CSI 4107
Assignment 2

Neural information retrieval system

Nick Ivanov, 300137667
Bradley Liu, 300125419
Kevin Le, 300053306

University of Ottawa

Link to github repository: https://github.com/nivanov8/csi4107-assignment2

1. **How to run:**
    a. Run: `pip install -r requirements.txt`
    b. In the root directory run:
        i. `python main.py`

- Note: If you run into an error, make sure Windows Long Path is enabled. You might have to check in your Windows Registry Editor to enable it in order to download these packages.

2. **Functionality:**

    a. Preprocessing:

The tokenization for this assignment was done the exact same as Assignment 1, however, we will explain it again in the next paragraphs as it is a big role in information retrieval.

Tokenization: The system tokenizes documents into individual words or tokens, removing punctuation and special characters.

Stop Word Removal: Common stop words such as "and", "the", and "is" are removed to reduce noise in the text.

The preprocessing functionality provided involves several steps to prepare textual data for indexing and retrieval. First, tokenization is performed using the tokenize method, which splits text into individual words or tokens. To ensure consistency, the text is converted to lowercase, and regular expressions are employed to remove punctuation, special characters, and underscores. Additionally, apostrophes are normalized to handle cases like double apostrophes.

The preprocess() method iterates through each document file in the specified folder. Documents are split into individual segments using the <DOC> tag as a delimiter. For each segment, the <DOCNO> tag is extracted to identify the document number, while the <TEXT> tag is used to obtain the textual content. This content is then tokenized using the tokenize() method, and the resulting tokens are stored in a list.

The tokenized text from each document segment is joined back into a single string and added to the corpus, representing the entire collection of documents. The method returns three items: a list of tokens, the corpus of documents, and the mapping of text to document numbers. The return values for the preprocessing used for tf-idf is a bit different due to used a different library.

Note that we also preprocess the documents held in the collection folder differently from the text-queries.txt. As their delimiters are different, in similar fashion we strip the whitespace and build the queries by line and return the list of queries.

b. BM25 retrieval

As mentioned per the project outline, BM25 from assignment 1 was used to retrieve the first 1000 documents to eliminate the need to use a neural model on a large corpus of documents. We used the same BM25 model from assignment 1 (BM25OKAPI) to retrieve the initial subset of documents. This retrieval takes place in the bm25 folder of the project.

c. Neural reranking

After the initial retrieval of the 1000 documents for each query using BM25, we then used 2 models to rerank the results neurally. The first model used was GIST large embedding v0 (model card: https://huggingface.co/avsolatorio/GIST-large-Embedding-v0). This model was selected due to its good performance on the MTEB (Massive Text Embedding Benchmark) Leaderboard on Huggingface. This model is primarily used for sentence similarity and we thought this would be appropriate to use as the nature of embeddings and similarities of embeddings is to capture sentence similarities. This model did the best out of the 2 used. The second model used was MxBai large embedding v1 (model card: https://huggingface.co/mixedbread-ai/mxbai-embed-large-v1.). This was an instruction based model so we had to reformulate the queries such that we can better assist the model in generating embeddings as this was how the model was trained. This model was a close alternative for reranking.

d. Evaluation:

Mean Average Precision (MAP): We compute the MAP score using trec_eval, a standard tool for evaluating information retrieval systems. MAP measures the average precision of the system across multiple queries.

Precision and Recall: We also calculate precision and recall metrics to assess the effectiveness of the retrieval system in retrieving relevant documents.

3. Results

The results obtained from this assignment concluded that using a neural model to rerank the initial documents obtained only slightly increased the resulting MAP score. In Assignment 1 (using purely BM25) we managed to achieve a MAP score of 0.3411. Using our best neural model, GIST Large Embedding V0, we obtained a MAP score of 0.3450, a 1.1% increase in performance. Although not provided, the MAP score obtained for using the MxBai model was 0.3120 and as previously mentioned this was a close alternative to the better performing GIST model.

4. Algorithms and Optimizations

a. Preprocessing

We used regex matching to separate each document and extract information like the title and the text. We then ran the documents through a tokenizer that would transform the text into all lower case, remove stop words, stem and remove any special characters. We would then return a list of tokens to be used for indexing and other information such as the corpus itself and all the doc numbers for easier computations later down the line. This was the exact same technique we used in assignment 1

b. Retrieval and Ranking

We only retrieved a subset of documents from the whole corpus using bm25 first. This drastically reduces the computation time as calculating vectors for the text can take a long time even for 1000 documents. This method helped reduce the computational power and sorting the documents according to similarities scores was also faster due to only 1000 documents being present.

5. 10 Results from query 3 and 20

a. Query 3

```
3 Q0 AP880419-0133 1 0.6601 BM25
3 Q0 AP880920-0017 2 0.619 BM25
3 Q0 AP880518-0053 3 0.6029 BM25
3 Q0 AP880523-0108 4 0.5272 BM25
3 Q0 AP880609-0027 5 0.5228 BM25
```

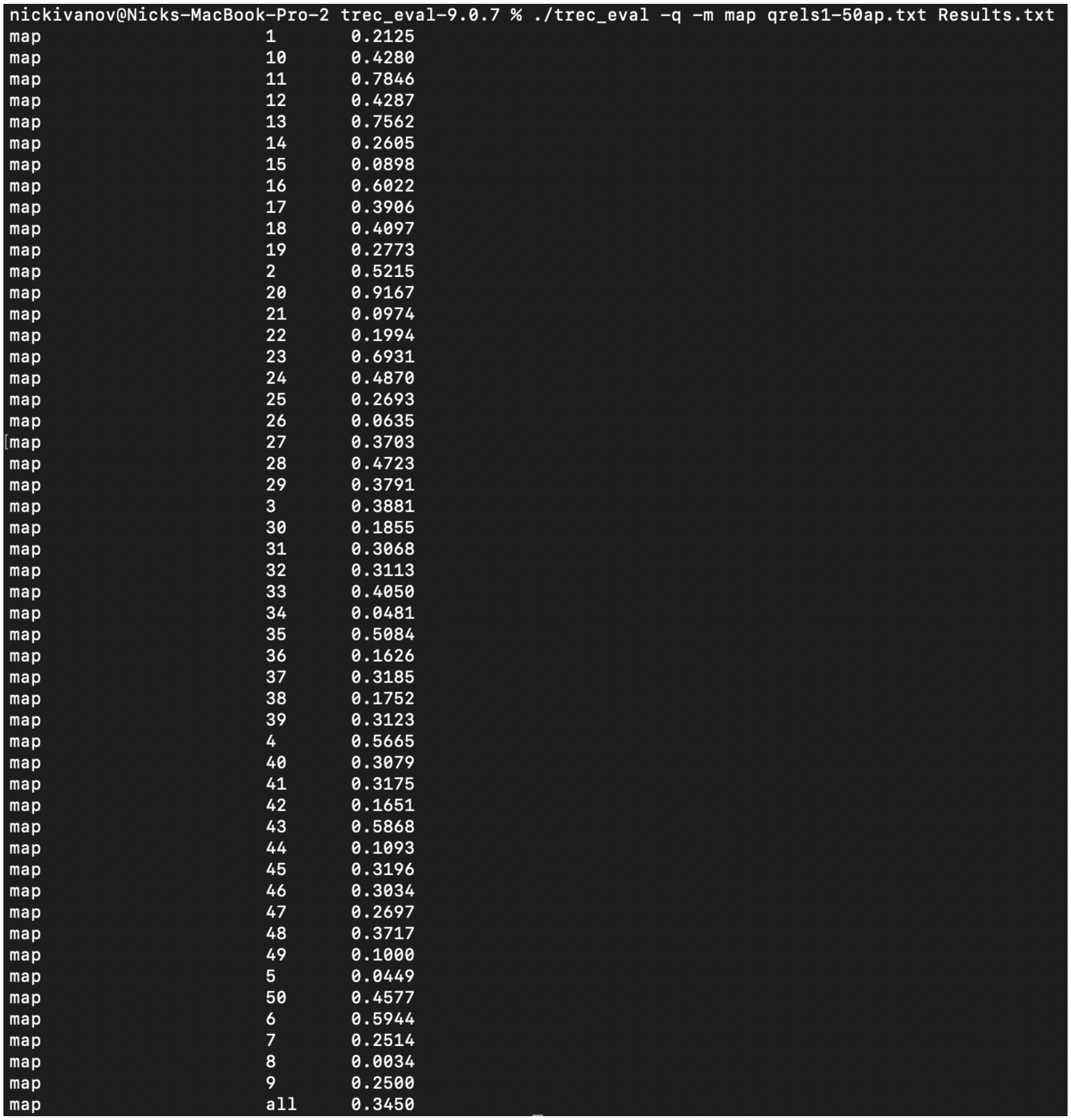
```
3 Q0 AP880628-0074 6 0.5206 BM25
3 Q0 AP880609-0025 7 0.5189 BM25
3 Q0 AP880701-0160 8 0.5186 BM25
3 Q0 AP880511-0043 9 0.5168 BM25
3 Q0 AP880829-0244 10 0.5087 BM25
```

b. Query 20

```
20 Q0 AP881111-0092 1 0.7423 BM25
20 Q0 AP881110-0035 2 0.7239 BM25
20 Q0 AP881122-0171 3 0.7235 BM25
20 Q0 AP881123-0037 4 0.6942 BM25
20 Q0 AP880627-0239 5 0.6887 BM25
20 Q0 AP880323-0094 6 0.5632 BM25
20 Q0 AP880314-0099 7 0.5528 BM25
20 Q0 AP881010-0289 8 0.5478 BM25
20 Q0 AP881103-0013 9 0.5416 BM25
20 Q0 AP880314-0323 10 0.537 BM25
```

Note: These results are straight from the Results.txt provided. BM25 and Q0 are just placeholder names and do not matter.

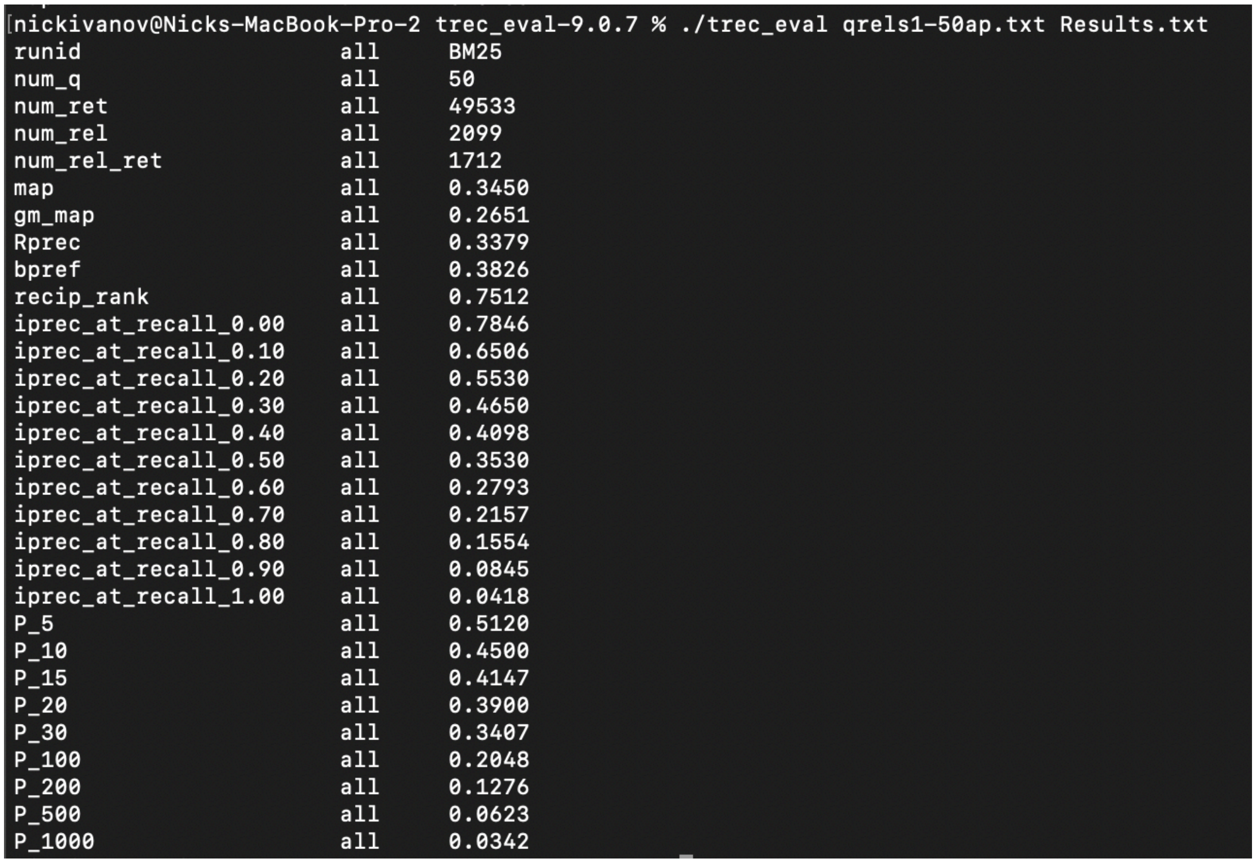
6. **MAP and P@10**
   a. MAP Score



```
nickivanov@Nicks-MacBook-Pro-2 trec_eval-9.0.7 % ./trec_eval -q -m map qrels1-50ap.txt Results.txt
map                     1       0.2125
map                     10      0.4280
map                     11      0.7846
map                     12      0.4287
map                     13      0.7562
map                     14      0.2605
map                     15      0.0898
map                     16      0.6022
map                     17      0.3906
map                     18      0.4097
map                     19      0.2773
map                     2       0.5215
map                     20      0.9167
map                     21      0.0974
map                     22      0.1994
map                     23      0.6931
map                     24      0.4870
map                     25      0.2693
map                     26      0.0635
[map                    27      0.3703
map                     28      0.4723
map                     29      0.3791
map                     3       0.3881
map                     30      0.1855
map                     31      0.3068
map                     32      0.3113
map                     33      0.4050
map                     34      0.0481
map                     35      0.5084
map                     36      0.1626
map                     37      0.3185
map                     38      0.1752
map                     39      0.3123
map                     4       0.5665
map                     40      0.3079
map                     41      0.3175
map                     42      0.1651
map                     43      0.5868
map                     44      0.1093
map                     45      0.3196
map                     46      0.3034
map                     47      0.2697
map                     48      0.3717
map                     49      0.1000
map                     5       0.0449
map                     50      0.4577
map                     6       0.5944
map                     7       0.2514
map                     8       0.0034
map                     9       0.2500
map                     all     0.3450
```

As seen from the screenshot at the very last line, the MAP score is 0.3450

b. P@10



```
[nickivanov@Nicks-MacBook-Pro-2 trec_eval-9.0.7 % ./trec_eval qrels1-50ap.txt Results.txt
 runid                  all     BM25
 num_q                  all     50
 num_ret                all     49533
 num_rel                all     2099
 num_rel_ret            all     1712
 map                    all     0.3450
 gm_map                 all     0.2651
 Rprec                  all     0.3379
 bpref                  all     0.3826
 recip_rank             all     0.7512
 iprec_at_recall_0.00   all     0.7846
 iprec_at_recall_0.10   all     0.6506
 iprec_at_recall_0.20   all     0.5530
 iprec_at_recall_0.30   all     0.4650
 iprec_at_recall_0.40   all     0.4098
 iprec_at_recall_0.50   all     0.3530
 iprec_at_recall_0.60   all     0.2793
 iprec_at_recall_0.70   all     0.2157
 iprec_at_recall_0.80   all     0.1554
 iprec_at_recall_0.90   all     0.0845
 iprec_at_recall_1.00   all     0.0418
 P_5                    all     0.5120
 P_10                   all     0.4500
 P_15                   all     0.4147
 P_20                   all     0.3900
 P_30                   all     0.3407
 P_100                  all     0.2048
 P_200                  all     0.1276
 P_500                  all     0.0623
 P_1000                 all     0.0342
```

As seen from the screenshot, the corresponding P@10 is 0.4500

Note: these scores are based of the GIST Large Embeddings V0 model as this model gave the best results.

7. Division of Tasks:

Kevin: Preprocessing, Evaluation, Implementation of MxBai model, Report
Nick: Preprocessing, Retrieval, Evaluation, Implementation of GIST model, Report
Bradley: Evaluation, Retrieval, Experimental evaluation of other models, Report