CS 599(02) - Information Retrieval

Programming Assignment #3

<u>Due date</u>: 11/06/14 23:59 hrs

Statement

So far, we have implemented only Boolean Retrieval model where we did not consider any order within retrieved documents. In this assignment, we will extend our previous assignments to incorporate scoring to rank all the retrieved documents.

There are many ways to score retrieved documents. For this assignment you need to implement the following for a consistent evaluation:

- Use TF-IDF for term weighting
- Use cosine similarity for scoring and ranking the documents based on user query

Please refer to 'Details' section for more information.

You need to satisfy the following requirements in this assignment: (1) index should be as efficient as possible for all types of queries (2) size of index has to be as small as possible, so that it fits in memory making search as fast as possible (3) indexing can take more time, as user does not wait for it.

The credit will be based on your accuracy, memory usage and time.

Details

Model

Please note the following details on algorithm to better match with golden result:

- TF: Use logarithmic scaled frequency model
- IDF: There is no need to use smoothing.
- Try to best possible cosine similarity to score all documents against query.
    - Note that, in classical search problem, you can reach all documents from terms but cannot enumerate all terms for a document. So ***at the time of computing similarity with a document enumerating over all terms from that document is not allowed***.

Implementation

You have to implement two methods for Ranked Retrieval:

- `index(String dir)`
    - `index()` supposed to go over all files under 'dir'. There will be no subdirectories inside it.

    - To be consistent in indexing, a rudimentary `tokenize()` method has been provided. ***Please don't tweak that, which might cause different result***.
- `retrieve()`
    - `retrieve()` supposed to return an ordered list if filenames of all the documents under 'dir' that best answers the given query. Note that, only basename of the files are to be returned, not the full path.
    - For your convenience, I have implemented some sorting code for you. Feel free to retain that and only implement `scoreAllDocuments()` which is called by the `retrieve()` method.
- For your convenience, a simple linked list implementation is also retained, feel free to use or delete it.

Code

Download PA3.zip from the Blackboard under Assignments → Programming Assignments → PA3.

Code Structure

`data/` → This directory contains some news data under `test`/, which will be used as corpus for this assignment. `test.txt` is the input search operation type, query and the retrieved documents.
`java/` → This directory contains all the JAVA code. <u>You only need to modify `RankedRetrievalModel.java`</u>
`run.py` -> A tool to compile, run & test your code.

Extract and Run

Download the file PA3.zip. Extract it. Let's say you extracted into 'PA3' directory.
From terminal:

```
$cd PA3
# To run and evaluate your code
$./run.py
```

Compatibility

Note that, all assignments (including this one) will be tested under Linux environment with Python and Oracle Java is installed. Given code might work on other platforms (like Windows, etc.) but has not been tested. Hence, it is encouraged to develop and test your code in a Linux based environment.

Submission

<u>You should only modify and upload `RankedRetrievalModel.java`</u> on Blackboard. **Any change in other files will not be accepted and you will not be evaluated in that case**.

Evaluation

There is some held out data and query set against which your code will be tested and evaluated. Hence, it is encouraged not to write generic code that work in most of the cases. Your main aim is to write an efficient data structure that should index and retrieve the documents very fast with the query operations. You are free to read different techniques and use them to improve your accuracy. You will be evaluated on the basis of accuracy in retrieval, speed and memory usage.

As we are going to evaluate ranking of individual docs retrieved and ranking is based on some real number (double or float), we will follow following evaluation method:

- **Correctness:** Weight: 5
    - We will consider only first 20 results for every query (feel free to employ any optimizations around this to increase score in memory and runtime).
    - Due to floating point comparison and different similarity model, ranking might vary. At the same time, there is no single right answer. The scoring is based on
        - Match between the top ranked result of yours with the golden data
        - Difference in rank, where maximum diversion is computed
        - You can also look the comments in the computeScore method (of ProcessFile.java) for the coditions
    - Thus your score will be based on the MSE
    - If the number of returned documents per query term is less than 20, then the comparison will be done with whatever documents present.
- **Algorithm:** As mentioned above, there is no right answer, some score will be given based on similarity algorithm employed. This is manual process and hence please comment enough about your theory, link to papers or articles from where you got it. Weight: 5
- **Memory Footprint:** Lower is better. Weight: 4
- **Runtime:** Lower is better. Weight: 6

Honor Code

I encourage students to discuss the programming assignments including specific algorithms and data structures required for the assignments. However, students should not share any source code for solution.

Code exists on the web for many problems including some that we may pose in problem sets or assignments. Students are expected to come up with the answers on their own, rather than extracting them from code on the web. This also means that we ask that you do not share your solutions to any of the homework - programming assignments, or problem sets - with any other students. This includes any sort of sharing, whether face-to-face, by email, uploading onto public sites, etc. Doing so will drastically detract from the learning experience of your fellow students.