College of Computing and Information Science
Northeastern University
Spring 2019

CS6200
Information Retrieval
February 9, 2019

Homework Assignment #2
Due Date: Monday, February 18th, 2019 @ 11:59pm

Instructions:

- The assignment is due on the time and date specified.

- This is an individual assignment. You may discuss the problems with your friends, but the code, analysis, interpretation and write-up that you submit for evaluation should be entirely your own.

- You are encouraged to use the Piazza discussion board, and seek help from TAs and instructors to get clarifications on the problems posed.

- If you receive help from others you must write their names down on your submission and explain how they helped you.

- If you use external resources you must mention them explicitly. You may use third party libraries but you need to cite them, too.

Text Transformation and Link Analysis using PageRank

Task 1: Tokenizing and creating trigrams (20 points)

In this task, you will be generating a clean corpus from the list of URLs provided in the accompanying file **BFS.txt** and find the frequency of trigrams in the corpus, following the steps outlined below:

a) Download the articles (raw html, in text format) corresponding to each URL in **BFS.txt** and store them with their respective URL. This is essentially your task 1-g) from the previous assignment.

b) Parse and tokenize each article and generate a text file per article that contains only the title(s) and plain textual content of the article. Ignore/remove ALL markup notation (HTML tags), URLs, references to images, tables, formulas, and navigational components. Keep the anchor text while ignoring the associated URL.

c) Each text file will correspond to one Wikipedia article. The file name you will use should is the same as the article title. For example, the parsed version of the article obtained from http://en.wikipedia.org/wiki/Space_exploration will be named Space_exploration.txt. You should ensure that the file names are unique.

d) Your parser should provide options for case folding and punctuation handling. The default setup should perform both. The punctuation handler is expected to remove all punctuation preserving only hyphens from text and retain only the alphanumeric characters (and hyphens).
e) Create a data structure to hold all of the trigrams in the corpus as well as the corpus frequency for each trigram.
f) Generate a log-log plot of the Frequency vs. Rank order of the trigrams (similar to the plot in Figure 4.2 of our text by Croft, *et al.*)
g) Comment on whether the trigram frequency obeys Zipf's law. If so, estimate the constant corresponding to Probability of Occurrence * Rank order.

Task 2: **Constructing directed web graphs (20 points)**

a) Build a graph over the set of 1000 URLs provided in **BFS.txt**. Your graph should have a structure as shown below:

D1 D2 D3 D4 // Node D1 has incoming links from nodes D2, D3 and D4.
D2 D5 D6 // Node D2 has incoming links from nodes D5, and D6.
D3 D7 D8 // Node D3 has incoming links from nodes D7, and D8.
….

In your web graph, the nodes will correspond to the webpage *docID* which is the article title directly extracted from the URL. (For example, *Space_exploration* is the *docID* corresponding to the article extracted from http://en.wikipedia.org/wiki/Space_exploration.

Each line indicates the **in-link relationship**, which means that D1 will have in-coming links from all the URLs that link to D1. You only need to build the graph for the 1000 web pages given, and do not need to consider any other web pages.

We will name this graph **G1**.

b) Build a graph for the 1000 URLs from running the focused crawler for HW1-Task2. For this, use the URLs provided in the accompanying file, **FOCUSED.txt**. We will refer to this graph as **G2**.

Task 3: Link analysis: PageRank Implementation (30 points)

a) Implement the PageRank algorithm given in Figure 4.11, Page 110 of our text by Croft, *et al.*)
b) To test for convergence, calculate the L2-norm for the difference in PageRank values from successive iterations. You will need to compute the difference *R-I,* between vectors *I* and *R*, before assigning *R* to *I* in line 24.

The L2-norm of a vector *x* is given by:

$$||x||_2 = \sqrt{\left(\sum_i x_i^2\right)} = \sqrt{x_1^2 + x_2^2 + \ldots + x_i^2}$$

For the PageRank convergence check, the summation is over all of the web pages *p* in our graph. PageRank can be considered to have converged when the L2-norm of the difference vector *R* - *I* is less than 0.0005 for at least four consecutive iterations.

c) (Ungraded task) You can first test your PageRank algorithm on the following small graph:

```
A D E F
B A F
C A B D
D B C
E B C D F
F A B D
```

The final ranking should be: A>E> (F, C)>B>D, where F and C have the same PageRank value.

d) Finally, run your iterative version of PageRank algorithm on **G1** and **G2** respectively until their PageRank values "converge". Your results should be a list of the URLs sorted by PageRank value in descending order.

Task 4: Experiments with PageRank (30 points)

a) Perform the following runs for **both** graphs **G1** and **G2**.
As a **baseline** for comparison, use the resulting PageRank from Task 3-d) above, for **G1** and **G2,** respectively.
   i. Re-run the PageRank algorithm using $\lambda = 0.25$, $\lambda = 0.35$ and $\lambda = 0.5$. What do you observe in the resulting PageRank values relative to the baseline? Discuss the results.
   ii. Re-run the PageRank algorithm in Task3-d) for exactly 4 iterations. Discuss the results obtained with respect to the baseline.
   iii. Sort the documents based on their raw in-link count. Compare the top 25 documents in this sorted list to those obtained in Task 3-d) sorted by PageRank. Discuss the pros and cons of using the in-link count as an alternative to PageRank (address at least 2 pros and 2 cons).

What to hand in:

1) The source code for text transformation, corpus frequency, and plotting (Task 1)
2) The source code of your PageRank algorithm implementation. This should include code for Task 3-d) as well as all subtasks of Task 4 a).
3) A README.txt file for instructions on how to compile and run your code.

For Task 1:

4) A file listing the trigrams and frequency for the corpus.
5) A brief document with the log-log plot and your response to Task 1 g).

For Task 2:

6) The graph files you generated for G1 and G2 (as text files).
7) A brief report on simple statistics for G1 and G2 including:
    a. The number of pages with no in-links (sources)
    b. The number of pages with no out-links (sinks)
    c. Maximum in-degree
    d. Maximum Out-degree

For Task 3:

8) A file listing the L2-norm values, and sum of PageRank values until convergence for G1 and G2.
9) A sorted list of the pages in G1 and G2 by the steady-state values of PageRank. Report the Top 50 pages by their docID and score.

For Task 4:

10) A short report on your findings from the suggested experiments in 4 a) (all 3 subtasks, one paragraph each).