Name: Nebiyou Daniel Hailemariam

Andrew ID: nhailema

Machine Learning with Graphs

Homework 3

1. **Statement of Assurance.** I certify that all work submitted for this assignment is original and has been completed by me alone.
2.
    a. The custom weighing scheme I've implemented combines the page rank score with the relevance score from the search engine through the multiplication operator. My motivation for doing this is to ensure that a high score is only given to a document when both the page rank and the search engine agree on its significance.

    When I multiply the page rank score and the search relevance score, I'm essentially saying that I want to prioritize documents that not only match the search query well but also have a strong authority on the web according to their page rank. This approach helps me strike a balance between relevance and authority, which are both crucial factors in determining the quality of search results.

    b.

    i. Metric: MAP

Method \Weighting Scheme	**NS**	**WS**	**CM**
GPR	0.0457	0.2636	0.1275
QTSPR	0.0433	0.2636	0.1275
PTSPR	0.0433	0.2636	0.1275

    ii. Metric: Precision at 11 standard recall levels
    Precision at 1

Method \Weighting Scheme	**NS**	**WS**	**CM**
GPR	0.1446	0.8405	0.7719
QTSPR	0.1339	0.8405	0.6846

PTSPR	0.1339	0.8405	0.6846

Precision at 2

Method \Weighting Scheme	**NS**	**WS**	**CM**
GPR	0.0875	0.5926	0.4238
QTSPR	0.0770	0.5926	0.3479
PTSPR	0.0770	0.5926	0.3479

Precision at 3

Method \Weighting Scheme	**NS**	**WS**	**CM**
GPR	0.0786	0.4732	0.2926
QTSPR	0.0724	0.4732	0.1979
PTSPR	0.0724	0.4732	0.1979

Precision at 4

Method \Weighting Scheme	**NS**	**WS**	**CM**
GPR	0.0737	0.3781	0.2093
QTSPR	0.0697	0.3781	0.1402
PTSPR	0.0697	0.3781	0.1402

Precision at 5

Method \Weighting Scheme	**NS**	**WS**	**CM**
GPR	0.0699	0.3145	0.1708
QTSPR	0.0657	0.3145	0.1140
PTSPR	0.0657	0.3145	0.1140

Precision at 6

Method \Weighting Scheme	**NS**	**WS**	**CM**
GPR	0.0653	0.2430	0.1140
QTSPR	0.0611	0.2430	0.1007
PTSPR	0.0611	0.2430	0.1007

Precision at 7

Method \Weighting Scheme	NS	WS	CM
GPR	0.0534	0.1677	0.0899
QTSPR	0.0493	0.1677	0.0673
PTSPR	0.0493	0.1677	0.0673

Precision at 8

Method \Weighting Scheme	NS	WS	CM
GPR	0.0300	0.0915	0.0552
QTSPR	0.0271	0.0915	0.0324
PTSPR	0.0271	0.0915	0.0324

Precision at 9

Method \Weighting Scheme	NS	WS	CM
GPR	0.0115	0.0550	0.0127
QTSPR	0.0115	0.0550	0.0120
PTSPR	0.0115	0.0550	0.0120

Precision at 10

Method \Weighting Scheme	NS	WS	CM
GPR	0.0074	0.0388	0.0075
QTSPR	0.0073	0.0388	0.0081
PTSPR	0.0073	0.0388	0.0081

Precision at 11

Method \Weighting Scheme	NS	WS	CM
GPR	0.0041	0.0101	0.0040
QTSPR	0.0040	0.0101	0.0041
PTSPR	0.0040	0.0101	0.0041

iii. Metric: Wall-clock running time in seconds

Method \Weighting Scheme	NS	WS	CM
GPR	6,985/4.50sec	6,985/5.338sec	6,985/5.422sec
QTSPR	7070.5/ 18.222sec	7070.5/18.954	7070.5/18.221sec
PTSPR	7070.5/14.538sec	7070.5/14.490	7070.5/15.294sec

c. Looking at the results, it's clear that using just the page rank alone doesn't work well. It seems like relying solely on page rank to judge a document's importance isn't the best approach here. But when we factor in the search relevance score, the performance considerably improves. It seems like the search relevance score really helps in making our rankings more accurate. Documents that are highly relevant to the search query tend to end up higher in the ranked list, which makes sense. It seems like users want results that match what they're looking for.

Now, when we compare the custom method (CM) with the weighted sum approach (WS), it's interesting to see that the weighted sum generally performs better. This probably has to do with the weighted sum allowing for a more flexible combination of the search relevance score and the page rank score. On the other hand, the custom method, which just multiplies the search relevance score and the page rank score, performs inaccurately when the page rank score is low.

In those cases, even if a document is highly relevant according to the search engine, it might still end up with a low overall score because of the low page rank score dragging it down. So, all in all, it seems like relying on both page rank and search relevance score is the way to go. When combining them, using a weighted sum approach gives us better results than just straightforward multiplication.

Additionally, there appears to be no significant difference among the various types of page rank: Global PageRank (GPR), Query-based Topic Sensitive PageRank (QTSPR), and Personalized Topic Sensitive PageRank (PTSPR) when using weighted sum or custom method for combining the search scores with the page rank results.

The average running time for the page rank variants is given above. GPR took the least time to run. This is expected as it doesn’t need to look at the document

topic file to build a topic page rank. For QTSPR and PTSPR, I have put the average running time for the twelve topic sensitive page ranks I have trained.

d. Analyzing these algorithms and parameters, and reflecting on my experience with PageRank algorithms, I've come across some interesting observations. Implementing the PageRank algorithm was quite challenging for me. The main problem was dealing with the large sparse Probabilistic Transition Matrix

   M. When I tried to use the classroom formulation directly, I ran into memory allocation problems. Even when I attempted to compress the array using scipy.sparse.csr_matrix, I still faced computation lag, indicating the complexity of handling such large matrices efficiently.

   However, I didn't give up. I found a solution by reformulating the classroom formula. Instead of processing the entire matrix at once, I modified the computation process to read one line of the M matrix at a time. This adjustment helped me overcome memory allocation issues and improve computational efficiency, making the implementation smoother and more manageable.

e.
   i. By using machine learning-based analysis of the user query, search engines can explore novel ways to estimate whether a query can benefit from personalization. One approach involves building a labeled dataset and training a classifier. This would allow the search engine to understand which queries are more likely to benefit from personalized results. Additionally, leveraging user feedback to assess the relevance of query results can provide valuable insights into whether personalization is beneficial for a particular query.
   ii. Similarly, employing machine learning techniques can also lead to innovative methods of identifying the user's interests, such as analyzing the user's topical interest by using a topic classification of the queries the user has searched in the past.

3.

a. In designing my software architecture, I choose to use a class-based approach to better organize my code to build the typical PageRank and TopicSensitivePageRank. Additionally, I utilized reusable functions to avoid redundancy and simplify routine calls. This, I believe, improves the efficiency and maintainability.
b. For speeding up the computation of PageRank, I primarily relied on NumPy arrays, which offer efficient handling of large matrices and operations.
c. I used Python 3.8.10 on Windows 11 as my programming environment. The primary library I utilized was NumPy for efficient array operations.
d. The strengths of my code include its modularity, achieved through the use of classes and reusable functions, which enhances code organization and

maintainability. However, weaknesses include the lack of parallelization in the PageRank algorithm, which could have improved computation speed, and the potential for faster implementation using a language like C++ instead of Python.