Neural Information Retrieval 2023 Project

This project[1] is composed of six parts. Each part corresponds to the material taught in the course lectures. All parts of this project are compulsory. The project will be graded as a whole. The project should be completed individually.

Midterm submission: You need to submit the first part of the report (only written report, no code) **on Absalon by Tuesday, the 16th of May by 23:59. The format of the report should be a PDF document, no more than 4 pages (not including references, if needed).**

Final submission: You need to submit the complete report (written report, revisions, code, read-me file) **on Digital Exam by Friday, the 9th of June by 23:59.** The final submission includes the following:

- PDF document with the report including revisions, **no more than 8 pages (not including references, if needed). Revisions should be marked with a different text color.**
- .zip file containing the code to run your experiments and documentation (readme file) on how to run it.

For the report, both midterm and final submission, you should **use the ACL template**[2]. Note that, at the end of this course, you will present your work to the course instructors using slides. These oral presentations will be held physically at DIKU.

1 Dataset & Indexing (week 17)

1.1 Dataset

You will use the MSMARCO document rank dataset[3]. MS MARCO (MicroSoft MAchine Reading Comprehension) is a large-scale dataset focused on machine reading comprehension. For each document, you have access to several fields, such as url, title, body for instance. In addition to documents, the dataset also has queries (also referred to as topics) and relevance assessments, which indicate which documents are relevant to each query. You will be given access to 367,013 queries and their corresponding relevance assessments for training purposes. In

[1] Note that small modifications in the project description may be made throughout the course.

[2] https://2021.aclweb.org/downloads/acl-ijcnlp2021-templates.zip

[3] `https://msmarco.blob.core.windows.net/msmarcoranking/msmarco-docs.trec.gz`

the later stages of the project, you will be provided with 5,000 new, unseen queries, which you will use to test your IR system (see Section 6).

Given that the computational requirements might be prohibitive to work with the full 367,013 training queries, you will also be given a sample of 5,000 queries out of the 367,013. **You are welcome to use either the full training queries or the 5,000 sampled queries for training.**

Relevance assessments are on a two-point scale: relevant (1), or not relevant (0). On Absalon, you can find the relevance assessments (qrels) of the training queries[4]. You will not have access to the relevance assessments of the unseen test queries.

As an initial step for the project, please present (in a table and/or figure) the following:

1. The total number of documents and queries in the version of the dataset you have been given.
2. The min, max, median and average length of the queries (in terms of number of words or characters).
3. The min, max, median and average length of the documents (in terms of number of words or characters).
4. The distribution of document lengths (e.g. histogram or box plot).
5. The distribution of query lengths (e.g. histogram or box plot).

Briefly discuss these statistics. Do you see anything unusual?

You should also download and familiarise yourself with one of the following IR systems libraries: Terrier[5], Indri[6], Elasticsearch[7], Anserini[8], etc.

1.2 Indexing

The objective of this section is to help you become acquainted with the dataset and construct an index that will serve as the foundation for your retrieval algorithms. The following tasks should be completed:

- Load and index the dataset. We suggest that you use existing tools and libraries (e.g., Terrier, Indri, Elasticsearch, Anserini, etc.) to build your index. Python wrappers for different libraries are also available (e.g., PyTerrier[9], Pyserini[10], Elasticsearch-py[11], etc.).
- You should build four different versions of your index: (1) full index, (2) stopwords removed, (3) stemming, (4) stopwords removed + stemming.

[4] `https://absalon.instructure.com/courses/64839/files/folder/project/train`
[5] `http://terrier.org/`
[6] `http://www.lemurproject.org/indri.php`
[7] `https://www.elastic.co/`
[8] `https://github.com/castorini/anserini`
[9] `https://pyterrier.readthedocs.io/en/latest/`
[10] `https://github.com/castorini/pyserini`
[11] `https://github.com/elastic/elasticsearch-py`

- After building each version of your index, report in a table, for each index version, the number of documents indexed, number of unique terms, total number of terms, index storage size, how much time it took to build each version of your index, and how much time (average searching time) it takes to process a query with each version of your index.

In your report, you need to include all the important details regarding the implementation of your index, including any preprocessing of the data. What are the advantages and disadvantages of your indexing approaches? What are the most and least efficient (indexing time or searching time) indices?

2 Ranking Models and Evaluation (week 18, 19)

Once you have the index, you can start implementing your ranking models. You should do the following: Tune and run BM25 and a Language Model (LM). You should do this for each of the four versions of the index separately. You are allowed to use existing search libraries. The same library that you have used to build your index will likely have a version of BM25 and LM.

To tune your ranking models, you should split your training queries into several folds and use cross-validation to tune your models. For example, if you split the training queries into 5 folds[12], you train your model using four folds and test on the remaining fold (test fold), and you repeat by alternating the folds until all folds have been used once as a test fold. You then compute the average performance of all the test folds to find the best configuration. This is the final configuration of your model (tuned model) that you should use on the unseen queries that we will release later on.

You should tune with respect to one evaluation measure only (either MRR or NDCG@10). In addition, you should present in a table, for each ranking model, and for each version of the index, NDCG, MRR, Precision and Recall at 5, 10 and 20 cutoffs. We require that you use `trec-eval`[13] for evaluation.

You are also expected to report your methods' mean response time (the time of getting the evaluation score of a query).

In your report, you need to include all the important details regarding the implementation of your models, for example the library you used, how you split the training queries, how you tuned the parameters of your model(s), etc. The goal of this section is to discuss and compare different retrieval approaches and the limitations of different evaluation measures. What limitations do you identify in different evaluation measures? Which combination of index and ranking model is the most effective? Which one is the least effective? What affects effectiveness more, the index or the ranking model? Different models will perform differently also depending on the evaluation measure. It is up to you to decide whether to report the results in tables, plots, box-plots, etc. Remember to give insights on which models and evaluation measures work the best and why, highlighting your design decisions.

Non-compulsory element: You are not limited to the above ranking models and evaluation measures; you can use further ranking models and evaluation measures and explore different cut-offs.

[12] Five folds are recommended.

[13] `https://github.com/usnistgov/trec_eval`

3 Mid-term submission (week 20)

You need to submit a report including all the compulsory components of the project that have been requested up to this point.

You need to submit the mid-term report no later than **May 16 2023 at 23:59**. The format of the report should be a PDF document using the ACL template[14], no more than **4 pages** (not including references).

4 Relevance feedback (week 21)

You should use pseudo relevance feedback to expand the query and run it with BM25 and LM. You should tune the parameters of both pseudo relevance feedback and the ranking model using x-fold validation using an evaluation measure of your choice. Five folds are recommended.

In addition, you should use word embeddings to do query expansion as done by Kuzi et al.[15][4]. A popular word embedding choice is the word2vec pretrained on Google News corpus,[16] but you can decide on another embedding. Another choice of embeddings is using contextualized word embeddings, such as BERT [3].

Please report the results of the above experiments with respect to NDCG, MRR, Precision and Recall at 5, 10 and 20 cutoffs. Also report your methods' mean response time (the time of getting the evaluation score of a query). For pseudo relevance feedback, plot (a) the number of expansion terms versus an evaluation measure of your choice, and (b) the number of expansion documents versus an evaluation measure of your choice. Then discuss how varying those two parameters (number of expansion terms and number of expansion documents) affects retrieval performance.

5 Re-ranking (week 22)

You should use the best (tuned) version of BM25 and LM from the mid-term report to generate an initial ranking (separately for BM25 and for LM), and then re-rank the top K[17] documents using contextualized embedding approaches such as BERT [2], or Sentence-BERT [5]. To do so, you have to build the query and document representations using these approaches and then compute the similarity score between them. The similarity score between each document and query can be used to re-rank the documents. You can also get some inspiration from [1]. Does re-ranking through embedding approaches lead to better retrieval?

If the authors of [2, 5] have trained a new method or model and they have released the trained model, you do not need to train it too; you can simply use it and cite them (your citation should include the url where the released model can be found).

Please present your results using the exact format of the results tables you used in Section 2.

[14] `https://github.com/acl-org/acl-style-files`
[15] `https://dl.acm.org/doi/pdf/10.1145/2983323.2983876`
[16] `https://github.com/mmihaltz/word2vec-GoogleNews-vectors`
[17] Typically, $K \in \{100, 500, 1000\}$.

In your report, you need to include all the important details regarding the implementation of your embedding models, as for example the library you used, how you tuned the parameters of your models and any external resource you used. The goal of this section is to discuss and compare your models with those implemented in Section 2. Is your reranking model performing better or worse than the simpler models in Section 2? Why or why not? Are your results aligned with those reported in the literature? How can you improve your models? Are there differences among queries? Are there easier or more difficult queries? Are the easy and difficult queries the same you found in Section 2?

6 Competition - unseen queries (weeks 21, 22)

To test the effectiveness of your models in a more realistic setting, we will evaluate your models on a test set of unseen queries. We will release the unseen queries on Absalon by May 26 at 09h00. You should choose your best configuration of (a) index, (b) ranking model, (c) relevance feedback or not, and (d) reranking or not, and run this configuration on the unseen queries. You should can submit at least three and at most five runs of different configurations on the unseen queries.

We ask you to submit your run files in trec-eval format, each of them in a separate .txt file, and all of them uploaded as one .zip file on Absalon by **June 1 at 16h00** at the latest. We ask you to choose a name for each run. The name should be the filename of the .txt file that contains that run. Each run should have a different name. Run names should be in the following format: {any 3 small letters of your choice}{any three numbers between 0-9 of your choice}{any three capital letters of your choice}{any three numbers between 0-9 of your choice}. For example, the following are valid run names:

- dog357IJS302
- lod547ASE667
- qok312POW987
- aqw769GHH095

The following are examples of invalid run names:

- dg357IJS302
- lod547.ASE
- 312POW987qok
- aqw7650GHH095

We will then evaluate all the runs we receive and we will release the results on Absalon by Friday 2 June at 17h00 at the latest. You should make sure that you remember the name of your own runs, so that you can look at their performance and **include a discussion of this in your report**. The rest of this section provides more details on this.

6.1 Format

The submission format of your runs must follow the standard TREC run format. The submission format of a ranked result list (run) is as follows:

```
qid Q0 docno rank score tag
```

The fields should be separated with a white space. The width of the columns in the format is not important, but it is important to include all columns and have some amount of white space between the columns. The above fields are:

- `qid`: the query number;
- `Q0`: unused and should always be Q0;
- `docno`: the official document id number returned by your system for the query `qid`.
- `rank`: the rank where the document is retrieved;
- `score`: the score (integer or floating point) that generated the ranking. The score must be in descending (non-increasing) order. The score is important to handle tied scores (`trec_eval` sorts documents by their scores values and not their ranks values);
- `tag`: your run name, in the format described above. **Each run should have a different tag**.

An example of a run is shown below:

```
1 Q0 doc1 1 14.8928003311 dgp357IJS302
1 Q0 doc2 2 14.7590999603 dgp357IJS302
1 Q0 doc3 3 14.5707998276 dgp357IJS302
1 Q0 doc4 4 14.5642995834 dgp357IJS302
1 Q0 doc5 5 14.3723001481 dgp357IJS302
...
```

The submitted runs should contain at most the top 1,000 documents for each query, with all the queries in each file. The submission file must be in `.zip`. The queries for the test set will be released on Absalon.

6.2 Evaluation of Runs

We will evaluate the submitted runs with the following measures: NDCG, MRR, Precision and Recall at 5, 10 and 20 cutoffs. We will post the results of the evaluation on Absalon by 2 June at 17h00. If a run is not in the correct format and cannot be processed by trec-eval, we will ignore it.

Note that we will release the qrels only for the training queries, thus, you will not have the qrels for the test queries.

7 Requirements of the written report that describes your project

The following considerations are applicable to all sections of this project. You need to keep them in mind when you write your report.

The final report should include everything that was included in the mid-term report, plus everything requested from Section 4 up to here. **You should also include tables with the evaluation measure scores (that we will release) of your runs on the unseen queries, as well as the mean response time (the time of getting the evaluation score of a query) for the runs on the unseen queries.** You should reflect on the differences in effectiveness between the seen and unseen queries. Find the 5 worst performing seen queries (according to MRR), and present their text and their respective MRR in a table. Then reflect on why you think these 5 queries have the worst performance. Do the same for the 5 best performing seen queries.

In all cases, you should describe *what* you tried, what worked, what did not work, and *why* you think it did or did not work. For example, did you use an off-the-shelf method? How did you adapt it for the given task? Why did this adaptation work or not? Did you do some preprocessing or cleaning of the dataset? Was is it useful? Why or why not?

It is not enough to report the evaluation scores of your methods. You need to explain the reasons why you think we see these scores. You need to show in the report that you have tried to understand why you got these specific evaluation scores, regardless of whether they are high or low.

You should also describe the limitations of what you did. What could have led to improvements in model performance? How could you have approached this task differently? What was particularly challenging about working with this dataset and why do you think that was? This discussion does not require further experiments, but requires you to examine your experimental results, critically think about your choices and the assumptions you made, hypothesise on how you can overcome some limitations and improve your solution. Your discussion should be based on evidence, such as lecture material and relevant literature.

Finally, in all cases, you should cite relevant literature which informed your choices in terms of modeling, analysis, etc.

Academic Code of Conduct

Discussion with fellow students regarding the project is allowed, but sharing of code or written content is strictly prohibited. Any instances of directly copying code or text from other students will be considered as plagiarism and will be subject to the University's plagiarism regulations.

Extra work

The above is a description of the minimum that we expect you to implement, but you are allowed to expand upon this to explore more complex ideas. If you decide to submit extra work, that extra work will be graded as well.

References

[1] Z. Dai and J. Callan. Context-aware Sentence/Passage Term Importance Estimation for First Stage Retrieval. *arXiv preprint arXiv:1910.10687*, 2019. `https://arxiv.org/abs/1910.10687`.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[4] S. Kuzi, A. Shtok, and O. Kurland. Query Expansion Using Word Embeddings. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, page 1929–1932, New York, NY, USA, 2016. Association for Computing Machinery. `http://publish.illinois.edu/saar-kuzi/files/2017/10/w2v_cikm16.pdf`.

[5] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.