IRDM 2020
February 12, 2021

IRDM Course Project Part I

Sahan Bulathwela,
m.bulathwela@ucl.ac.uk

Omer Kirnap,
omer.kirnap.18@ucl.ac.uk

1 Task Definition

An information retrieval model is an essential component for many applications (e.g. search, question answering and recommendation). Your task in this project is to develop an information retrieval model that solves the problem of passage retrieval, i.e., a model that can effectively and efficiently return a ranked list of short texts (i.e. passages) relevant to a given query.

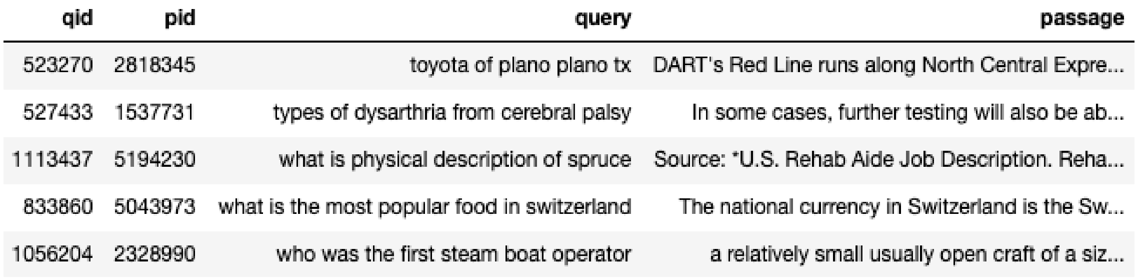
This is an individual project, so everyone is expected to submit their own code and project reports. This is the first part of a larger project, which consists of two components. In the second part of the project, we will be building upon this first part and will be working on building more advanced retrieval models.

In this part of the assignment, our final goal is to build a passage re-ranking system: Given a candidate list of passages to a query (that have already been retrieved using some initial retrieval model that we have developed), re-rank these candidate passages using the retrieval models specified in the assignment.

2 Data

The dataset you will be using is available through this url. Our dataset consists of 3 files:

- *test-queries.tsv* is a **tab** separated file, where each row contains a query ID (qid) and the query (i.e., query text).
- *passage_ collection.txt* contains passages in our collection where each row is a passage.
- *candidate_ passages_ top1000.tsv* is a **tab** separated file, containing initial rankings that contain 1000 passages for each of the given queries in file test-queries.tsv. The format of this file is <qid pid query passage>, where qid is the query ID, pid is the ID of the passage retrieved, query is the query text and passage is the passage text, all tab separated. Figure 1 shows some sample rows from the file.



qid	pid	query	passage
523270	2818345	toyota of plano plano tx	DART's Red Line runs along North Central Expre...
527433	1537731	types of dysarthria from cerebral palsy	In some cases, further testing will also be ab...
1113437	5194230	what is physical description of spruce	Source: *U.S. Rehab Aide Job Description. Reha...
833860	5043973	what is the most popular food in switzerland	The national currency in Switzerland is the Sw...
1056204	2328990	who was the first steam boat operator	a relatively small usually open craft of a siz...

Figure 1: Sample rows from candidate_passages_top1000.tsv file

3 Subtasks

The course project involves several subtasks that are required to be solved. The four subtasks of this project are described below.

1. Text Statistics (20 marks). Perform any type of pre-processing on the collection as you think is required. Implement a function that counts the frequency of terms from the provided dataset, plot the distribution of term frequencies and verify if they follow Zipf's law. Report the values of the parameters for Zipf's law for this collection. You need to use the full collection (file named *passage_collection.txt*) for this question. Generate a plot that shows how the results you get using the model based on Zipf's law compare with the values you get from the actual collection.

2. Inverted Index (20 marks). Build an inverted index for the collection so that you can retrieve passages from the initial set of candidate passages in an efficient way. To implement an effective inverted index, you may consider storing additional information such as term frequency and term position. Report what type of information you have stored in your inverted index. Since your task in this project is to focus on re-ranking candidate passages you were given for each query, you can generate a separate index for each query by using the candidate list of passages you are provided with for each query (using the file *candidate_passages_top1000.tsv*).

3. Retrieval Models (30 marks). Extract the tf-idf vector representations of the passages using the inverted index you have constructed. Implement the vector space model and BM25 using your own implementation and retrieve 100 passages from within the 1000 candidate passages for each query. For both the vector space model and BM25, submit the 100 passages you have retrieved in sorted order (sorted in decreasing order – passage with the top score should be at the top) for both models.

4. Retrieval Models, Language Modelling (30 marks). Implement the query likelihood language model with i) Dirichlet smoothing, where $\mu = 2000$, ii) Laplace smoothing, and iii) Lindstone correction with $\epsilon = 0.5$ using your own implementation and retrieve

100 passages from within the 1000 candidate passages for each query. For all three smoothing variants, submit the 100 passages you have retrieved in sorted order (sorted in decreasing order – passage with the top score should be at the top) for both models. Which smoothing version do you expect to work better? Explain.

You should have one file per model (named VS.txt and BM25.txt, LM-Dirichlet.txt, LM-Laplace.txt, LM-Lindstone.txt, respectively), where the format of the file is:

<qid1 A1 pid1 rank1 score1 algoname1>
<qid1 A1 pid2 rank2 score2 algoname1>
<qid1 A1 pid3 rank3 score3 algoname1>
<qid1 A1 pid4 rank4 score4 algoname1>
...

The width of columns in the format is not important, but it is important to have exactly six columns per line with at least one space between the columns. In this format:

- The first column is the query number.

- The second column is currently unused and should always be "A1", to refer to the fact that this is your submission for Assignment 1.

- The third column is the passage identifier.

- The fourth column is the rank the passage/document is retrieved (starting from 1, down to 100).

- The fifth column shows the score (integer or floating point) of the model that generated the ranking.

- The sixth column refers to the algorithm you used for retrieval (would either be VS or BM25, depending on which model you used) .

4 Submission

You are expected to submit all the codes you have implemented for text pre-processing, Zipf's law, inverted index, and retrieval models. All the code should be your own and you are not allowed to reuse any code that is available from someone/somewhere else. You are allowed to use both Python and Java as the programming language.

Additionally, you should also submit five files that contain the retrieval results of the vector space model, BM25 model and language models with the three different smoothing variants in the format that was described above.

You are also expected to submit a written report whose size should not exceed 4 pages, including references. Your report should describe the work you have done for each of the aforementioned steps. Specifically, your report should consist of the following:

1. Describe how you perform the text pre-processing and justify why text pre-processing is required.
2. Explain how you implement Zipf's law, provide a plot comparing your model with the actual collection and report the values of the parameters for Zipf's law for this collection.
3. Explain how you implemented the inverted index, what information you have stored and justify why you decided to store that information.
4. Describe how you implemented the vector space and BM25 models, and what parameters you have used for BM25.
5. Describe how you implemented the language models, and how you expect their performance to compare with each other.

You are required to use the SIGIR 2020 style template for your report. You can either use LaTeX or Microsoft Word templates available from the ACM Website [1] (use the "sigconf" proceedings template). Please do not change the template (e.g. reducing or increasing the font size, margins, etc.).

5 Deadline

The deadline for this part of the assignment is **4:00pm on 23 March 2021 (based on GMT timezone)**. All the material will be submitted via Moodle.

[1] https://www.acm.org/publications/proceedings-template