**Name: Uzi Smadja**

**Andrew ID: usmadja**

**Machine Learning for Text Mining**

**Homework 3 - Template**

1. **Statement of Assurance**

All material submited is original work and was done only by me. I have used the Wikipedia pages on PageRank and a few papers discussing PageRank (including StackOverflow) but all the code and implementation was done by me.

1. **Experiments**
2. Describe the custom weighing scheme that you have implemented. Explain your motivation for creating this weighting scheme.

I decided to try a weighing scheme that normalizes the rank and retrieval scores in relation to the max and min values.

Score =

I wanted to check and see if maybe normalizing/adjusting the rank and score according to the minimum and maximum scores of all the documents would maybe give a more accurate result and express the relations in our dataset more accurately.

1. Report of the performance of the 9 approaches.

I. Metric: MAP

|  |  |  |  |
| --- | --- | --- | --- |
| Method \ Weighting Scheme | NS | WS | CM |
| GPR | 0.0484 | 0.3326 | 0.3043 |
| QTSPR | 0.0525 | 0.3326 | 0.3038 |
| PTSPR | 0.0580 | 0.3326 | 0.3069 |

II. Metric: Precision at 11 standard recall levels

|  |  |  |  |
| --- | --- | --- | --- |
| Method \ Weighting Scheme | NS | WS | CM |
| GPR | 0.1588 0.1012 0.0810 0.0766 0.0709 0.0648 0.0570 0.0281 0.0078 0.0045 0.0037 | 0.8920 0.7061 0.5997 0.5280 0.4352 0.3233 0.2360 0.1001 0.0558 0.0383 0.0078 | 0.8140 0.6290 0.5420 0.4854 0.3972 0.3127 0.2254 0.0950 0.0538 0.0390 0.0083 |
| QTSPR | 0.1797 0.1083 0.0878 0.0802 0.0751 0.0685 0.0597 0.0306 0.0099 0.0059 0.0040 | 0.8920 0.7061 0.5996 0.5280 0.4350 0.3233 0.2360 0.1002 0.0559 0.0383 0.0078 | 0.8137 0.6173 0.5498 0.4824 0.3996 0.3092 0.2247 0.0975 0.0551 0.0410 0.0086 |
| PTSPR | 0.1980 0.1198 0.0978 0.0892 0.0820 0.0725 0.0594 0.0291 0.0108 0.0064 0.0040 | 0.8920 0.7061 0.5996 0.5280 0.4350 0.3233 0.2360 0.1002 0.0559 0.0383 0.0078 | 0.8200 0.6287 0.5444 0.4899 0.3999 0.3147 0.2266 0.0982 0.0570 0.0413 0.0087 |

III. Metric: Wall-clock running time in seconds

|  |  |  |  |
| --- | --- | --- | --- |
| Method \ Weighting Scheme | NS | WS | CM |
| GPR | 3343ms | 3411ms | 3318ms |
| QTSPR | 30204ms | 27407ms | 23936ms |
| PTSPR | 23077ms | 22531ms | 25492ms |

IV. Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| Method \ Weighting Scheme | NS | WS | CM |
| GPR | Dampening factor: 0.9 | Rank weight: 0.2 retrieval weight: 0.8 | Rank weight: 0.1 retrieval weight: 0.9 |
| QTSPR | Alpha:0.75 beta:0.15 Gamma:0.1 | Alpha:0.75 beta:0.15 Gamma:0.1 Rank weight:0.2 retrieval weight: 0.8 | Alpha:0.75 Beta:0.15 Gamma:0.1 rank weight:0.1 retrieval weight:0.9 |
| PTSPR | Alpha:0.75 beta:0.15 Gamma:0.1 | Alpha:0.75 beta:0.15 Gamma:0.1 Rank weight:0.2 retrieval weight: 0.8 | Alpha:0.75 Beta:0.15 Gamma:0.1 rank weight:0.1 retrieval weight:0.9 |

1. Compare these 9 approaches based on the various metrics described above.

I am getting significantly lower MAP results for NS, and better results in WS. The WS MAP results are similar across GPR/QTSPR/PTSPR, the reason can be one of two. Either the query personalization or topic personalization don’t have a significant effect or I have made an error in the code – which is always an option but I believe it is unlikely.

1. Analyze these various algorithms, parameters, and discuss your general observations about using PageRank algorithms.  
   The best MAP and Precision scores were received when the PageRank got significantly less weight than the relevance search. This means that the PageRank, even if personalized, has little influence over the relevance score which should get a much higher weight. It seems that the NS (only PageRank) gets the lowest MAP and Precision scores by far – meaning that the relevance score makes most of the decisions and accounts for most of the search.
2. 1. What could be some novel ways for search engines to estimate whether a query can benefit from personalization?  
   A novel way for search engine to estimate whether a query can benefit form personalization is when we believe the query has a personal nature, to find intent behind the query. If the intent is personal like locations, ambiguous terms, then the query could benefit a lot from personalization. Another case is in social networks, when the user is logged in and the search engine has additional information on the user, it might be useful to use the data. For example, search matches based on users I follow (twitter), friends (google+,facebook). Since most users are now logged in to a website most of the time, the user id and personal information is usually available to the search engine to use.

2. What could be some novel ways of identifying the user’s interests (e.g. the user’s topical interest distribution Pr(t|u)) in general?   
Signed users can provide insight on user's interests, cookies with user information that can be used to retrieve data. Also, the browsing history, click history, previous queries and even query sessions with timestmaps. Another useful way would be to try and analize dwell time of users on search results. Even though the information is lost the moment the user leaves the page and goes after one of the results, it is possible to analize if the user comes back after a certain period of time, and decide wheter or not the query was good. The time spent on each page and separation between good clicks and bad clicks can be hard because we need to differentiate between abundanment of information and satisfaction.

1. **Details of the software implementation**
2. Describe your design decisions and high-level software architecture;

I decided to use Java and created an abstract PageRank class with the common variables and functions for all three PageRank algorithm. There is a GlobalPageRank class that implements GPR and a Topic-Sensitive-PR class that works both for QTSPR and PTSPR (just receiving different input files for personalization).  
  
with the transition matrix and the transpose of the transition matrix(which all three PageRank algorithms need and use) and some basic functions like iterate and run.

1. Describe major data structures and any other data structures you used for speeding up the computation of PageRank;  
   The transition matrix is represented as a Map<Integer,Set<Integer>>, that way we don’t lose time and space saving information in a matrix and just the non-zero “cells” are kept in the transition matrix. Also, the transpose of the matrix is calculated and saved once in the initialization of the algorithm to save computation time. For the same reason, the document topics are saved in a Map<Integer,Set<Integer>>. Ranks for GPR are saved in a List<Double> and ranks for QTSPR/PTSPR are saved in a List<List<Double>> because we have information per topic.
2. Describe any programming tools or libraries that you used;

I used normal Java libraries like Map,Set,List and also used Map.Entry in order to be able to easily sort and compare the final ranks.

1. Describe strengths and weaknesses of your design, and any problems that your system encountered

A strength of my program is the modularity and the abstract class which makes it easy to change a certain part of the code without reprogramming the entire algorithm. There probably is a better/faster implementation but my code runs fairly fast due to the fact that the big calculation (like transposing the transition matrix) is done once and it is saved in a Map instead of a matrix, just like a sparse matrix representation.  
On the other hand, the fact that I used a Map instead of some kind of matrix turns every calculation or multiplication into multiple loops, causing the code to be less “readable” than would be to use libraries that implement matrix multiplication for example.

1. **Describe how to run your code (programming environment, command line, etc.)**

I used Eclipse to run my code in Java. In order to give program arguments, click on   
Run -> Run configurations -> GlobalPageRank/TopicSensitivePageRank -> Arguments -> Program arguments  
  
In order to launch GPR the input arguments line should be in the following format:  
<test\_directoru> <method> <output>  
example:   
C:\Smuzi\workspace\PageRank\indri-lists 2 tmp\_out2.txt

In order to launch TSPR the input arguments line should be in the following format:  
<test\_directory> <method> <output> <topic\_number> <distribution\_file>  
example:  
C:\Smuzi\workspace\PageRank\indri-lists 3 qtspr3.txt 12 query-topic-distro.txt

**Note:** method = 1 – runs NS weighting scheme  
 method = 2 – runs WS weighting scheme  
 method = 3 – runs CM weighting scheme

The program assumes that the user-query files for search relevance are given in a format of   
user-query.results.txt

The transition file must be named “transition.txt” and be in the root folder of the eclipse project.

The document topics file must be name “doc-topics.txt” and be in the root folder of the eclipse project