CSE 5337/7337: Information Retrieval and Web Search (Doing based on 7337 because we are a group)

Spring 2017, Project 2: Query engine implementation (100 points)

Due: Tuesday May 9, 6:00 pm

Demonstrations are ***encouraged***, but not required.

Preston Tighe & Arturo Laguna

Deliverables:

1. Complete code in a compressed archive (zip, tgz, etc)
2. A readme file with complete description of used software, installation, compilation and execution instructions to allow me to install and run your program if needed. *If a demonstration is not scheduled, then I must be able to install and run your program, so test your directions! Our demo can be found as demo.mov*
3. A document with the results for the tasks below.

Task:

**Develop a simplified query engine.**Test your data only on the data in:

http://lyle.smu.edu/~fmoore

1. Use the web crawler you built in Project 1 that crawled a limited space, looking for text and html files ***and php files***. You may need to modify how you saved the words from the pages that you traversed to support the query engine. Describe in detail what you changed to support the second half of the project. [10 points]

* Added a def calculate\_rankings: parses a text string and outputs files described in item 5, expanded query is run if less than N/2 documents are returned
* Thesaurus is manually added as a dictionary which has a list of alternatives words
* Added a def calculate\_cos\_sim: finds the cosine similarity of the search query compared to each document
* Altered the way we stored self.visited\_urls & self.docs for different functional requirements
* Added a def query\_engine: runs a CLI after the crawl is run so the user can run searches
* Change the ability to index PHP files just by added the extensions to indexable extensions
* We used numpy several times to do a couple of calculations

1. You will need a dictionary of words. [20 points]  
   a) What is your definition of “word”? Did it change from project1?

No it didn’t but our definition from project1 has changed. A word in our book is not part of an html tag, is separated by a space, doesn’t include punctuation, and isn’t a stop word.

b) How many words are in your dictionary?

1421  
c) What technique did you use to store your dictionary (fixed size, string array, one-large-string)? We used a 2d list for the doc term matrix, we used a list for doc urls, we used a dictionary for keeping track of words -> (url, tf) and words -> (total documents, total terms in all documents). Our doc term matrix is a fixed sized but ideally if we wanted to we could dynamically increase its dimensions quickly using numpy.

1. For the purpose of this project, you may assume a maximum of 50 documents. You will need to create a word/document frequency matrix to support item 5. [20 points]  
   a) remove documents if the content has already been seen. Marked as duplicate in program output  
   b) remove stop words from documents. What list did you use? A random stop list found online, the stop words are read in by file
2. The user will be able to enter multiple queries, consisting of one or more query words separated by space. The single word query “stop” will cause your program to stop.[10 points]   
   a) What happens if a user enters a word that is not in the dictionary? CLI output: <query> not found in domain  
   b) What happens if a user enters a stop word?

CLI output: Only disallowed words were searched upon. Please add more words to your query.  
c) A set of queries will be provided.

All queries and other random queries can be found in the program\_log.pdf or demo.mov

1. Implement the cosine similarity of the query against all documents. [40 points]   
   a) If any of the query words appear in the <title>, add 0.5 to the query score.

def has\_words\_in\_title checks and adds 0.5 if true  
b) Display the similarity measure, document URL, and document title in descending numerical order for the top 6 results.

Can be seen in the program\_log.pdf and demo.mov

**GRADUATE STUDENTS:**

1. Include in the display, the first 20 words of the document (this can be the stemmed version).

Can be seen in the program\_log.pdf and demo.mov

1. If less than N/2 documents are returned for a query, rerun the query using thesaurus expansion. A list of words, along with 1 – 3 synonyms will be provided.

Can be seen in the program\_log.pdf and demo.mov