Search Engine Design Report – Assignment 2 (Group 13)

Keyword-Based Retrieval Search Engine

1. Aim, Task and Dataset:

Aim: The aim of this project is to develop a sophisticated search engine optimized for academic research purposes. This engine will systematically index '**Wikipedia articles**', within the dataset facilitating precise and efficient information retrieval. This approach is intended to significantly enrich text analysis and retrieval capabilities, thereby enhancing the search engine's ability to efficiently sift through a large corpus of textual content. (Zubiaga, 2023)

Task: The task at hand involves creating a streamlined indexing framework within a JSON dataset named '**wiki-articles.json**'. This framework is tailored to facilitate efficient keyword-based searches, cataloging the occurrences of each word within the articles. This approach ensures that the search engine effectively identifies articles relevant to the keywords entered by users. (Zubiaga, 2023)

Dataset: The dataset comprises Wikipedia articles, each represented by a structured entry that includes an article's ID, URL, title, and body text, with non-textual content removed (Zubiaga, 2023). This selection of documents capitalizes on the diversity and depth of information present in Wikipedia articles, providing a robust platform for testing the search engine across a variety of topics and query requirements.

Foundation: At its core, the search engine integrates the **Vector Space Model (VSM), Cosine Similarity**, and the **K-means Algorithm** to measure cluster relevance to user query.

Evaluation Methodology: The effectiveness of the Search Engine will be quantitatively and qualitatively evaluated through several metrics (Zhang, 2024):

- **Indexing Accuracy Metrics:**
  - **Keyword Relevance Precision:** Assessing the precision of our indexing methodology by evaluating how effectively the search engine identifies and associates relevant keywords with the appropriate Wikipedia articles. This metric focuses on the accuracy of keyword-to-article mapping, ensuring that search results are closely aligned with user queries.
  - **Keyword Identification Recall:** Evaluating the recall of the indexing system to ensure that all pertinent keywords are successfully identified and cataloged within the search engine's index structure.
- **Retrieval Performance Metrics:**
  - **Query Response Efficiency:** Measuring the efficiency and scalability of our search engine by evaluating the response time for user queries, ensuring swift and responsive information retrieval.
- **Ranking Effectiveness Metrics:**
  - **Mean Average Precision (MAP):** Examining the search engine's ranking effectiveness by considering precision at different recall levels, providing a

nuanced understanding of how well the system prioritizes and ranks relevant articles.
    - **Discounted Cumulative Gain (DCG):** Assessing the engine's ability to rank relevant articles higher, offering insights into the ranking capabilities, concerning the graded relevance of retrieved articles.
- **K-Means Algorithm:**
    - **Elbow Method**: Determining the optimal number of clusters for the dataset.
    - **Silhouette Score**: Assessing the quality of clustering.

2. General Architecture of the Search Engine:

The architecture of the search engine is specifically focused on keyword-based retrieval from a dataset of Wikipedia articles. It integrates sophisticated text representation, relevance scoring, and clustering techniques to ensure precise and efficient information retrieval. Here's an overview of the components:

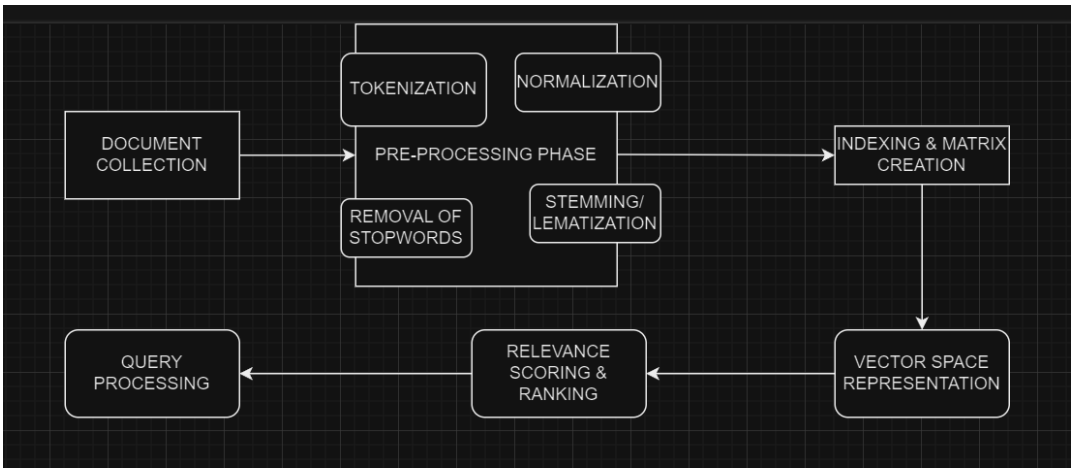

Figure 1: Keyword-Based Search Engine Architecture incorporating Vector Space Model

Document Collection

The engine's document collection comprises a curated dataset of Wikipedia articles ('wiki-articles.json'), each containing an article ID, URL, title, and body text. This comprehensive collection serves as the primary resource for indexing and retrieval, offering a broad spectrum of information across various domains.

Preprocessing Phase

Tokenization: Text documents undergo a series of preprocessing steps before indexing. This phase includes tokenization, where text is broken down into individual terms or words, preparing it for further processing. This step is fundamental for identifying the key elements within the text, facilitating the search engine's ability to perform efficient keyword-based searches. The focus is on ensuring that each word is accurately identified and made searchable, enhancing the engine's capacity to match user queries with relevant documents based on keyword relevance. (Cambridge University Press, 2008)

Normalization, Stop Word Removal, and Stemming: These steps are optimized for articles content, ensuring the removal of irrelevant terms and the standardization of vocabulary to facilitate accurate matching between queries and documents. (Otten, 2023)

Indexing and Relevance Matrix Creation

A specialized indexing structure will be utilized, moving beyond a basic term-document matrix. This framework catalogs each word's occurrences within the articles, optimizing the engine's capability to retrieve information based on keyword relevance. By focusing on the presence and frequency of keywords, this structure enhances the search engine's ability to identify and prioritize documents that best match the user's query, ensuring efficient and accurate information retrieval without the need for tracking the specific location of terms within texts. (Zhang, 2024)

Vector Space Representation

In the VSM, both documents and queries are mapped as vectors within a multidimensional space, where each dimension corresponds to a unique term from the corpus. (Tam, 2021) The weight assigned to each term in a document's vector can vary, typically influenced by metrics like term frequency (TF) and inverse document frequency (IDF), balancing term importance based on occurrence and uniqueness. (Zhang, 2024)

K-Means Clustering Algorithm, Relevance, and Query Processing

A K-Means clustering component would be integrated into our search engine design, specifically for text analysis. Inspired by the works of various academics (Huang, 2008; Ravindran, R.M. and Thanamani, A.S., 2015), the clustering algorithm would be modified to be reflective of cosine similarity. As opposed to traditional methods of minimizing the Euclidean distances, our model minimizes angular distances between vectors. The algorithm would indirectly assess the cosine similarity of documents in forming topic clusters. Upon receiving a query, the search engine would identify the most relevant topic cluster. This would be achieved through a comparison of the query vector and cluster centroids, here utilizing true cosine similarity to gauge thematic alignment. By confining the search to a subset of the entire corpus (those documents within the pertinent cluster), the engine would lead to a drastic reduction in the computational overhead required for query processing. This approach tends to enhance the search experience by providing both textual and contextual relevance, as well as dynamic scalability. The decision to employ a modified K-Means algorithm for the search engine design, as opposed to hierarchical clustering algorithms, stems from its superior effectiveness in managing large document collections with reduced computational demands. (Huang, 2008, p.52).

Algorithmic Details

The algorithmic backbone of the VSM involves computing TF-IDF weights for terms in documents and cosine similarity scores between query and document vectors. This computation relies on principles of linear algebra and vector mathematics, illustrating the model's reliance on mathematical underpinnings for operational efficacy. (Fang and Wang, 2012)

The architecture presents a comprehensive and robust framework for the development of a keyword-based retrieval search engine, emphasizing the importance of TF-IDF weighting in conjunction with indexing to achieve superior search performance.

3. Retrieval Model:

As already mentioned in the previous sections, for this project, the Vector Space Model (VSM) would be employed as the cornerstone retrieval model, incorporating the cosine similarity metric and the K-Means algorithm for article clustering. Upon receiving a query, the model retrieves the articles with the highest similarity score within the most relevant topic cluster. The VSM model is favored over other retrieval models due to its simplicity, flexibility & intuitive nature. (Linkedin, 2024) This model's integration aligns perfectly with our goal of leveraging a dataset composed of Wikipedia articles, ensuring the retrieval process is both thorough and relevant.

3.1. Methodology:

The process of information retrieval would begin with the Loading and Preprocessing stage, where text from compressed files is extracted, cleaned (punctuation removal & lowercasing), and broken down (via tokenizing or lemmatizing) to prepare the data for analysis. After this, the preprocessed text would be vectorized using TF-IDF, enabling the model to quantify the significance of words within the dataset. The formula for TF-IDF is as follows:

$$tf - idf_{t,d} = tf_{t,d} * idf_t$$
$$idf_t = \log_{10}\left(\frac{N}{df_t}\right)$$

Once the text is vectorized, the TF-IDF vectors undergo normalization in order to set their magnitude to 1. This ensures that the distance calculations during the subsequent clustering phase are more reflective of the angle between vectors. The K-means algorithm would be employed to cluster the normalized vectors minimizing the angular differences, forming groups of documents with related articles. An index would be created mapping articles to their respective clusters. This would facilitate the retrieval of articles belonging to a specific topic cluster. (Ravindran, R.M. and Thanamani, A.S., 2015)

After the clustering is complete, the user submits a search query, which is processed by the model to pinpoint the most pertinent cluster. Within this cluster, cosine similarities are compared to identify the top N articles most closely aligned with the query. (Huang, 2008) This approach minimizes the need for broader comparisons. The formula for cosine similarity is defined as:

$$\text{Cosine Similarity } (d, q) = \frac{d.q}{|d|.|q|}$$

Where d = Cluster Document Vector & q = Query Vector.

4. Tools for Building the Search Engine:

In terms of the tools required to build the search engine, the following programming language and its corresponding libraries will be used:

- **Programming Language**: Python, due to its extensive support for data manipulation and natural language processing (NLP).

Libraries (Burhan, 2019):

- **NLTK (Natural Language Toolkit)**: Essential for text processing tasks such as tokenization, stopword removal, and more NLP operations.
- **NumPy**: For efficient numerical operations, crucial for handling vector and matrix calculations in the Vector Space Model (VSM).
- **Scikit-learn**: Offers robust tools for TF-IDF vectorization, cosine similarity calculations, and implementing the K-Means algorithm.

5. Team Responsibilities:

Andrew Graham Porter

- **Dataset Acquisition, Preprocessing & VSM Creation**: Manage **wiki-articles.json** acquisition, conduct text preprocessing (tokenization, normalization, stop word removal, stemming), and initiate the VSM by creating a term-document matrix and calculating TF-IDF values.

Abhishek Vijaykumar Mane

- **VSM Implementation & Clustering**: Further develop the VSM through vector normalization and implement K-Means clustering, including optimizing the number of clusters and mapping documents to their respective clusters.

Mohammad Asghar

- **Search Functionality, VSM Utilization & Evaluation**: Process queries using the VSM, calculate cosine similarity, rank articles based on relevance, and evaluate the system's performance focusing on evaluation strategies like precision and recall.

Shreya Murigendra Pattanashetti

- **Integration, Testing, Deployment & VSM Optimization**: Lead the integration of VSM into the search engine, ensure robust testing, manage deployment, and optimize the VSM for performance and scalability.

Cross-functional Responsibilities: All members are involved in optimizing the system, scaling it for larger datasets, and refining based on user feedback, ensuring the VSM's effectiveness throughout the project.

6. Project Timeline and Milestones:

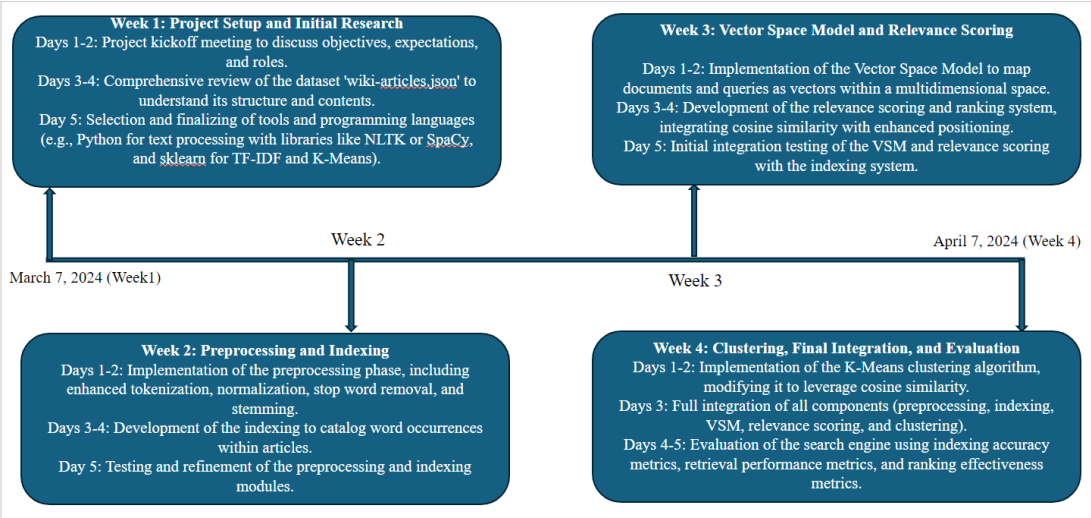

Figure 2: Project Timeline for the Keyword-Based Search Engine Design

References:

1. Burhan, M. (2019) Peermohtaram/vector-space-model: Vector space model: Implementation of vector space model (VSM) in Python, GitHub. Available at: https://github.com/peermohtaram/Vector-Space-Model?tab=readme-ov-file (Accessed: 06 March 2024).
2. Cambridge University Press (2008) Tokenization. Available at: https://nlp.stanford.edu/IR-book/html/htmledition/tokenization-1.html (Accessed: 07 March 2024).
3. Fang, K. and Wang, J., 2012. An Optimized Features Extraction Algorithm on VSM. In: 9th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2012). IEEE, pp.1471-1473. DOI: 978-1-4673-0024-7/10/$26.00
4. Huang, A., 2008, April. Similarity measures for text document clustering. In Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand (Vol. 4, pp. 9-56).
5. How do you choose the Best Information Retrieval Model for your domain? (no date) How to Choose the Best IR Model for Your Domain. Available at: https://www.linkedin.com/advice/0/how-do-you-choose-best-information-retrieval (Accessed: 06 March 2024).
6. Otten, N.V. (2023) How to use text normalization techniques in NLP with python [9 ways], Spot Intelligence. Available at: https://spotintelligence.com/2023/01/25/text-normalization-techniques-nlp/#3_Stop_Word_Removal (Accessed: 07 March 2024).
7. Ravindran, R.M. and Thanamani, A.S., 2015. K-means document clustering using vector space model. Bonfring International Journal of Data Mining, 5(2), p.10.
8. Tam, A. (2021) A gentle introduction to vector space models, MachineLearningMastery.com. Available at: https://machinelearningmastery.com/a-gentle-introduction-to-vector-space-models/ (Accessed: 07 March 2024).
9. Zhang, Q., 2024. Indexing and TF-IDF. [Lecture] Information Retrieval, Week 2, Queen Mary University of London
10. Zhang, Q., 2024. Evaluation. [Lecture] Information Retrieval, Week 5, Queen Mary University of London.
11. Zubiaga, A., 2023. Programming for Artificial Intelligence and Data Science Coursework 2 (ECS7023P). Unpublished coursework, Queen Mary University of London, Department of Electronic Engineering and Computer Science.