IRDM Course Project Part II

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

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

1 Task Definition

An information retrieval model is an essential component for many applications (e.g. search, question answering, recommendation etc.). Similar to the first part of the project, your task in this assignment 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. In this part of the assignment, your goal is to improve the basic models that you implemented in the first part.

This is an individual project, therefore, everyone is expected to submit their own project report.

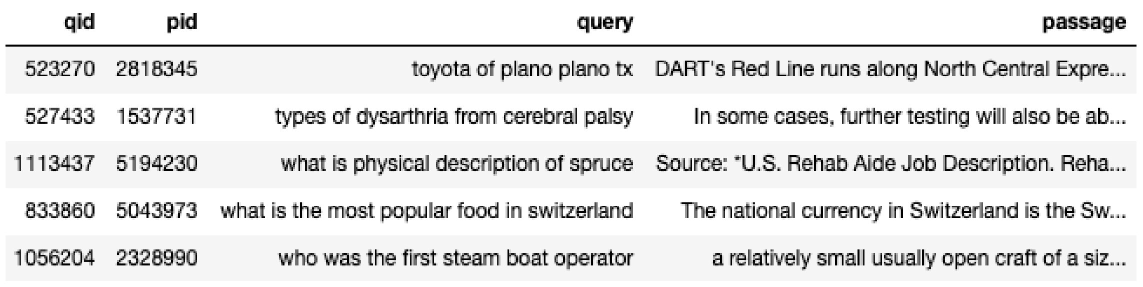
2 Data

The dataset from previous task is available through this url and the dataset for training and validation is available through this url. Our dataset consists of 5 files:

- *test-queries.tsv* is a **tab** separated file, where each row contains a query ID (qid) and the query (i.e., query text).
- *candidate_passages_top1000.tsv* is a **tab** separated file, containing initial rankings that contain 1000 passages for each of the given queries (as it was in the first part of the assignment) 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.
- *train_data.tsv* and *validation_data.tsv*. These are the datasets you will be using for training and validation. You are expected to train your model on the training set and evaluate your models' performance on the validation set. In these datasets, you are given additional **relevance** column indicating the relevance of the passage to the query which you will need during training and validation.

3 Subtasks

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



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

1. Evaluating Retrieval Quality. (20 marks) Implement methods to compute the average precision and NDCG metrics. Compute the performance of using BM25 as the retrieval model using these metrics. Your marks for this part will mainly depend on the implementation of metrics (as opposed to your implementation of BM25, since we already focused on that as part of the first assignment).

2. Logistic Regression (LR). (25 marks) Represent passages and query based on a word embedding method, (such as Word2Vec, GloVe, FastText, or ELMo). Compute query (/passage) embeddings by averaging embeddings of all the words in that query (/passage). With these query and passage embeddings as input, implement a logistic regression model to assess relevance of a passage to a given query. Describe how you perform input processing & representation or features used. Using the metrics you have implemented in the previous part, report the performance of your model based on the validation data. Analyze the effect of the learning rate on the model training loss. *All implementations for logistic regression algorithm must be your own for this part.*

   Important Notes:

   - The training data size you are given is quite small, so it should not cause you much difficulty in training but in case you have any issues with the data size, please feel free to use a sample of the training data.
   - If you think it is necessary, you are allowed to use negative sampling for generating a subset of training data (possibly together with other sampling methods if needed).

3. LambdaMART Model (LM). (25 marks) Use the `LambdaMART` [1] learning to rank algorithm (a variant of LambdaRank we have learned in the class) from `XGBoost` gradient boosting library [1] to learn a model that can re-rank passages. You can command XGBoost to use LambdaMART algorithm for ranking by setting the appropriate value to the `objective` parameter as described in the documentation [2]. You are expected to carry out hyper-parameter tuning in this task and describe the methodology used in

[1] https://xgboost.readthedocs.io/en/latest/index.html
[2] https://xgboost.readthedocs.io/en/latest/parameter.html#learning-task-parameters

deriving the best performing model. Using the metrics you have implemented in the first part, report the performance of your model on the validation data. Describe how you perform input processing & representation or features used.

4. Neural Network Model (NN). (30 marks) Using the same training data representation from the previous question, build a neural network based model that can re-rank passages. You may use existing packages, namely Tensorflow or PyTorch in this subtask. You are expected to justify your neural network architecture by providing the motivation and how it fits to our problem. You are allowed to use different types of (deep) neural network architectures (e.g. feed forward, convolutional, recurrent and/or transformer based neural networks) for this part. Using the metrics you have implemented in the first part, report the performance of your model on the validation data. Describe how you perform input processing & representation or features used.

3.1 Submission of Test Results.

You should have one file per model (named LR.txt, LM.txt, and NN.txt, respectively), where the format of the file is:

```
<qid1 A1 pid1 rank1 score1 algoname2>
<qid1 A1 pid2 rank2 score2 algoname2>
<qid1 A1 pid3 rank3 score3 algoname2>
<qid1 A1 pid4 rank4 score4 algoname2>
...
```

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 LR, LM or NN, depending on which model you used) .

4 Submission

You are expected to submit all the codes you have implemented for all the parts of the assignment (e.g. evaluation metrics, data representation, logistic regression, LambdaMART training, neural network implementation, etc.) All the code should be your own and you are not allowed to reuse any code that is available online. You are allowed to use both Python and Java as the programming language.

You are also expected to submit a written report whose size should not exceed 6 pages, including references. Your report should describe the work you have done for each of the aforementioned steps. Your report should explicitly describe the performance of the models you have implemented, the input representations (or features) you have used, how you have used the training and validation sets (any sub-sampling done, etc.), how you have done hyper-parameter tuning, the neural architecture you have used and why, etc.

You are required to use the SIGIR 2020 style template for your report. You can either use LaTeX or Word available from the ACM Website [3] (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 will be announced later.

References

[1] C. J. Burges. From ranknet to lambdarank to lambdamart: An overview. Technical Report MSR-TR-2010-82, June 2010.

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