Search Engine for CS Papers

Miaomiao Zhang
Asmita Prabhakar

1 INTRODUCTION

Due to the fact that hundreds of research papers in computer science come out day by day, finding relevant and qualitative information becomes a problem. The project aimed at building a computer science paper search engine, including keyword and semantic searching[2] with an effortless UI interface. The backend search functionalities will be developed in Python, while Streamlit[1] is used for developing the front-end interface; hence, this particular system would be quite user-friendly.

2 OBJECTIVES

The chief goals of this project will be as follows:

1) Keyword Search: A user can search for the availability of papers containing particular keywords. 2) Semantic Search: Provide semantic search that could enable smarter context-aware queries, ensuring users find papers even when their queries are not exact keyword matches. 3) User Interface: Construct a clean and simple UI so working with the application would be as intuitive as possible using Streamlit.

3 IMPLEMENTATION

3.1 Dataset

We use "citeseer2"[2] dataset for this search engine project. It consists 9999 scientific publications and each document is text file in a single folder.

3.2 User Interface

Development of the user interface is based on Streamlit, a powerful library to create interactive web applications in pure Python. This is the main reason that we implement search engine in Python instead of using lucene[3] in Java. The key UI features are as follows:

Search Bar: A basic input field where users can insert certain keywords or phrases. -Results Display: Dynamic display area, which would show results of search, including paper titles, authors, abstracts, links to full papers. Filtering: The option to filter results by keywords and also limit how many results the user needs.

We run the code in Windsor University server, delta.cs, which is a GPU machine, enables to search on a large dataset efficiently. Once we run the code, it will return a fixed URL. Here, in our project, it returns "http://10.60.8.51:8501/". Users under the same network can access to a web page designed by streamlit and do some search work.

3.3 Backend Development

We use these libraries to implement the backend of the search engine. WHOOSH: It is a library used for indexing and searching. The keyword will allow the documents to be searched and retrieved with ease[1]. It is similar to lucene in Java, which is also used for index search. PANDES: It is helpful in manipulating and analyzing data and, hence, makes the metadata and contents of papers easy to handle. Sentence-Transformer[4]: It is a pre-trained model to encode documents for semantic search. To be more specific, we use “all-MiniLM-L6-v2” model. The model outputs a single embedding that represents the entire document[3]. This embedding captures the context and meaning of the document, not just individual words. It would convert each document into a vector with 384 dimensions.

Search Steps: -Data Collection: Scrape data from arXiv, IEEE Xplore, and Google Scholar into a database by storing the titles, authors, abstracts, and publication dates in a formatted way. -Indexing: Index the papers with Whoosh so it can enable quicker searching. -Search Functionality: The whole process enables the user to query the search functionalities. It searches for relevant papers on certain keywords. Besides, it will rank those results by certain specified criteria.

4 CONCLUSION

This would make it easier to find relevant academic resources by providing a generic search engine for computer science papers. It’s supported with strong backend search capabilities, mixed with a really user-friendly Streamlit interface, to offer a very effective tool.

REFERENCES

[1] Johannes Huck. 2020. Development of a Search Engine to Answer Comparative Queries.. In *CLEF (Working Notes)*.

[2] Eetu Mäkelä. 2005. Survey of semantic search research. In *Proceedings of the seminar on knowledge management on the semantic web*. Department of Computer Science, University of Helsinki, Helsinki.

[3] Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021. Hi-transformer: Hierarchical interactive transformer for efficient and effective long document modeling. *arXiv preprint arXiv:2106.01040* (2021).

[1] https://streamlit.io/

[2] https://linqs.org/datasets/

[3] https://lucene.apache.org/

[4] https://www.sbert.net/sentence_transformer.html