Universidade Federal de Minas Gerais
Departamento de Ciência da Computação

TCC/TSI/TECC: Information Retrieval

Research Challenge
Entity Search

Deadline: Jun 23, 2025 23:59 UTC-3 via Moodle

Overview This challenge aims to assess the student's skills as a search engineer based upon the knowledge acquired during the course. In particular, the challenge focuses on the problem of entity search in a knowledge base. Given a query q and an entity corpus D, the goal is to produce a ranking $R_q \subseteq D$, with $|R_q| \leq k$, sorted in decreasing order of relevance with respect to q.

Kaggle This assignment uses Kaggle Community as a platform for automatically evaluating the effectiveness of your produced rankings. You can register to join the Kaggle competition by clicking on the following link:

`https://www.kaggle.com/t/5cff49e66f9d4e59a2c1f8e90db6d2ac`

Input Your implementation must take the following files as input:

- `corpus.jsonl`, containing structured representations (with title, keywords, and descriptive text) for a total of 4,641,784 entities
- `test_queries.csv`, containing 233 natural language queries (e.g. `electricity source in france`), each preceded by an identifier (e.g. `014`)

To improve your ranking performance, you may choose to experiment with supervised learning approaches (e.g. learning to rank, neural ranking models). To this end, the following annotated data will be provided as additional input:

- `train_queries.csv`, containing 234 natural language queries (e.g. `tango music composers`), each preceded by an identifier (e.g., `035`)
- `train_qrels.csv`, containing 8,202 relevance judgments for entities associated with the queries in `train_queries.csv`; each relevance judgment is expressed in a 2-degree scale (1: partially relevant; 2: highly relevant)

Note that queries and relevance judgment files contain a header line. All these files can be downloaded from the data description page on Kaggle.[1]

Output For each ⟨`QueryId`, `Query`⟩ pair in the `test_queries.csv` file, you must produce a ranking of entities by searching the index produced for the documents in the provided `corpus.jsonl` file. The rankings output by your implementation must be stored in a submission (csv) file to be uploaded to Kaggle. In total, a submission file must contain a header line plus a maximum of 233×100 lines (i.e. 233 queries times at most 100 entities per query, totaling 23,301 lines maximum). An example submission file is provided below:

```
QueryId,EntityId
002,0878002
002,3056323
002,3056336
...
465,0270254
465,3989010
465,3411664
```

Your submission should be uploaded to Kaggle[2] to be automatically evaluated. Through the course of this challenge, you should try different indexing and query processing strategies, as exemplified below, in order to improve the quality of the rankings produced. To this end, you can upload a maximum of 20 submissions per day to Kaggle. The platform will maintain a live leaderboard indicating the relative performance of your submissions in comparison to those made by your fellow classmates. Keep track of the performance of your submissions, so you can analyze what worked in your final report.

Implementation You are free to use your favorite programming language for this assignment, including any high-level search library. Example libraries with Python APIs that include extensive support for advanced indexing and ranking strategies are Pyserini[3] and PyTerrier.[4]

Indexing strategies As part of this challenge, you should try different strategies for indexing the provided entity corpus. Example strategies include:

- Different tokenizers, stemmers, stopword removers
- Different index organizations (frequency-based, position-based, field-based)
- Different encoders (sparse, dense)

[1] `https://www.kaggle.com/c/ir-20251-rc/data`
[2] `https://www.kaggle.com/c/ir-20251-rc/submissions`
[3] `https://github.com/castorini/pyserini`
[4] `https://github.com/terrier-org/pyterrier`

Query processing strategies In addition to indexing strategies, you should also try different query processing strategies. Example strategies include:

- Different tokenizers, stemmers, stopword removers (aligned with indexing)
- Different ranking models, including unsupervised and supervised ones

Deliverables Before the deadline (Jun 23, 2025 23:59 UTC-3), you must submit a package file (`zip`) via Moodle containing the following:

1. Report (pdf) describing your 5 most effective submissions uploaded to Kaggle through the course of this challenge. Each submission must be described in a separate section, clearly indicating:
   (a) the identification of the submission (as displayed in Kaggle)
   (b) the submission effectiveness (as reported by Kaggle)
   (c) the hypothesis investigated by the submission
   (d) what indexing and query processing strategies were used for the submission and what motivated you to try these particular strategies (you can argue based on theoretical or empirical aspects seen in class)
2. The `csv` files corresponding to the 5 submissions
3. The source code needed to generate each submission

Evaluation criteria Your challenge participation will be assessed based on:

(50%) Report (creativity of the proposed hypotheses, clarity, correctness, and depth of discussions; maximum 5 pages, one per submission)[5]

(50%) Effectiveness of your best submission uploaded to Kaggle, measured in terms of nDCG@100,[6] relative to your fellow contestants

Late submissions Late submissions will be penalized in $2^{(d-1)} - 0.5$ points, where $d > 0$ is the number of days late. In practice, a submission 5 or more days late will result in a zero grade.

Teams This assignment must be performed either individually or in teams of at most 3 students. In all cases, students must register individually on Kaggle. Teams must be formed on the platform itself anytime before Jun 16, 2025 23:59 UTC-3. Any sign of plagiarism will be investigated and reported to the appropriate authorities.

[5] Your report should use the ACM `sample-sigconf.tex` template available from the link: `https://portalparts.acm.org/hippo/latex_templates/acmart-primary.zip`

[6] `https://en.wikipedia.org/wiki/Discounted_cumulative_gain`