Name: Liruoyang
Andrew ID: liruoyay

Machine Learning for Text Mining
Homework 3 - Template

1. Statement of Assurance

I certify that all of the material that I submit is original work that was done only by me.

2. Experiments

a) Describe the custom weighting scheme that you have implemented. Explain your motivation for creating this weighting scheme.

$$s_i = \frac{r_{max} - r_i}{r_{max} - r_{min}} \times w_1 \times pr_i + w_2 \times r_i$$

where, s_i is the final score of doc i, r_{max} is the max relevance score, r_{min} is the min relevance score, r_i is the relevance score for doc i, pr_i is the PageRank score for doc i, w_1 and w_2 are the weights.

My motivation for creating this is that I intend to let the docs with high relevance get less influence from the PageRank and stay high on the ranking list. Because high relevance means good result for a query. And as we go down the relevance list, the PageRank comes into play, letting documents that are not quite relevant but are authoritative go up higher.

b) Report of the performance of the 9 approaches.

I. Metric: MAP

Method \ Weighting Scheme	NS	WS	CM
GPR	0.0584	0.3105	0.3133
QTSPR	0.0563	0.3034	0.3111
PTSPR	0.0637	0.3087	0.3140

II. Metric: Precision at 11 standard recall levels

Method \ Weighting Scheme	NS	WS	CM
GPR	0.1991 0.1238 0.0989 0.0894 0.0789 0.0729 0.0598 0.0277 0.0086 0.0046 0.0036	0.8163 0.6516 0.5558 0.4989 0.4166 0.3102 0.2244 0.0977 0.0553 0.0383 0.0090	0.8227 0.6551 0.5628 0.4999 0.4219 0.3152 0.2273 0.0989 0.0556 0.0387 0.0089

QTSPR	0.1956 0.1167 0.0935 0.0854 0.0779 0.0714 0.0616 0.0302 0.0101 0.0053 0.0037	0.8011 0.6309 0.5494 0.4846 0.4063 0.3066 0.2254 0.0958 0.0533 0.0406 0.0087	0.8295 0.6490 0.5591 0.4948 0.4180 0.3100 0.2271 0.0977 0.0545 0.0408 0.0087
PTSPR	0.2115 0.1353 0.1071 0.0988 0.0886 0.0782 0.0659 0.0295 0.0106 0.0053 0.0037	0.8107 0.6442 0.5519 0.4985 0.4151 0.3125 0.2263 0.0949 0.0537 0.0406 0.0087	0.8318 0.6587 0.5631 0.5021 0.4201 0.3150 0.2275 0.0970 0.0546 0.0409 0.0087

III. Metric: Wall-clock running time in seconds

Method \ Weighting Scheme	NS	WS	CM
GPR	0.00054	0.00054	0.00054
QTSPR	0.00361	0.0037	0.0037
PTSPR	0.00356	0.00375	0.00365

IV. Parameters

GPR: alpha 0.9, beta 0.1

PTSPR, QTSPR: alpha 0.8, beta 0.15, gamma 0.05

Weights: w_1(weight for PageRank scores) 1000, w_2(weight for relevance scores) 0.5

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

As we see from the metrics above, as the algorithm can utilize more and more information, the results usually get better and better.

In terms of PR strategies, it seems PTSPR has the best performance, GPR next, QTSPR the least. I think the reason for this is that personal interests are more relevant than query classification.

In terms of weighting strategies, CM performs best, and NS performs least. This makes perfect sense, because the metrics measure the relevance of the documents retrieved. In other words, the results should be very specific and close to the query. Therefore, NS performs poorly, because PageRank is rather a global result, which is not specific to any query. And CM further solidifies the influence of the relevance scores, hence the better performance.

In terms of time efficiency, GPR runs fastest because it does much less iterations because it doesn't need to compute the topic sensitive PageRanks for every topic, while PTSPR and QTSPR do. Weighting strategies don't make any difference in the time taken, since retrievals are relatively cheap computations.

d) Analyze these various algorithms, parameters, and discuss your general observations about using PageRank algorithms.

I think algorithms have been analyzed in part c).

As for parameters, I observed that when I decrease alpha and increase beta in the TSPR computation, the metrics usually get better. But I feel it's some kind of overfitting, in which the algorithm takes less and less consideration of the authoritativeness of the documents. For the weighting parameters in retrieval, the metrics usually get better if I give relevance score more weight. Maybe because the validation set is relevance based?

After the experiments, my observation is that PageRank alone can barely produce good results for information retrieval tasks, because it's not designed to be specific to the queries. It's more of a authority detection tool, rather than a relevant information retrieval tool. Thus, I think in information retrieval, PageRank can only work as an adjunct to refine the results.

e) 1. What could be some novel ways for search engines to estimate whether a query can benefit from personalization?

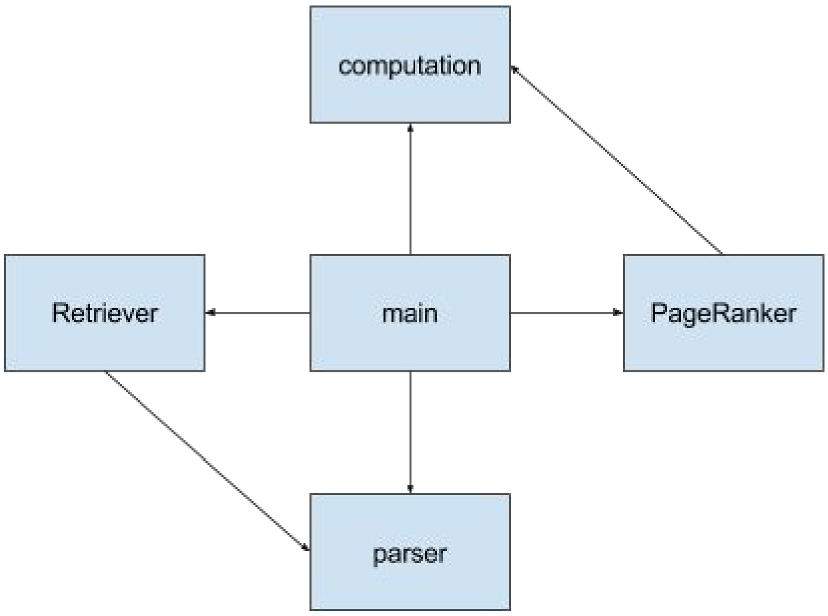
One way I can think of is training a classifier to classify a query as personal or non-personal. If it's personal, it should be able to benefit from personalization.

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?

If we can get a sample of a user's searching or browsing records, we can classify the records into topics, and then we can compute a approximate distribution based on that.

3. Details of the software implementation

a) Describe your design decisions and high-level software architecture;
Architecture:



The main module is the user interface, which reads user input coordinate other modules to do the computations and finally output the results.

The PageRanker class holds all the logic for PageRank algorithms. It perform computations according to the parameters.

The Retriever class is responsible for retrieving documents based on the PageRank results and relevance scores. It holds the weighting logics.

The computation and parser modules are helper modules that are responsible for doing mathematical computations and parsing various text files.

b) Describe major data structures and any other data structures you used for speeding up the computation of PageRank;

Transposed transition matrix: stored as in sparse matrix format as a dictionary of lists of tuples. Every key k locates a inner list representing a row, in which each tuple in the format of (doc_id, weight), meaning doc k has a outlink to doc doc_id. So, when iterating, it only needs to compute the products of a few elements in the sparse rows and the corresponding elements in the PageRank vector, which speeds up the computation.

PageRank vector: a list of numbers.

Doc-topic dictionary: doc_id and topic_id pairs. This helps to quickly find which topic a doc is in when computing TSPR.

c) Describe any programming tools or libraries that you used;

Tools: Jetbrains Pycharm IDE 4.5.4

Libraries: No external libraries used.

d) Describe strengths and weaknesses of your design, and any problems that your system encountered

Strengths:
Pagerank converges very quickly.
Parameter settings don't lead to overfitting because I didn't give beta a large value.
Can automatically perform PageRanking and retrieving.
Loosely coupled design maximizes reusability and easiness of maintenance.

Weakness:
In terms of MAP, incorporating PageRank to the equation doesn't seem improve the results effectively.

Problems encountered:
None.

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

Environment: python 2.7

How to run:

1. Locate the main.py script in the top directory after unzipping.
2. Use command "python main.py [NO] GPR|PTSPR|QPSTR [NS|WS|CM]" to run the program. The PageRank type, GPR or PTSPR or QPSTR, must be provided. If "NO" is provided as the first argument, the software won't run PageRank. If NS or WS or CM are provided, the software will run retrieval accordingly as the last step. Otherwise, it won't run retrieval.
3. The program stores the resulting PageRanks in the pageranks subdirectory, and the retrieval results in the retrieval subdirectory. The files of the results are named according to the PageRank types and the weighting strategies.
4. Make sure hw3-resources directory presents and is in the format as what the handout looks like (already included in the zip).
5. The sample files required are located in the Sample-Files directory.