Software Documentation

NOTE: Link to the presentation video is in the Course Project's `README.md` file.

For Task 1) Metapy compatible with recent version of python (3.8 and 3.9)

Implementation:

- From our repository, when cloned, open the submodule "metapy-Updated-Source-Code" and unzip the metapy.zip file.
- Make sure all submodules are fetched, running the command `git submodule update --init --recursive`, although they should be included in this repo.
- If using windows, make sure you have the appropriate version of cmake installed.
- Then open the build directory.
- Run command `cmake .. -DCMAKE_BUILD_TYPE=Release`.
- Run command `make`
- After a few minutes, this build should succeed, meaning that the metapy library was successfully built.
- It should be written to metapy.so in the build directory

Note: when following these steps using the original metapy library, the user runs into cmake errors failing to download files. This shows that our updates fixes the compatibility issues.

Usage + Overview of Functions:

In this case, metapy should have all usage and functions as the official metapy library, only that building it succeeds on newer versions of python.

Task 2) Implement and Analyze Additional Ranking Functions for MetaPY

Background

Our second task of the project was to research, implement, and analyze different ranking functions to see if there is a significant difference between them, Currently, the existing ranking functions in meta are the following: Absolute Discount, Dirichlet Prior, Jelinek-Mercer, KL-DIvergence, LM Ranker, Pivoted Length, and Okapi-BM25.

For reference, the standard BM25 retrieval function is denoted below as follows:

$$\text{score}(D, Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)}$$

We were curious if the outputs and the mean precisions would be significantly different based on the variations in the ranker function, and after some research we found different variations of BM-25.

Link to page containing research article:
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7148026/

ATIRE

$$\sum_{t \in q} \log\left(\frac{N}{df_t}\right) \cdot \frac{(k_1+1)\cdot tf_{td}}{k_1 \cdot \left(1-b+b\cdot\left(\frac{L_d}{L_{avg}}\right)\right)+tf_{td}}$$

BM25L

$$\sum_{t \in q} \log\left(\frac{N+1}{df_t+0.5}\right) \cdot \frac{(k_1+1)\cdot(c_{td}+\delta)}{k_1+c_{td}+\delta}$$

BM25+

$$\sum_{t \in q} \log\left(\frac{N+1}{df_t}\right) \cdot \left(\frac{(k_1+1)\cdot tf_{td}}{k_1 \cdot \left(1-b+b\cdot\left(\frac{L_d}{L_{avg}}\right)\right)+tf_{td}}+\delta\right)$$

Each of the functions have certain variations. In BM25+ and BM25L, the most notable difference is the introduction of a free parameter.
Each of the functions have trade-offs. The standard BM25 function penalizes both longer documents and shorter documents too harshly. Theoretically, BM25L penalizes longer documents shorter and BM25+ penalizes shorter documents less. BM25 Atire is a modification similar to the standard BM25 with a slight change in the IDF component.

Usage

We used python and the existing metapy library to implement the ranking functions. In order to run the program, you must have python and metapy installed locally.

1. Ubuntu Linux 20.04 LTS works best, as the metapy fix from part 1 works on Linux systems.
2. Python and Metapy installed.
3. Conda virtual python environment to easily switch between python versions (not required, but recommended).
4. Git clone https://github.com/tpjwm/CourseProject.git
5. Within the working repository folder, there are three files, each containing the source code for the different ranking functions: `BM25ATIRE.py, BM25L.py, and BM25PLUS.py`.
6. In order to run the different functions, run the command
   `Python [ranker.py] [config.toml file]`
7. And example run would look like : `Python BM25L.py config.toml`
8. There are two config files: config.toml which is a dataset of standard size and config2.toml which is a dataset of a smaller size.
9. If it successfully runs, the program should output the mean average precision of running that ranker function on the particular dataset.

Results

We made some interesting observations regarding the ranking functions.
We first ran each of the ranking functions we researched along with the standard BM25 function with default parameters.

On dataset of normal length:

Mean Average Precision chart with base parameters: k1 = 1.2, b = 0.75, k3 = 500

Retrieval Function	Mean Average Precision
BM25	0.25511867318944054
BM25+	0.24279468030010365
BM25L	0.24682213123932706
BM25 ATIRE	0.2558733469107809

Here, the mean average precision of BM Atire is the highest, followed by BM25, BM25L, and BM25+. Using the p-value test, we found that there are significant differences between every single one of the functions.

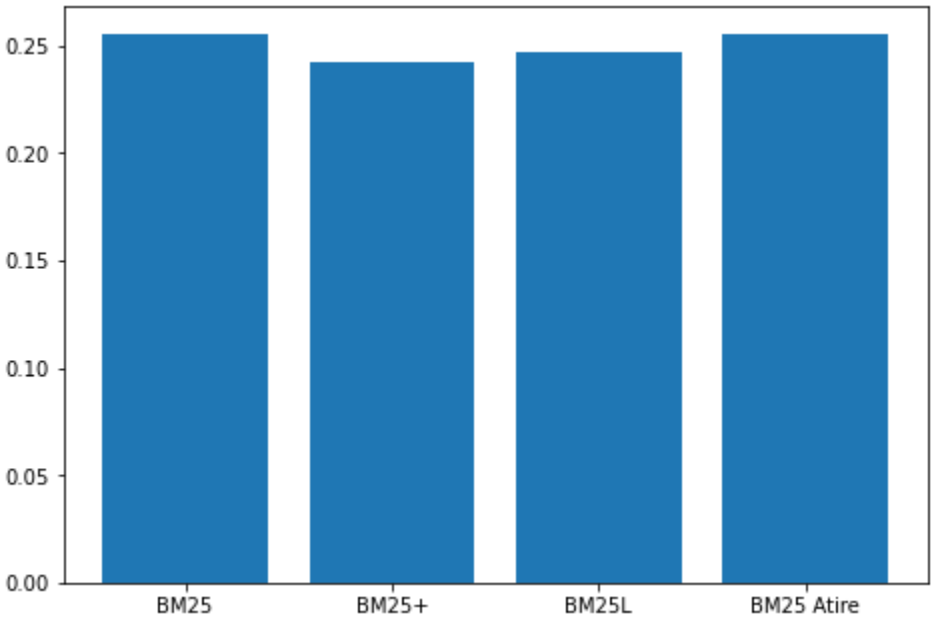
On dataset of small size:

Mean Average Precision chart with base parameters: k1 = 1.2, b = 0.75, k3 = 500

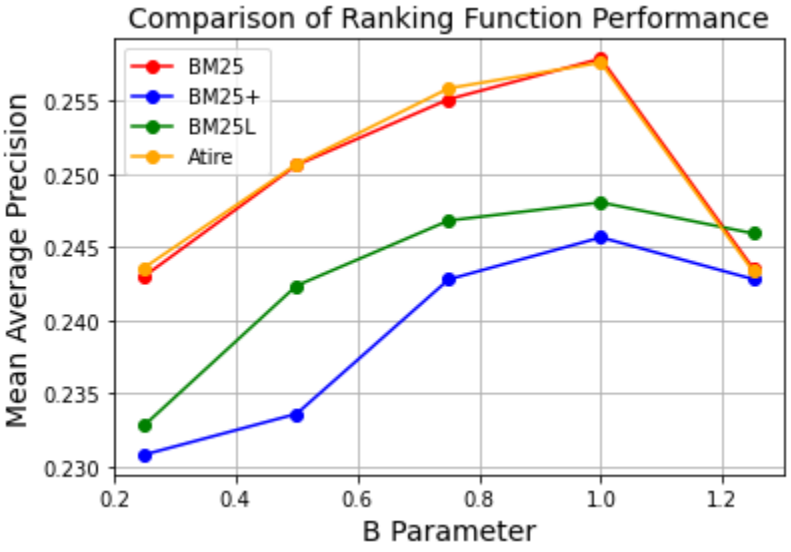
Retrieval Function	Mean Average Precision
BM25	0.3874
BM25+	0.45925
BM25L	0.43030
BM25 ATIRE	0.38980

On the dataset of a smaller size, we found out that BM25L and BM25+ actually have a higher mean average precision than the other corresponding functions.

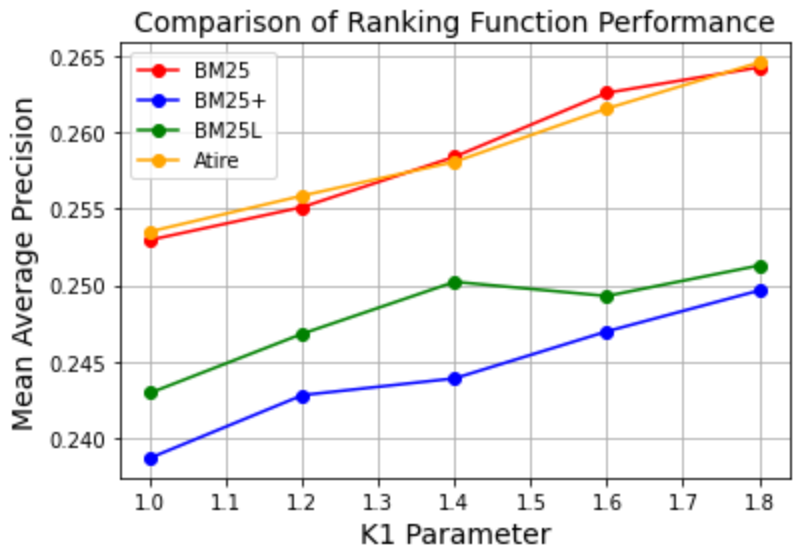
Using the matplotlib and pandas libraries for python, we created several visualizations of the comparisons between ranking functions in order to better communicate our findings.



The above chart displays the mean average precision of 4 ranking functions.



We also modified the parameters of the ranking functions in order to maximize the MAP.
In the above chart, the b parameter was changed in the range of [0.25, 1.25]
We found out that standard BM25 and BM Atire massively outperform BM25L and BM25+ on longer documents, but after the b parameter increases after 1.0, BM25L performs better.
BM25 Atire seems to perform slightly better than BM25 in certain situations.



The k1 parameter was also changed within the range of [1.0, 1.8].
In similar fashion, BM25 Atire and BM25 consistently outperform BM25+ and BM25L.

Overall, in terms of using standard parameters, BM25 Atire and standard BM25 have a higher MAP than the BM25+ and BM25L modifications. BM25+ and BM25L should only be used in niche situations, i.e if the document and the queries are of shorter length. Even though the mean average precision is lower in longer documents, in shorter documents it is far higher and thus it may be the case that the function is scoring the documents more accurately and penalizing the score to a lesser extent. However, BM25 and BM25 Atire seem to still maintain the highest overall mean average precision and for ease of simplicity should be used more often.