UNIVERSITY OF WEST ATTICA
DEPARTMENT OF COMPUTER ENGINEERING AND INFORMATION TECHNOLOGY

Information Retrieval Task

Creating a search engine for academic papers

The work concerns practical practice and experience in creating a simple search engine with the aim of understanding the fundamental concepts of information retrieval, indexing, ranking and information search, as well as practical skills in natural language processing and the application of search algorithms.

Objectives of the work:

- The design and implementation of a document retrieval system that can index and search efficiently in a collection of text documents.
- Developing retrieval algorithms and evaluating their performance using common evaluation metrics such as precision, recall, and F1 score. • Providing a user-friendly interface for users to enter queries. and retrieve relevant documents.
- Gaining practical experience in various information retrieval techniques, such as Boolean retrieval, Vector Space Model and Probabilistic retrieval models.

Job description

Implement a search engine that retrieves academic papers based on user queries. The search engine will have two main components: a. a web crawler to collect academic papers and

b. a search interface for users to search for and retrieve documents.

Step 1. Web Crawler:

a. Select a target website or scholarly repository (e.g. arXiv, PubMed, or a university repository).
b. Implement a web crawler in Python (e.g. with BeautifulSoup) to collect metadata of academic papers (title, authors, abstract, publication date, etc.) from the selected source.
c. Store the collected data in a structured format, such as JSON or CSV.

Step 2. Text Processing:

Preprocess the text content of academic papers to prepare them for indexing and searching. This may include tasks such as tokenization, stemming/lemmatization, and stop-word removal and removing special characters.

Step 3. Indexing:

a. Create an inverted index data structure to efficiently match terms to the documents in which they appear.

b. Implement a data structure to store the index.

Step 4. Search Engine:

a. Develop a user interface for searching academic papers using Python (e.g. a command line interface or a simple web interface).

b. Implement multiple (at least 3) retrieval algorithms, such as Boolean retrieval, Vector Space Model (VSM) and Probabilistic retrieval models (e.g. Okapi BM25) to retrieve relevant tasks based on user queries. The user will be able to choose the retrieval algorithm.

c. Allow users to filter search results by various criteria, such as publication date or author.

Query Processing: Develop a query processing module that will preprocess the queries received from the user, parse them and retrieve relevant documents using the inverted index. You can use simple queries based on words (terms). Users should be able to search for documents using one or more words. The module will receive user queries which will be tokenized and will perform Boolean operations (AND, OR

and NOT).

Ranking results: Apply a basic ranking algorithm. You can start with a simple TF-IDF (Term Frequency-Inverse Document
Frequency) and later you can include more advanced ranking techniques. Sort and present search results in a user-friendly format.,

Step 5. System evaluation:

In the next step (documentation), present in detail the methodology (data sets, metrics, python libraries, etc.) that you would follow to evaluate the
effectiveness of the search engine you implemented.

Step 6. Reporting and documentation:

Write a comprehensive **report and documentation** explaining the design, implementation, and evaluation of the search engine (no code explanation needed). Mention the difficulties you encountered and the suggested improvements. Include case studies of user queries with screenshots to demonstrate the functionality of the search engine.

Job evaluation criteria:
The evaluation of the work will be based on the quality and completeness of both the report and the search engine, the effectiveness of the ranking algorithm, the evaluation metrics, and your ability to explain the concepts and decisions you made.
during the work defending their design choices and performance in evaluating their recovery system.

Examination:
The exam for the laboratory part will be held together with the theoretical exam, in each examination. The grade from the assignment will only count if the questions corresponding to the laboratory part are answered.

The work can be undertaken by a team of up to 2 people.

Deliverables: *Report in .pdf* format *with file name* **Surname1 AM1-Surname2 AM2.pdf** *and code* with the search engine integrated in a **zip** file

Date of submission (by **only one member** of the team) to eclass: **22/01/2024**

no possibility of late submission

Questions will only be answered during the laboratory session and not by email