CS 753/853: Topics / Information Retrieval Fall 2018
Programming Assignment 2: Evaluation Measures and TREC_EVAL

About the assignments

- **Grading:**
  1. Programming Assignments need to be worked on in a team. If you have not yet done so, find team mates and self-sign up for a team on mycourses.
  2. The homework is due 1 week after release.
  3. The assignments are mandatory.
  4. The points shown below are for your information only. The grading and feedback you get will only be fail or pass.
  5. Flawless work submitted on time may earn extra credits.
  6. Tasks marked with "**Graduate students:**" are only mandatory for graduate students and optional for undergraduates.

- **Provide a URL to github or upload an archive containing:**

  source code The source code and all *your* files (no data files!)

  results.pdf One PDF explaining your results and observations.

  install.pdf One PDF or Text file describing
  1. how to install your code and all its dependencies and necessary libraries (on Linux).
  2. which compiler and which version is necessary to compile your code.
  3. how to run your code to reproduce your results.
  4. Consider providing a maven file (for Java/Scala), a setup.py (for python), or a bash script and explain how to use it.

  Your code must be installed within 10 minutes.

- **Please list on the results.pdf of your submission:**
  1. For each team mate: Full name, UNH id, Webcat username, and the Assignment number.

- **Heads up:** You will have to perform peer reviews for this homework. After the due date, you will be assigned some other team's code that you have to install, run, and report results.

Introduction

From Programming Assignment 1, use

- Lucene 6 or higher: `http://lucene.apache.org/`,

- Download the TREC Complex Answer Retrieval “test200“ dataset of release-v2.1 or v2.0 and unpack: `http://trec-car.cs.unh.edu/datareleases/v2.0/test200.v2.0.tar.xz`
- Tools to read the data, trec-car-tools—available for Python3 and Java 1.7 (or higher) are provided on the data release page (see “Support Tools”: `http://trec-car.cs.unh.edu/datareleases/`)
- your code that implements:
    1. Lucene’s default ranking function and
    2. The custom scoring function: $score(D, q) = \sum_i \#\{q_i \in D\}$,

Use your previous homework for verify that your code is correct, and if necessary correct your code before starting this assignment.

Feel free to share code and solutions for Programming assignment 1 on Piazza. I want to encourage you to ask (and answer) questions about this assignment on Piazza.

1 Read Queries and Write Rankings

The test200 benchmark contains a file called `./test200/test200-train/train.pages.cbor-outlines.cbor`. This file contains the page id and a page name together with other things we don’t care about in this task.

Write code that, given a retrieval function

1. Use the trec-car-tools to extract page id and page name
2. take the page name as a *keyword query*, and use the page id as a *queryId*
3. For each keyword query, produce a ranking of the top 100 paragraphs
4. Write all rankings to the same file using this format (rankings concatenated, one line per ranked item):
   `$queryId Q0 $paragraphId $rank $score $teamname-$methodname`

This is called the “run file format”. We call this the *run file* in the following. The $paragraphId will look like a string with random hexadecimal numbers—please only include this paragraphId in your runfile. Do not include Lucene’s internal document id. Do not include the full text.

For each of the following retrieval models create one run file containing rankings for all queries in the benchmark.

1. Lucene’s default ranking function and
2. The custom[1] scoring function: $score(D, q) = \sum_i \#\{q_i \in D\}$,

For each of the two files, please include the first 5 lines in your report.

[1]If you did not implement the custom scoring function yet, please ask another student for her/his solution.

2 Evaluation with trec_eval

Download, compile, and familiarize yourself with the evaluation program *trec_eval*: `http://trec.nist.gov/trec_eval/`

The test200 benchmark contains a file called `./test200/test200-train/train.pages.cbor-article.qrels`. We call this the *qrel file* in the following.

Use the trec_eval tool to compute the following measures for both retrieval models (using the run files form above)

1. Precision at R ("RPrec")
2. Mean-average precision ("map")
3. NDCG@20 ("ndcg@20")

3 Precision at R

In a programming language of your choice, implement the Precision at R metric. Your code must handle the case where some relevant documents are missing in the ranking. When computing the arithmetic mean across all queries, don't forget that queries without results should get a zero score.

The qrel file contains data on which paragraphs are relevant (1) and which are non-relevant (0 or missing). The format is:
`$queryId 0 $docId $isRelevant`
In this case, queryIds refer to pages, and docIds refer to paragraph ids. When $isRelevant is 1 the paragraph/document is relevant, if $isRelevant is 0 or not contained in the file, it is non-relevant.

Write code to read the qrel file and use the relevance information to compute the precision at R score for your two rankings. Keep in mind that a document that is relevant for one query, might not be relevant for a different query, so don't forget to store the query id with your relevance data.

In your report, include the Precision at R evaluation score obtained by each of the scoring functions. Are you obtaining the same Precision at R evaluation score as trec_eval? If not, look for the error and/or post on Piazza.

4 Graduate students: Mean average Precision

In a programming language of your choice, implement the mean-average Precision measure as described in the lecture. Don't forget the case where a relevant paragraph is missing in the ranking.

Use the qrel file to load the ground truth of which paragraphs are relevant.

In your report, include the Mean-average evaluation score obtained by each of the scoring functions. Are you obtaining the same Mean-average evaluation score as trec_eval? If not, look for the error and/or post on Piazza.

5 Graduate students: NDCG@20

In a programming language of your choice, implement the NDCG@20 measure as described in the lecture. Don't forget to normalize with the ideal ranking loaded from the qrel file (first all relevant documents, then the rest).

In your report, include the NDCG@20 eval score obtained by each of the scoring functions. Are you obtaining the same NDCG@20 eval score as trec_eval? If not, explore how your NDCG@20 implementation is different from the one in trec_eval, which is based on this paper by Jarvelin and Kekalainen (ACM ToIS v. 20, pp. 422-446, 2002): `http://www.sis.uta.fi/infim/julkaisut/fire/KJJK-nDCG.pdf`

6 Analysis of your Findings

Read about how to use standard error for statistical analyses:

- `http://www.biostathandbook.com/standarderror.html`
- `http://egret.psychol.cam.ac.uk/statistics/local_copies_of_sources_Cardinal_and_Aitken_ANOVA/errorbars.htm`

Using your own code, or trec_eval (using the -q option), produce evaluation scores for each query and compute the arithmetic mean and the standard error.

Of measures $x_1, x_2, ...x_{|Q|}$, the arithmetic mean $\bar{x}$ is computed as $\bar{x} = \frac{1}{|Q|}\sum_i x_i$

The standard error Δ is computed as $\Delta = \frac{1}{\sqrt{|Q|}}\sqrt{\frac{\sum_i (x_i - \bar{x})^2}{|Q|-1}}$, where the second factor is the standard deviation from the mean.

Compute mean and standard error of both retrieval functions and these evaluation measures:

1. Precision at R (“RPrec”)
2. Mean-average precision (“map”)
3. NDCG@20 (“ndcg@20”)

Create a plot showing mean +/- error bars, or equivalently $\bar{x} + / - \Delta$

Which retrieval function is better? Is this difference significant (i.e., are the error bars overlapping)?