to efficiently scale to larger document collections as needed.
5. **Evaluation Tools**: Built-in metric and comparison framework to measure system performance against baseline methods.

5.0 Dataset

For this project, we have used a subset of the CORE sample dataset consisting of 21, 841 academic papers, balanced across disciplines, time periods, and sources. This subset provides sufficient diversity to test our LSI implementation while keeping computational requirements manageable during development.

Our end goal is to scale to the full CORE dataset, which contains over 304 million searchable scholarly resources and approximately 237 million free-to-read full-text papers from more than 11,000 data providers across 150+ countries.

This phased approach allows us to develop and refine our LSI implementation with a realistic dataset before scaling to the comprehensive academic paper collection.

Of the fields present we will make use of the title, abstract and body. While preserving the metadata for search displaying search results.

6.0 Search Components

6.1 Field-Weighted Latent Semantic Indexing (LSI)

Our primary approach for improving LSI's synonymy and polysemy handling is through field weightings. Instinctively terms in the title and abstract are more relevant than words in the main body. We plan to:

- Create a term-document matrix of all words in the Title, Abstract and Body.
- Apply TF-IDF weighting to improve relevance ranking.
- Apply additional weighting:
  - Terms in titles receive the highest weight (e.g., 3.0x)
  - Terms in abstracts receive medium weight (e.g., 1.5x)
  - Terms in body text receive standard weight (1.0x)

Conventional (SVD) then decomposes this matrix. Reducing dimensionality to 150 -300 dimensions, as discussed by Bradford (2008), to capture significant semantic patterns while filtering noise. Positioning both documents and queries within the same semantic space and ranking based on conceptual similarity rather than exact term matching.

6.2 BERT-Enhanced Semantic Processing

6.2.0 Dual Application of BERT Technologies

We will leverage different adaptations of BERT in two distinct ways, each serving a different purpose. These two applications of BERT are complementary but distinct. The first enhances how documents are represented in the index; while the second, optional stage, is computed at during the query processing and refines how results are ranked.

6.2.1. keyBERT for Enhanced Indexing (applied during document indexing):

KeyBERT is a minimal keyword extraction technique that leverages BERT embeddings to identify keywords that are semantically most like a document. It extracts candidate keywords and ranks them based on their semantic similarity to the document, identifying terms that best represent the document's semantic content. The use is as follows:

- Using keyBERT to identify semantically significant keywords and phrases in each document
- Applying additional weight (e.g., 2x) to these key terms in the LSI matrix

This enhances the underlying semantic space before any queries are processed and improves the foundation of our search system by creating more meaningful document representations

6.2.2. BERT for Semantic Re-Ranking (applied during query processing optionally):

Integrating BERT-based re-ranking into our retrieval pipeline enables context-aware ranking of documents, refining the results obtained from Latent Semantic Indexing (LSI). Unlike traditional term-based ranking methods, BERT leverages deep contextual embeddings, capturing complex semantic relationships between queries and documents.

FAISS is a library developed by Facebook Research for similarity search and clustering of dense vectors. It allows for quick nearest neighbour search in high-dimensional spaces, making it ideal for retrieving semantically similar documents from large collections with minimal computational overhead.

Approximate Nearest Neighbor (ANN) search is an efficient technique that approximately finds vectors that are closest to a query vector, trading accuracy for significant speed improvements. This approach is essential for scaling semantic search to large document collections where exact nearest neighbour computation would be prohibitively expensive.

Sentence-BERT (sBERT) (Reimers et al. 2019) is a modification of the BERT architecture optimized for generating semantically meaningful sentence embeddings. It derives fixed-size vector representations of text that capture semantic meaning, enabling efficient

similarity comparisons between queries and documents.

The re-ranking process consists of three key steps:

1. Reduce the computational cost of BERT by filtering the most relevant results using FAISS:
    - Convert all documents into dense vector embeddings using sBERT.
    - Instead of applying sBERT to all search results, which is computationally expensive, FAISS pre-filters the search space, allowing BERT to focus only on the most relevant documents according to the LSI search.
    - These embeddings are stored in FAISS, which then performs ANN search to retrieve the 50 most relevant documents.
2. Abstract-Based BERT Classification:
    - A fine-tuned BERT classifier will classify the 50 documents retrieved via FAISS based on their abstracts.
    - The documents will receive a relevance score via logits → sigmoid activation → probability score (0-1).
    - The higher the probability the more relevant the document is.
3. Final Re-Ranking:
    - Of the 50 results from FAISS now sorted by relevance assigned by the BERT classifier. The top 10 are selected
    - These make up the results that are shown to the user – should they select this optional feature.

This feature will increase query run time so is implemented as an optional extra for the user to select.

6.3.0 Training BERT for Query-Based Semantic Re-Ranking

For effective BERT-based ranking, the model must be fine-tuned on a dataset containing query-document pairs. Unlike off-the-shelf BERT embeddings, a fine-tuned model learns IR-specific ranking behaviour, aligning document representations to real-world user queries.

6.3.1 Data Preparation for Training

- Dataset: Training requires a query-document relevance dataset, such as:
    - MS MARCO (Microsoft Passage Ranking Dataset) → Large-scale web search data.
    - TREC-CAR (Complex Answer Retrieval) → Wikipedia-based structured passages.
    - S2ORC (Semantic Scholar Open Research Corpus) → Academic paper abstracts and citations.
- Training Format:
    - Query → *"What are convolutional neural networks?"*
    - Relevant Abstract → *"CNNs are a class of deep neural networks designed for processing structured grid-like data, such as images."* (Label = 1)
    - Irrelevant Abstract → *"The agricultural sector has seen major advancements with irrigation systems."* (Label = 0)
- Loss Function: Uses Binary Cross-Entropy Loss (BCE) to classify relevant vs. non-relevant abstracts.

A variety of these datasets will be trailed and evaluated.

6.3.2 Dataset Split

Split	Purpose	% of Data
Training Set	Used for model learning	80%
Validation Set	Used to fine-tune hyperparameters	10%
Evaluation (Test) Set	Used for final model evaluation on unseen data from our corpus	10%

7.0 Performance Metric and Evaluation Method

7.1 Primary Evaluation Metric

nDCG@10 is a particularly valuable metric for search evaluation because it effectively captures the graded relevance judgments provided by systems like JudgeBlender (section 7.3), distinguishing between results with varying degrees of usefulness rather than using binary assessments. By focusing specifically on the top 10 results, it aligns with actual user behavior, as most users rarely examine results beyond the first page, making it more representative of real-world search experiences. Additionally, as the established industry standard for evaluating search quality throughout research literature and commercial applications, nDCG@10 enables meaningful comparisons across different search algorithms and systems

$$DCG@10 = \sum_{i=1}^{10} \left(\frac{rel_i}{\log_2(i+1)} \right)$$

Where:

- **rel_i**: Relevance score of the item at position i in the ranked list
- **i**: Position in the ranked list (1 to 10)

7.2 Test Collection / Query Design

Create 30 diverse academic queries across three categories:

- **Terminology-focused queries** (10): Testing synonymy handling (e.g., "machine learning vs statistical learning methods")
- **Concept-focused queries** (10): Testing semantic understanding (e.g., "impact of dimensionality reduction on information retrieval")
- **Interdisciplinary queries** (10): Testing cross-domain retrieval (e.g., "applications of neural networks in bioinformatics")

7.3 Automated Relevance Judgment Methodology

7.3.1 JudgeBlender Implementation

Implement the JudgeBlender methodology (Rahmani et al. 2023) Eliminating the need for domain experts and manual judgements.

- Apply this methodology to assign graded relevance scores (0-3 scale) to query-document pairs:
    - 0: Not relevant
    - 1: Marginally relevant
    - 2: Relevant
    - 3: Highly relevant

These values will be directly used in the nDCG@10 metric

7.4 Document Pooling Process

1. For each of the 30 queries, run all system variants (outlined in section 7.5)
2. Collect top 50 results from each system
3. Remove duplicates to create a pool of unique documents for each query
4. Apply JudgeBlender to assess relevance for all query-document pairs
5. Store judgments in a structured format for evaluation

7.5 System Variants for Comparison

Compare these system configurations:

1. **Baseline 1**: BM25 implementation
2. **System Variant 1**: Basic LSI (k=150 dimensions, uniform weighting)
3. **System Variant 2**: Field-Weighted LSI (title=3.0, abstract=1.5, body=1.0)
4. **System Variant 3**: Field-Weighted LSI with BERT-enhanced indexing
5. **System Variant 4**: Complete system with BERT re-ranking

7.6 Analysis Plan

7.6.1 Overall Performance Comparison

- Compute primary metric (nDCG@10 for each system variant) for each query.
- Calculate bootstrap confidence intervals:
    - Generate 1000 bootstrap samples by resampling queries with replacement
    - Calculate nDCG@10 metric for each bootstrap sample
    - Derive 95% confidence intervals from the 2.5th and 97.5th percentiles
- Conduct significance testing between system pairs

7.6.2 Component Impact Analysis

- Isolate and quantify the impact of each component:
    - Field weighting impact: Compare Variant 1 vs 2
    - BERT indexing impact: Compare Variant 2 vs 3
    - BERT re-ranking impact: Compare Variant 3 vs 4

7.6.3 Query-Type Analysis

- Calculate per-category performance by comparing the 3 query types.
- Identify which components benefit particular query types

Analyse where synonymy and polysemy handling shows the greatest improvement

8.0 Team Work Split

The team members are going to implement the following features

Alen Abdrakhmanov – BERT reranking

Implement BERT-based reranking component

- Train and fine-tune Sentence-BERT models on academic datasets
- Create FAISS vector database
- Develop abstract-based BERT classifier for relevance scoring
- Package reranking application for containerized deployment

Christabel Hammond – UI, Test Queries and Literature Review

- Completed comprehensive literature review on LSI and BERT for IR
- Design and create test collection with 30 diverse academic queries
- Design intuitive search interface
- Implement responsive web interface with search filters
- Create result visualization components
- Prepare documentation and user guides

Kasey Purvor – Indexing and tooling

- Implement indexing component including keyBERT keyword enhancement
- Develop LSI indexing pipeline with field weights
- Implement SVD dimensionality reduction
- Lead overall system architecture and component integration
- Manage tooling selection and technical infrastructure
- Coordinate containerization and deployment workflows

Rasul Abdullayev – Evaluation

- Implement JudgeBlender for automated relevance assessment
- Develop evaluation metrics calculation (nDCG@10, etc.)
- Conduct comparative analysis of system variants
- Create statistical significance testing framework
- Prepare visualizations and reports of evaluation results

9.0 Tooling

Development Environment

- Python 3.7+: Core programming language for backend components
- Git & GitHub: Version control and collaboration
- VSCode / Anaconda: Development environments
- Jupyter Notebooks: Prototyping and experimentation
- Docker: Basic containerization for consistent environments

Data Processing & Indexing

- NumPy/SciPy: Matrix operations and SVD implementation
- Gensim: Production-ready LSI implementation
- KeyBERT: Keyword extraction from documents
- NLTK: Text preprocessing and tokenization
- Pandas: Data manipulation and processing

Neural Components

- PyTorch: Framework for BERT model training
- Hugging Face Transformers: BERT models
- Sentence-Transformers: Sentence embedding models
- FAISS: Vector similarity search

Backend & Frontend

- Flask: Lightweight web framework for APIs
- SQLite: Local database for development
- React: Frontend UI framework
- TailwindCSS: CSS framework for responsive design
- Axios: API request handling

Evaluation Tools

- scikit-learn: Evaluation metrics calculation
- Matplotlib: Visualization of evaluation results
- pytest: Unit and integration testing

10.0 Time Allocation

Implementation Timeline (6 Weeks)
Week 1: Setup & Initial Development

- **All**: Project setup, dataset acquisition, and preprocessing
- **Christabel**: Develop UI wireframes and design interface mock-ups
- **Kasey**: Configure development environment and begin LSI implementation
- **Alen**: Start BERT model selection and training preparation
- **Rasul**: Create evaluation framework skeleton

Week 2-3: Core Components Development

- **Christabel**: Implement search interface and user preference controls
- **Kasey**: Implement basic and field-weighted LSI indexing
- **Alen**: Train BERT models and implement FAISS integration
- **Rasul**: Develop query collection and JudgeBlender implementation

Week 4: Integration & Enhancement

- **Kasey**: Integrate keyBERT with LSI indexing
- **Alen**: Complete reranking component development
- **Christabel**: Implement results visualization and filtering components
- **Rasul**: Finalize test queries and evaluation metrics

Week 5: System Integration & Testing

- **All**: Integrate all components into unified system
- **Kasey & Alen**: Connect indexing and reranking pipelines
- **Christabel & Rasul**: Link frontend with backend and prepare evaluation
- **All**: Begin system testing with all variants

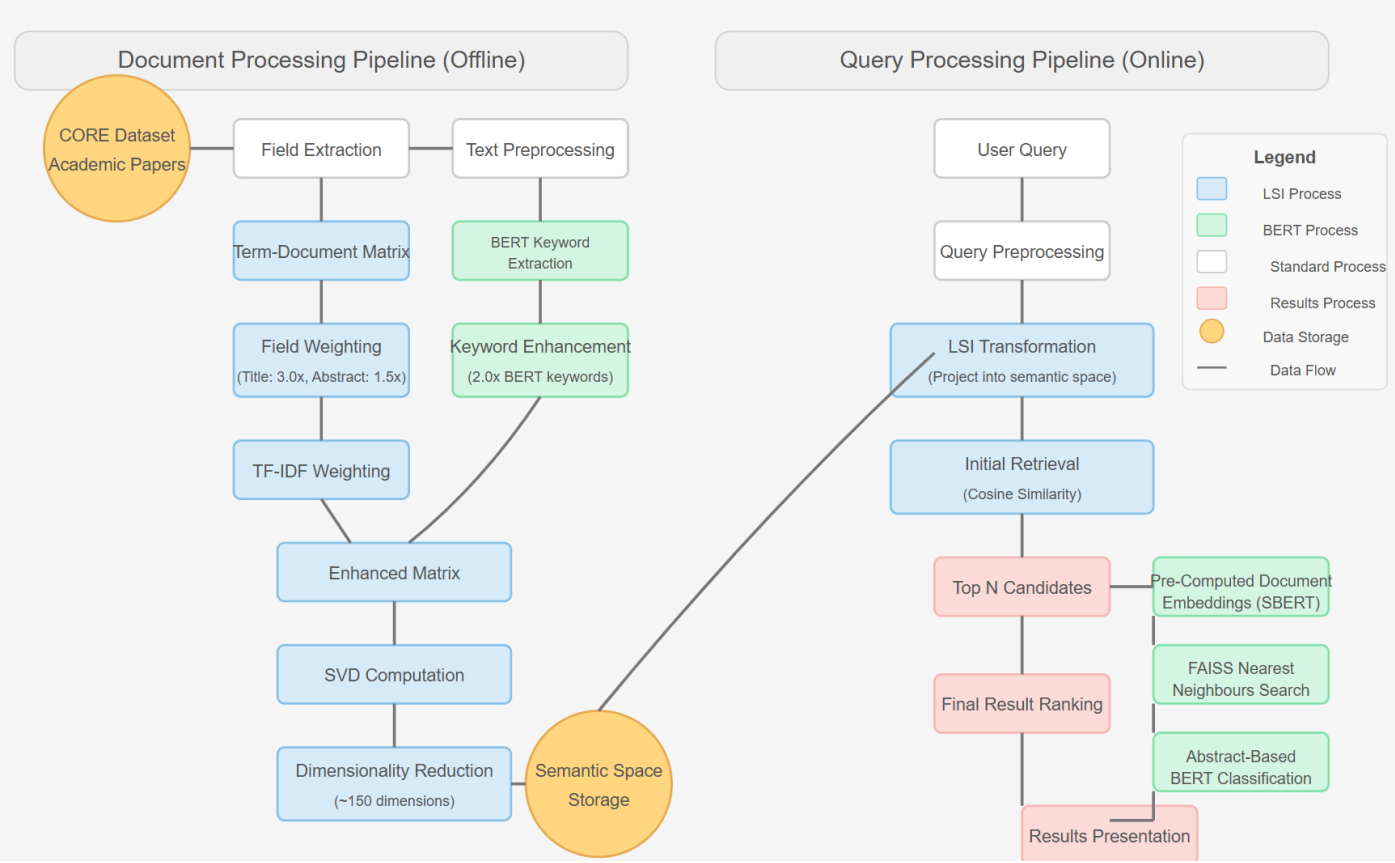
Week 6: Evaluation & Finalization

- **Rasul**: Complete evaluation across all system variants
  - All: Fix issues and optimize performance
  - Christabel: Finalize documentation and user guides
  - All: Prepare final report and presentation

11.0 Architecture

11.1 Two-Stage Processing Pipeline

- Stage 1: Initial retrieval via field-weighted LSI for candidate selection
- Stage 2 (optional to user): BERT-based semantic re-ranking of top candidates



12.0 References

- Bradford, R.B. (2008) 'An empirical study of required dimensionality for large-scale latent semantic indexing applications', in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. Napa Valley, California, USA: ACM, pp. 153-162.
- Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K. and Harshman, R. (1990) 'Indexing by latent semantic analysis', *Journal of the American Society for Information Science*, 41(6), pp. 391-407.
- Grootendorst, M. (2020) 'KeyBERT: Minimal keyword extraction with BERT', *Zenodo*. Available at: https://doi.org/10.5281/zenodo.4461265
- Johnson, J., Douze, M. and Jégou, H. (2017) 'Billion-scale similarity search with GPUs', *IEEE Transactions on Big Data*, 7(3), pp. 535-547.
- Koopman, B. and Zuccon, G. (2014) 'Why assessing relevance in medical IR is demanding', in *Proceedings of the Medical Information Retrieval Workshop at SIGIR*. Gold Coast, Australia, pp. 16-19.
- Larsen, P.O. and von Ins, M. (2010) 'The rate of growth in scientific publication and the decline in coverage provided by Science Citation Index', *Scientometrics*, 84(3), pp. 575-603.
- Luhn, H.P. (1957) 'A statistical approach to mechanized encoding and searching of literary information', *IBM Journal of Research and Development*, 1(4), pp. 309-317.
- MacAvaney, S., Yates, A., Cohan, A. and Goharian, N. (2019) 'CEDR: Contextualized embeddings for document ranking', in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Paris, France: ACM, pp. 1101-1104.
- MacAvaney, S., Yao, H.-R., Yang, E., Russell, K., Goharian, N. and Frieder, O. (2019) 'Efficient document re-ranking for transformers by precomputing term representations', *arXiv preprint arXiv:2004.14245*. Available at: https://arxiv.org/abs/2004.14245 (Accessed: 4 March 2025).
- Melucci, M. (2005) 'Context modeling and discovery using vector space bases', in *Proceedings of the 2005 ACM CIKM International Conference on Information and Knowledge Management*. Bremen, Germany: ACM, pp. 808-815.
- Mooers, C.N. (1950) 'Information retrieval viewed as temporal signalling', in *Proceedings of the International Congress of Mathematicians*, 1, pp. 572-573.
- Nogueira, R. and Cho, K. (2019) 'Passage re-ranking with BERT', *arXiv preprint arXiv:1901.04085*. Available at: https://arxiv.org/abs/1901.04085 (Accessed: 4 March 2025).
- Rahmani, R., Shao, X., Kucukozyilmaz, E., Suchindranath, A., Tan, S., Zhao, X. and Chen, J. (2023) 'JudgeBlender: Ensembling judgments for automatic relevance assessment', in *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Taipei, Taiwan: ACM, pp. 2392-2402.
- Reed, C., Madabushi, H.T. and Barker, E. (2020) 'Efficient passage retrieval with hashing for open-domain question answering', in *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, pp. 2583-2595.
- Reimers, N. and Gurevych, I. (2019) 'Sentence-BERT: Sentence embeddings using Siamese BERT-networks', in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Hong Kong, China: ACL, pp. 3982-3992.
- Robertson, S.E. (1977) 'The probability ranking principle in IR', *Journal of Documentation*, 33(4), pp. 294-304.
- Robertson, S.E. and Jones, K.S. (1976) 'Relevance weighting of search terms', *Journal of the American Society for Information Science*, 27(3), pp. 129-146.
- Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M. and Gatford, M. (1995) 'Okapi at TREC-3', in *Proceedings of the Third Text REtrieval Conference (TREC-3)*. Gaithersburg, Maryland: NIST, pp. 109-126.
- Robertson, S.E., Zaragoza, H. and Taylor, M. (2004) 'Simple BM25 extension to multiple weighted fields', in *Proceedings of the 13th ACM International Conference on Information and Knowledge Management*. Washington, D.C., USA: ACM, pp. 42-49.
- Salton, G. and Buckley, C. (1988) 'Term-weighting approaches in automatic text retrieval', *Information Processing & Management*, 24(5), pp. 513-523. *ACM Conference on Information and Knowledge Management*. Napa Valley, California, USA: ACM, pp. 153-162.