CS6501 Information Retrieval Homework

Tianye Song, ts7fx

4/9/17

1. Copy and paste your implementation of each ranking algorithm and evaluation function into your report, together with the corresponding final MAP/P@10/MRR/NDCG@10 performance you get from each ranking function. *Use the default parameter settings suggested*[*here*](http://www.cs.virginia.edu/~hw5x/Course/IR2017/_site/mps/2017/03/29/mp/#default) (30pts + 20pts)
   1. Ranking algorithm & Evaluation function implementations
   2. MAP, P@10, MRR, NDCG@10 performances
2. Please carefully tune the parameters in BM25 and Dirichlet prior smoothed Language Model. Report the best MAP you have achieved and correposnding parameter settings. (15pts)
3. In edu.illinois.cs.index.SpecialAnalyzer.java, we defined a special document analyzer to process the document/query for retrieval purpose. Basically, we built up a pipeline with filters of LowerCaseFilter, LengthFilter, StopFilter, and PorterStemFilter. **Please disable some of the filters**, e.g., without stopword removal or stemming, and test the new analyzer with the **BM25** model (with your best parameters of step II). What is your conclusion about the effect of document analyzer on retrieval effectiveness? (15pts) ***Note: this analyzer has to be used in both indexing time and query time!***
4. With the default document analyzer, choose one or two queries, where TF-IDF dot-product model performed significantly better than Boolean dot-product model in average precision, and analysis what is the major reason for such improvement? Perform the same analysis for TF-IDF dot-product model v.s. BM25, and BM25 v.s. Dirichlet Prior smoothed Language Model and report your corresponding analysis (using your best parameters for BM25 and Dirichlet Prior smoothed Language Model). (10pts)
5. Pick one of the previously implemented scoring functions out of
   1. Okapi BM25
   2. Pivoted Length Normalization
   3. Langauge Model with Dirichlet Smoothing  
      to analyze under what circumstance the chosen scoring function will mistakenly favor some less relevant document (*i.e.*, ranks a less relevant document than a more relevant one).

After reading the paper [An Exploration of Axiomatic Approaches to Information Retrieval](http://www.eecis.udel.edu/~hfang/pubs/sigir05-axiom.pdf), how do you think you can fix the problem? Please relate your solution and corresponding implementation in the report. Also report the corresponding ranking performance of your revised ranking algorithm. (10pts)

Copy and paste your implementation of each ranking algorithm and evaluation function into your report, together with the corresponding final MAP/P@10/MRR/NDCG@10 performance you get from each ranking function. *Use the default parameter settings suggested*[*here*](http://www.cs.virginia.edu/~hw5x/Course/IR2017/_site/mps/2017/03/29/mp/#default) (30pts + 20pts)

* Ranking algorithm & Evaluation function implementations
  + Ranking algos:
  + Evaluation functions are in the evaluate.java file. There are 4 of them:
    - MAP
    - P@10
    - MRR
    - NDCG@10 <- where do we get the relevance label??
* MAP, P@10, MRR, NDCG@10 performances