CS 585 Homework 5
Search
Sean Karlage

1 Introduction

This code builds off the previous assignments by adding site scraping capability to the crawler and a searching capability to search text recovered from the scrape. The search uses a form of boolean term comparison explained below.

2 Augmenting the scrapy Crawler

The scrapy crawler presented in an earlier assignment (under the 'engrscrape' subdirectory) had to be augmented for text scraping capabilities. This required a simple change that stored the scraped text from the following HTML tags into a new field in the scrapy.Item passed throughout the scrapy pipeline:

- <title>
- <p>
- <h1>
- <h2>

The text from these tags were pasted together into one string, then compressed with Python's built-in zlib compression function. The compressed text was put into a new row in the database.

To my knowledge, this additional capability did not affect the crawler's performance that much. The crawler performed with an average of 3.84 requests/sec.

3 Building the Inverted Index

In order to construct the inverted index, each set of document terms had to be decompressed and parsed. The following transformations were made to each term to simplify them in some way:

- Removal of stopwords from the python NLTK stopwords corpus
- Removal of non alpha-numeric characters
- Stemming via python NLTK PorterStemmer

After these changes were made, a simple dictionary was created in memory, and then all words and the documents that contained them were inserted into a table in the database.

4 Searching Procedures

I attempted to get the vector space model working for my search procedure, but was unable to complete it in time. Instead, I chose the following approach:

1. Apply the same transformations to the search query as were done to terms in the inverted index
2. Compute the power set of all search query terms
3. For each query term, get the documents that contain that term
4. For each element of the power set (e)
    a. Compute the intersections of each set of documents associated with each term in (e)
    b. Store into list (documents)
5. For each group of documents in (documents)
    a. Rank documents by PageRank
6. Merge (documents) into one list
7. Report top N results

Effectively, I take the power set and compute the intersection of documents that contain different combinations of terms. Therefore, I primarily rank by those documents that contain all terms over those that only have some terms. As a second ranking metric, I rank each resulting list of documents by PageRank to get the most relevant pages in each group.

5 Selected Results

Note: only top 10 results are displayed (in order of ranking metric described above)
query = “jeffery ashley 280”
documents:

- https://www.engr.uky.edu/ece/faculty_staff/jeffreyashley
- https://www.engr.uky.edu/alumni/brick/ms-ashley-bielefeld
- https://www.engr.uky.edu/news/tag/jeffrey-ashley
- http://www.engr.uky.edu/~heath/EE280.html
- http://www.engr.uky.edu/alumni/brick/mr-jeffery-gilliam
- https://www.engr.uky.edu/news/2012/04/ashley-named-2012-lutes-award-recipient
- http://www.engr.uky.edu/alumni/brick/mrs-ashley-brinegar
- https://www.engr.uky.edu/news/2013/04/goldsmith-ashley-and-mihail-named-recipients-of-provosts-award-for-outstanding-teaching

query = “brent seales”
documents:

- http://www.cs.engr.uky.edu/people/faculty/bseales
- https://www.engr.uky.edu/research/researchers/william-seales
- http://www.engr.uky.edu/news/2005/01/seales-work-featured-in-odyssey
- https://www.engr.uky.edu/news/2014/01/bbc-news-magazine-publishes-article-involving-seales-research
- http://www.cs.engr.uky.edu/rss.xml
- https://www.engr.uky.edu/news/2015/01/seales-leading-major-breakthrough-reading-ancient-scrolls
- https://www.engr.uky.edu/news/2014/12/seales-taylor-win-great-teacher-awards
- http://www.cs.engr.uky.edu/people/faculty/bseales
- http://www.cs.engr.uky.edu/people/faculty/bseales
- http://www.cs.engr.uky.edu/node/505

query = “computer engineering major”
documents:

- http://www.engr.uky.edu/news/2009/07/electrical-and-computer-engineering-professor-receives-defense-agency-award
- https://www.engr.uky.edu/ece/faculty_staff/richard-anderson
- https://www.engr.uky.edu/ece/faculty_staff/wilda-moore
- https://www.engr.uky.edu/ece/faculty_staff/paul-a-dolloff
- https://www.engr.uky.edu/ece/faculty_staff/joseph-elias
- https://www.engr.uky.edu/ece/faculty_staff/linda-baldwin
- https://www.engr.uky.edu/ece/faculty_staff/reginahannemann
- https://www.engr.uky.edu/ece/faculty_staff/stephen-m-lipka
- http://www.engr.uky.edu/ece/faculty_staff/clayton-r-paul
- https://www.engr.uky.edu/ece/faculty_staff/robert-lodder

6 Evaluation

As can be seen above, for very targeted queries, the results are generally good. For both queries about Dr. Ashley and Dr. Seales, the web pages are generally relevant. Of note is that the query for Dr. Ashley returns other people named “Ashley”, which are incorrect results in this context.

For more general queries like the third about the computer engineering major, seemingly-random pages are returned. Overall, this searching method is not good for general queries, as terms can be included on various pages that might not be as relevant. A model such as the vector-space model would be much more appropriate, as it takes into account the term frequency and document frequency, which would serve move many of these noisy pages down in the rankings.

Overall, my search is decent for targeted searches for specific subjects, but lacks accuracy when it comes to more general queries. I believe the vector-space model would be much more appropriate.

7 Execution of code

In order to execute the code, the following commands must be run:

Crawling/Scraping

```
cd /path/to/extract/engrscrape
scrapy crawl engrspider -s JOBDIR=jobs/job-1
<... some time passes ...>
mv links.db ../db/
```

Ranking

```
cd /path/to/extract/rank
./pagerank.py ../db/links.db <alpha> <num results>
```

Building Inverted Index

```
cd /path/to/extract/search
./build_inverted_index ../db/links.db
```

Searching

```
./search ../db/links.db <query> <num results>
```