Northeastern University

College of Computer and Information Science
Information Retrieval CS6200 – Fall 2018

Overview: You have been introduced to the core information retrieval concepts and processes throughout the course of this semester. In this project, you will get to put these into practice by building and using your very own retrieval systems!

Goals: Design and build your information retrieval systems, evaluate and compare their performance levels in terms of retrieval effectiveness

Dataset: CACM test-collection which is comprised of the following:

1- Textual corpus: cacm.tar.gz (3204 raw documents – except for Task 3 part B: use stemmed version cacm_stem.txt)
2- Queries (64 unprocessed cacm.query – except for Task 3 part B: use cacm_stem.query)
3- Relevance judgments (cacm.rel)
4- Stoplist: common_words.txt

Team: Teams of **3 members** are to be formed. Teams across the two sections are allowed. Declare team members on the designated Piazza post by Wednesday November 28 5:00pm. Once formed, teams cannot be altered.

Milestones:

November 21st: Release of the online description for the project
November 28 by 5:00pm: Team declaration due
December 8th by 11:59pm: Project & report (implementation & documentation) submission due.

Assessment: The project will be graded out of 100 points and then scaled to 20% of your overall grade (see syllabus for course grade details)

Implementation: 75 points (detailed point breakdown in project task descriptions)

Documentation: 25 points. Project submissions lacking documentation (report) will NOT be accepted and hence will NOT be graded at all.

Extra credit: 20 points: All or nothing. Awarded credit applies to project & home work.

Academic honesty: If you get help from others, you must write their names down on your submission and explain how they helped you. If you use external resources you must mention them explicitly. You may use third party libraries but you need to cite them, too.

Project Description:

Implementation – Phase 1 :: Indexing and Retrieval

Task 1 (15 points): Build your own retrieval systems:

A- From scratch! (You may re-use your indexer and searchers from HWs 3 and 4)

B- Using Lucene: an open source library that provides indexing and searching functionalities (you may re-use your code from HW 4)

Task 1 Output: *Four baseline runs*.

Setup: Use *BM25*, *tf-idf,* and *JM Smoothed Query Likelihood Model (λ = 0.35)* as retrieval models combined with your word unigram indexer. The fourth run uses Lucene's default retrieval model. The top 100 (at most) retrieved ranked lists (one list per run/retrieval system) are to be reported.

Task 2 (15 points): Pick *one*[1] of the four runs above and perform *query enrichment.* You may use the any approaches explained in the course (e.g.: query time stemming, pseudo relevance feedback, semantic query expansion, etc.) or adopt other ones. Make sure to properly cite related literature and resources. Justify your design decisions, technical choices, and parameter setting and back them up with demonstrated evidence from literature and/or experiments whenever applicable.

Task 2 Output: *One* query enrichment run.

Task 3 (20 points): Perform the following on *three* baseline runs of your choice:

A- Stopping (using common_words.txt) with no stemming.

B- Index the stemmed version of the corpus (cacm_stem.txt). Retrieve results for the queries in cacm_stem.query. Examine and analyze the results for three queries that you find interesting.

Note: When parsing the documents, you may ignore the digits that commonly appear in the end of the documents' content.

Task 3 Output: *Six runs:* using *three* baselines X 2 variations (with stopping, with the stemmed corpus and stemmed query subset).

Implementation – Phase 2 :: Displaying Results (10 points)

Implement a *snippet generation* technique and *query term highlighting* within results in one of the baseline runs. It is for you to figure out the techniques, however, you are required to back up your choices with the algorithm(s)/ technique(s) details and cite the respective literature.

[1] In practice, it is advised to perform Tasks 2 and 3 using all three base search engines from Task 1 to obtain a broad view of indexing strategies and retrieval models combinations. This, however, is not mandatory for this project

Implementation – Phase 3 :: Evaluation (15 points)

By now, you should have *eight* distinct runs with results for all 64 queries. Namely, 4 baseline runs, 1 query refinement run, and 3 stopping runs (we're not counting the stemming runs here). It is now time to assess the performance of your retrieval systems (runs) in terms of their effectiveness.

Implement and perform the following (do NOT use TREC-Eval):

1- MAP
2- MRR
3- P@K, K = 5 and 20
4- Precision & Recall (provide full tables for all queries and all runs)
5- Plot a Recall-Precision curve (one curve per run. All runs in one figure)

Note: Queries that don't have any entries in the relevance judgment should be excluded from evaluation.

Documentation (25 points):

A- ReadMe.txt: Explain in detail how to setup, compile, and run your project.
B- Report NOT to exceed 2000 words[2] in PDF format, named as follows:

firstNameInitialLastName1_firstNameInitialLastName2[_firstNameInitialLastName3].pdf

Please follow this structure:

i. First page: Project members' names, course name and semester, instructor name.
ii. Introduction: Short description of your project, detailed description of each member's contribution to the project and its documentation
iii. Literature and resources: overview of the techniques used (chosen query refinement approach, snippet generation approach) citing the scholarly work and research articles you used to back up your technique and algorithm choices.
iv. Implementation and discussion: More thorough description of your project and design choices. Include query-by-query analysis in this section.
v. Results: tables reporting all results obtained for all runs and queries for all required metrics. For query level results, please provide tables/spreadsheets, too.
vi. Conclusions and outlook: state your findings, observations and analyses of the results. Which system do you think works best? Why? For "outlook": write a few sentences stating what you would envision doing to improve your project, what other features you would choose to incorporate.
vii. Bibliography: citations and links to resources and references

[2] This document's word count is about 1000 words

viii. Don't forget to submit your code! Make sure to include intermediate result files (e.g. estimated language models)

Extra credit (20 points):

This part is optional, and is all or nothing. Awarded extra credit points apply to project and homeworks.

Design and implement an "advanced search"[3] tool that allows for users to choose certain search criteria such as searching an exact phrase or finding results that contain *any* of the query terms. You should think of this as building a query language and supporting it in your retrieval system's infrastructure. Your system should **always** rank by **relevance**.

Your retrieval system should allow the following options:

1- ***Exact match*** search: all query terms must appear and in the same order

2- ***Best match*** search: A document is shown in the results if it contains at least one query term.

3- ***Ordered best match within proximity N*** search: same as above but order matters and any two query terms should separated by no more than *N* other tokens.

You do not need to implement a GUI (graphical user interface) for your retrieval system. Instead, your program may provide a command line interface and/or allow parameter passing to select the type of search.

[3] Inspired from Google's advanced search: https://www.google.com/advanced_search?q=google&hl=en