Names: Cara Failer, Siddharthen Sridhar
Class: CS 121 / INF 141
Instructor: Professor Lopes
Project: Assignment 3: M3

Query Report

Introduction

This document shall present a list of queries used to test our search engine. The queries were decided based on one or more of the following criteria:

- Guaranteed production of results
- Guarantee of no results
- Real world scenarios
- Complexity (more than two phrases)
- Simplicity (one or two phrases

Included with the queries are notes on how they responded to the retrieval process, along with any struggles encountered retrieving information.

Query List for Testing

1. clubs at UCI?
    a. Used the Jaccard coefficient and tf-idf to ensure that both “UCI” and “clubs” were present, in addition to emphasizing the header text since an ideal return case for this would be to receive a list with every single club at uci
2. contact helpdesk?
    a. Previous milestones may have had trouble with punctuation, so the regular expression function was altered to only consider alphanumeric characters
3. Security research
    a. 2-gram query which verified that jaccard coefficient was being used correctly and returning accurate results
4. Professors at UCI
    a. This is to ensure that the full word is tokenized and no close, fuzzy matched words were utilized when receiving any responses
5. works of Chen Li
    a. Similar to previous, where “works” is emphasized as opposed to just “chen li”. This was to see if results would differ from just the name.
6. Majors in ICS
    a. This is an example of where a stop word is useless since ICS Majors would have returned better results. Since stop words were not discarded, the jaccard index sorting ensured that the relevancy was accurate
7. UCI collabarations(spelled wrong on purpose)
    a. This is a test with a typo, which has not been integrated, and should instead simply return the same results for UCI.
8. Alfred Chen

    a. This professor is notable within the Computer Science department, and because of this, his name was added to the list of queries. This query encountered similar issues to the other names listed on the Query List. After some adjustments, the search engine produces results not only for the exact query, but for the singular words within the query. Because of this, adjustments had to be made to the ranking process (tf-idf and Jaccard) to ensure the exact query was the top result.
9. Clubs at UCI
    a. With the use of abbreviations and the vagueness of the query, it was a curiosity to see what results would be produced from this. Should this have worked like Google or Bing, the search results would have been a lot different.
10. Crista Lopes
    a. During testing, results for this query were split into words, but never the full phrase. After reworking the stemming, n-gram and tokenization process to include multi-word queries, it produced quick and positive results (the top results were “Crista Lopes”). After observing the results from the Analyst, the results were replicated with the Developer file collection.
11. Master of Software Engineering
    a. Due to the amount of terms within this query, results took a long time to load (about 1.4 seconds). Even with multi-threading, this query continued to take the longest to process.
12. ACM
    a. This query was chosen for its simplicity and relevance to the data. There were never any issues producing results for this query.
13. University of California, Irvine
    a. Had to deal with punctuation, which was fixed with regular expressions when tokenizing. Because of the length, however, this query took a long time to retrieve results.
14. Cybersecurity
    a. Simple one word query that was used to verify if tf-idf was working correctly
15. UCI OIT
    a. This is to see if anagrams would result in correct results, which may be a future feature to add in
16. Chen Li
    a. This query was chosen as it highlighted a notable member of the ICS department, meaning there would be a significant amount of results for this query. After adjusting the n-grams and stemming, accurate results were obtained.
17. To be or not to be
    a. This is a test query to gauge the outcome of queries that may not have exact results.
18. Machine Learning
    a. 2-gram query that was tested heavily, difficult to verify results without being able to see the pages but when the n-gram didn’t show up, the jaccard coefficient filtered it out well
19. How to draw a snowflake

    a. This is a query that shouldn't have much data in the database so it may rely heavily on less important words to attempt to provide relevant results
20. Google
    a. Works relatively well since it is just a one word query, very basic and doesn’t require n-grams, used tf-idf to use frequency to get better results
21. Data science
    a. 2-gram query that was used moderately near the end to test, lot of possible results so runtime is higher since it falls in the vertical search category, and used the jaccard coefficient to filter out if data and science didn’t go together. Can eventually use a kernel to seek for data x science where x is an arbitrary number of words