Scientific Article Search Engine

Authors:  
Spilios Spiliopoulos (ID: 4495)  
Konstantinos Chatzopoulos (ID: 1796)

# Abstract

This project presents the design and implementation of a search engine for scientific research papers, developed using Apache Lucene. The system allows users to perform advanced queries on a dataset of research papers, supporting features such as synonym-based query expansion, wildcard search, search history, and results highlighting.  
  
The dataset, derived from the NIPS conference papers (1987–2019), was preprocessed to create a smaller corpus of 500 entries. A Lucene index was built on this dataset to enable fast and efficient retrieval.  
  
The final system provides multiple modes of result presentation (console, text file, HTML) and offers advanced functionality for query refinement, making it a practical and extendable solution for research paper retrieval.

# Introduction

The primary goal of this project was to create a search engine capable of retrieving scientific articles based on user-defined queries. Unlike generic search tools, this engine supports query customization and relevance optimization through synonym expansion, wildcard search, sorting options, and search history management.

# Dataset

Source: Kaggle NIPS Papers 1987–2019.  
  
Preprocessing:  
- Random sample of 500 records  
- Removed the abstract column due to missing values  
- Cleaned newline inconsistencies in the full\_context field  
- Final dataset stored in papers\_cleaned.csv  
  
Fields used: source\_id, year, title, full\_context.

# System Architecture

The system follows a modular design. Each component is implemented as a dedicated class:  
- Main → Entry point; handles user interface and search execution  
- CSVReader → Loads and parses the dataset  
- SearchHistory → Manages past queries; provides suggestions  
- SearchResultsWriter → Stores results in .txt format  
- SearchResultsWriterHTML → Stores results in .html format with highlighting  
- LuceneSearch → Builds index, performs searches, applies ranking, sorting, and highlighting.

# Query Processing

1. User Input: User selects a field and provides a query.  
2. Search History Integration: Suggests similar past queries.  
3. Synonym Expansion: Suggests alternative queries based on a synonym map.  
4. Wildcard Search: Supports \* (multi-character) and ? (single-character).  
5. Sorting: Results can be sorted by year (ascending/descending).

# Search Functionality

LuceneSearch class handles indexing and searching:  
- Indexing: Builds a Lucene index at MetaData/, uses StandardAnalyzer, stores papers as Documents.  
- Searching: Executes queries, ranks results by relevance score, highlights matching terms, removes duplicates.  
- Pagination: Results shown in pages of 10, with user navigation.

# Results Presentation

Results presented in three formats:  
1. Console: 10 results per page, paginated.  
2. Text File (.txt): Plain text output with metadata.  
3. HTML File (.html): Results with highlighted query terms.  
  
Each search is also logged in SearchHistory with field, query, and number of results.

# Technologies Used

- Java  
- Apache Lucene  
- Python (for preprocessing script)

# Future Improvements

- Support infix wildcard placement  
- Integration of NLP libraries for synonym expansion  
- Bi-directional navigation through result pages  
- Web-based user interface

# Conclusion

This project demonstrates the construction of a scientific article search engine using Apache Lucene. By combining advanced features such as synonym expansion, wildcard queries, and history-based suggestions, the system provides an efficient way to explore research papers. The modular design allows for easy extension, making it suitable for further development into a fully-fledged research retrieval platform.