Elsevier at SimpleText: Passage Retrieval by Fine-tuning GPL on Scientific Documents

Artemis Capari, Hosein Azarbonyad, Georgios Tsatsaronis and Zubair Afzal

Elsevier, Amsterdam

Abstract

CLEF SimpleText Lab is centered around finding relevant passages from a large collection of scientific documents in response to a lay query, detecting and explaining difficult terminology within those passages, and finally simplifying the passages. The first task is similar to the ad-hoc retrieval task in which given a topic/query, the goal is to retrieve relevant passages, but in addition to the relevance, ranking models should assess documents based on their readability/complexity as well. This paper describes our approach towards building a ranking model to tackle the first task. To build the ranking model, we first evaluate performance of several models on a proprietary test collection constructed based on scientific documents across multiple science domains. Then, we fine-tune the best performing model on a large collection of unlabelled documents using the Generative Pseudo Labeling approach. The key contribution and findings of our approach is that a bi-encoder model, trained on the MS-Marco dataset, fine-tuned further on a large collection of unlabelled scientific passages achieves the highest performance on the proprietary dataset which is specifically designed for the scientific passage retrieval task. Finally, fine-tuning a model in the same fashion, but only using the Computer Science queries from the test collection has proven to be successful for SimpleText Task 1.

Keywords

Information Retrieval, Scientific Documents, Domain Adaptation, Scholarly Document Processing

1. Introduction

Scientists and researchers employ specialized language and ideas to effectively communicate information. Consequently, there exists a substantial, increasing volume of scientific concepts and information within any given scientific field, which contributes to the challenges scientists face in keeping pace with the expanding scope of technical concepts and novel content. Understanding scientific documents is even more challenging for the public audience. It has been shown that the readability of scientific documents is decreasing over time [1]. This poses challenges and opportunities towards both researchers and publishers to think about way to increase the readability of complex scientific documents for public audience.

SimpleText Lab [2] is specifically focused around addressing these challenges. The aim of this lab is to first find relevant passages to users' queries, spot and explain difficult terminology within relevant passages, and finally simplify the passage by re-writing it in a more readable way. The very first task in the series of tasks associated with this lab, is a passage retrieval task namely "What is in (out)", where the goal is, given a query/topic, to retrieve all passages relevant to the query/topic that can be used to create a simplified summary around the topic.

CLEF 2023: Conference and Labs of the Evaluation Forum, September 18–21, 2023, Thessaloniki, Greece

CEUR Workshop Proceedings (CEUR-WS.org)

In addition to the relevance, ranking models should also consider the complexity of passages when ranking them and prioritize less complex passages.

The state-of-the-art ranking models are semantic matching models using either a cross-encoder or bi-encoder (or a combination of) architectures [3]. These models are trained on publicly available datasets such as MS-Marco [4] which do not contain scientific documents. The retrieval task of the SimpleText lab itself and the underlying training/evaluation sets are centred around scientific documents. Therefore, existing ranking models might not perform very well in this setting as the language of scientific documents is usually more complex and there might be specific scientific terminology within scientific documents that is specific for such documents.

In this paper we build our model on top of the existing state-of-the-art ranking models. To address the domain difference challenge, we use a domain adaptation technique, namely *Generative Pseudo Labeling* (*GPL*) to fine-tune the pre-trained models on a set of unlabelled scientific documents. To evaluate ranking models and fine-tune them, we build a proprietary test collection containing 5000 query document-pairs annotated by relevance labels. Our results on this dataset, shows that a bi-encoder model fine-tuned on a large collection of scientific unlabelled documents achieves a stronger performance than the zero-shot counterpart. We use this model to re-rank documents ranked by the Elastic Search system. Our results show that some of the fine-tuned models achieve a better performance than the zero-shot models on the SimpleText dataset as well. In the remainder of the paper, we briefly review related work in Section 2, we describe the technical details of the designed system in Section 3, we empirically evaluate the models in Sections 4 and 5 and we conclude in Section 6 by arraying some limitations of the current technical solution and provide pointers to future work.

2. Related Work

Dense retrieval models are a type of information retrieval (IR) model that use fixed-length dense vector representations to represent both queries and documents, allowing for efficient and accurate retrieval of relevant information from a large corpus of text by computing the similarity score between query- and document vectors. These models have been shown to outperform traditional sparse retrieval models, such as BM25 [5], in a variety of tasks, including open-domain question answering and document ranking.

Two popular types of such dense retrieval models are bi-encoders and cross-encoders. Both models still have the same objective, i.e. capturing the semantic meaning of queries and documents into dense vector representations, but differ in the architecture of the neural network used to learn their representations.

Bi-encoders use two separate encoders to independently encode the query and the document into dense vectors, which are then compared using a similarity function to produce a relevance score. One of the most popular bi-encoders is the *Dense Passage Retrieval* (*DPR*) model [6]. *DPR* uses a two-stage retrieval process, in which a large set of passages is first retrieved using sparse techniques, which is used in turn to compute a dense vector representation of each passage using a pre-trained language model such as *BERT* [7]. The query is represented using a similar dense vector representation as well. The passages are then ranked based on the cosine similarity

between the query and passage vectors.

Cross-Encoders however, use a single encoder to encode the query and document into a joint embedding space. Documents are then ranked based on the similarity score that is computed between this joint embedding and the learned representation of the positive document. They can capture more complex interactions between query and document. However, they are computationally more expensive as it requires a unique embedding for each query-document pair, while bi-encoders encode queries and documents separately and therefore it only requires a single document corpus for all queries [8]. Therefore, they are often only used as re-rankers [9, 10, 11, 12, 13, 14].

3. Methodology

To train and fine-tune our models, we first build a test collection using a set of scientific documents. Then, we fine-tune existing ranking models using this dataset as well as a large collection of scientific documents to make these model more suitable for retrieving scientific passages.

3.1. Test Collection

To build a test collection, we select 100 queries spread across 20 different scientific domains[1]. We select the queries to be a known scientific concept on which we can collect credible and relevant documents/passages. Once the queries are selected, we then use the well-known pooling mechanism to retrieve candidate documents to be annotated per query. We select five different models (two lexical matching, two bi-encoders, and one cross-encoder) as the models to be used to build the pool. These models are selected based on their performance on a small set or to ensure the diversity of models (and hence diversity of document within the pool). We select 50 documents per query using the pooling approach. These documents are then labeled by experts per domains as "relevant", "partially relevant", or "non-relevant". We use this dataset to evaluate the performance of different ranking models.

3.2. GPL

Generative Pseudo Labeling (GPL) is an unsupervised domain adaptation method first introduced in [15]. The proposed framework leverages the structure of a pre-trained generative model to generate pseudo labels for the target domain data, which are then used to train a retrieval model in a supervised manner. GPL outperforms existing unsupervised domain adaptation methods on several benchmark datasets and achieves state-of-the-art performance in unsupervised domain adaptation of dense retrieval. Considering we intend to use- and experiment with dense-retrieval models, and the importance of large amounts of data has often been highlighted in previous

[1] Genetics and Molecular Biology, Computer Science, Economics, Agricultural and Biological Sciences, Biochemistry, Econometrics and Finance, Toxicology and Pharmaceutical Science, Chemical Engineering, Veterinary Science and Veterinary Medicine, Chemistry, Materials Science, Earth and Planetary Sciences, Engineering, Food Science, Immunology and Microbiology, Mathematics, Nursing and Health Professions, Medicine and Dentistry, Neuroscience, Pharmacology, Psychology, Physics and Astronomy, Social Science

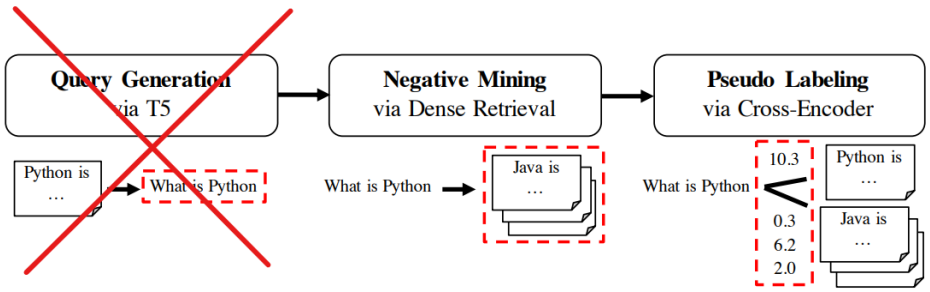

Figure 1: Generative Pseudo Labeling (GPL) for training domain-adapted dense retriever [15]

Table 1
Details on fine-tuning of various models

Model Name	Bi-Encoder	Queries	Documents	Batch Size	Training Steps	Epochs
MS-DB-v4-GPL-CS	msmarco-distilbert-base-v4	218 (10 golden)	23670	16	15000	1
MS-DB-tas-b-GPL-CS	msmarco-distilbert-base-tas-b	218 (10 golden)	23670	16	15000	1
MS-DB-v4-GPL-all	msmarco-distilbert-base-v4	4637 (100 golden)	893110	32	280000	1
MS-DB-tas-b-GPL-all	msmarco-distilbert-base-tas-b	4637 (100 golden)	893110	32	280000	1

work on dense retrieval methods [7, 6, 3], our manually annotated dataset might not suffice as it only consists of 5000 snippets from a set of a 100 queries. However, there are many more snippets and possible queries that can be extracted from a large collection of unlabeled scientific documents (research articles), which could be labeled by GPL on their relevance in order to fine-tune and adapt the existing ranking models to the scientific document retrieval task.

We adapt GPL to our use-case, by first removing the query generation part. Instead, we select a set of known scientific concepts per domain, and then per concept, we find all passages mentioning the concept.

Finding an exact mention of a scientific concept in a document can be a very good indicator of relevance of the document to the concept. Then, per concept, each document mentioning it is regarded as positive, and a bi-encoder is used to find negative document per query.

The GPL framework uses a cross-encoder as a teacher model on the collected positive and negative documents to fine-tune the underlying bi-encoder model, which is used to adapt the bi-encoder model to our scientific document ranking setting. For our use-case, we have fine-tuned two different bi-encoders *msmarco-distilbert-base-v4*[8] (MS-DB-v4) and *msmarco-distilbert-base-tas-b*[16] (MS-DB-tas-b) using our whole test collection, spanning 20 different scientific domains, consisting of 5 queries each. We found that *msmarco-distilbert-base-tas-b* was most suitable for tasks that require understanding of a wide range of domains.

However, as the SimpleText task aims at finding references in Computer Science, we have also fine-tuned the aforementioned models on queries and articles from just the Computer Science and Mathematics domains. Naturally, these models were fine-tuned on far less data (See Table 1).

Each of the models were fitted on pseudo labels created with *ms-marco-MiniLM-L-6-v2*, using the Adam Optimiser [17] with a learning rate of $2e{-}5$ and 1000 warm-up steps.

4. Experiments

We have applied our models in several settings before selecting the final 10 submitted runs. Different variations of the best performing models (on the proprietary test collection) were selected to make the final submissions. As shown in Table 2, the rankings for runs 1-7 were retrieved by taking the top-k documents found for each of the 29 queries from `Simpletext_2023_task1_train.qrels` by the Elastic Search API. These were then re-ranked using our fine-tuned models. The rankings for the first 4 runs were obtained with the model that was only fine-tuned on Computer Science and Mathematics data, while we used the model fine-tuned on all Science Direct Domains for runs 5-7. For run 8, the top-500 documents were retrieved by searching for "query, topic", and then re-ranked using our CS fine-tuned model, again using "query, topic" as the query input. For run 9, we used the model that performed best on our own test collection to search the entire corpus for each query, rather than pre-filtering with Elastic Search. Finally, we used our best CS-trained model once again, but searched per topic instead of per query.

Table 2
Configurations of official submissions

Run	Query Input	Corpus	Model
1	query	ES Top-500	MS-DB-v4-GPL-CS
2	query	ES Top-100	MS-DB-v4-GPL-CS
3	query	ES Top-1000	MS-DB-v4-GPL-CS
4	query	ES Top-5000	MS-DB-v4-GPL-CS
5	query	ES Top-100	MS-DB-tas-b-GPL-all
6	query	ES Top-500	MS-DB-tas-b-GPL-all
7	query	ES Top-1000	MS-DB-tas-b-GPL-all
8	query, topic	ES Top-500	MS-DB-v4-GPL-CS
9	query	Whole corpus	MS-DB-tas-b-GPL-all
10	topic	ES Top-500	MS-DB-v4-GPL-CS

5. Results

We have selected our runs based on our own evaluation, which uses the qrels provided to us. However, to our knowledge, these qrels are biased towards passages retrieved by ElasticSearch, which is a *lexical* search method. Naturally, the recall for our *semantic* search models may therefore be limited. As the test qrels that have been used for the official evaluation are based on pooling the submissions of 2023 participants [2], these qrels include passages from various types of neural rankers as well as lexical matching models. Hence the results from our own evaluation differ from the official results. Nonetheless, they are included as they still provide insight on our training process and our decisions behind selecting certain runs.

5.1. Selecting Best Runs

In this section, we describe the results of fine-tuning different ranking models on a large collection of unlabeled documents using the GPL model.

Table 3
Evaluated per Query: Performance of zero-shot models vs fine-tuned models on `Simpletext_2023_task1_train.qrels`

Model	P@10	R@10	RR@10	nDCG@5	nDCG@10	nDCG@50	nDCG@100
0-shot MS-DB-tas-b	**0.224**	**0.148**	**0.543**	**0.225**	**0.214**	**0.284**	**0.340**
MS-DB-tas-b-GPL-all	0.207	0.132	0.413	0.211	0.209	0.270	0.332
0-shot MS-DB-v4	0.217	0.142	0.452	0.203	0.203	0.263	0.325
MS-DB-v4-GPL-all	0.224	0.137	0.429	0.206	0.206	0.249	0.309

While *ms-marco-distilbert-base-tas-b* proved most suitable for fine-tuning on our use-case, Table 3 shows that it underperforms its zero-shot equivalent on the train set. A possible explanation could be the pooling bias or the shallow depth of the training set. To be able to explain this result and make solid conclusions based on these results, we need to evaluate the performance of these models on an unseen test set. On the other hand, the fine-tuned *ms-marco-distilbert-base-v4* model outperforms the zero-shot version which shows the effectiveness of fine-tuning on the performance of this model.

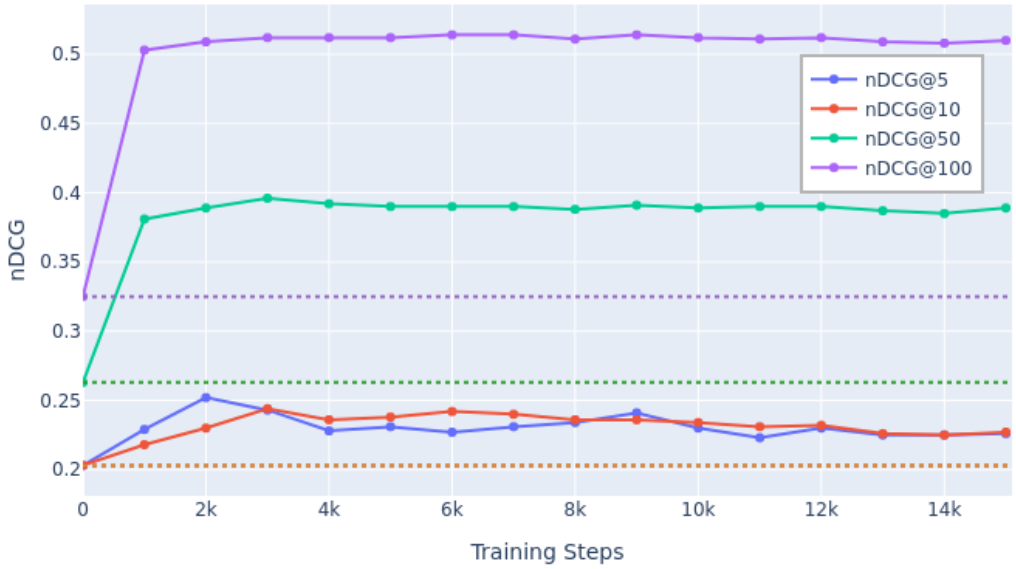

Figure 2: Performance of distilbert-base-v4 finetuned with GPL on CS data at various training steps on Top-100 Elastic Search Documents retrieved per query. Dashed lines indicate the performance of the zero-shot *distilbert-base-v4* model.

Furthermore, Figures 4, 2, and 3 show the performance of the GPL-based fine-tuned model at different training steps for different configurations. As can be seen, the *distilbert-base-v4*

Figure 3: Performance of distilbert-base-v4 finetuned with GPL on CS data at various training steps on Top-500 Elastic Search Documents retrieved per query. Dashed lines indicate the performance of the zero-shot *distilbert-base-v4* model.

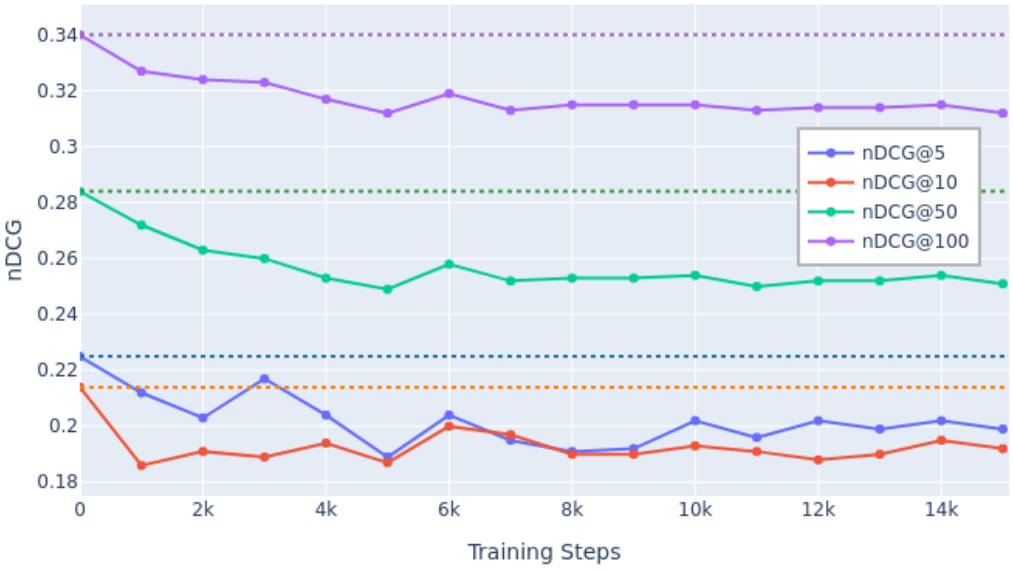

Figure 4: Performance of distilbert-base-tas-b finetuned with GPL on CS data at various training steps on Top-500 Elastic Search Documents retrieved per query. Dashed lines indicate the performance of the zero-shot *distilbert-base-tas-b* model.

fine-tuned on CS data and evaluated based on top-100 ES documents achieves significant improvements with more training steps in the early stages of the training, but the model converges after $2k$ training steps.

The converged model has a significantly higher performance than the zero-shot version in terms of most evaluation metrics. While the *distilbert-base-v4* gets improved by more training steps, the same behavior is not observed for the *distilbert-base-tas-b* model. In fact, this model's performance steadily drops by more training steps. A more detailed analysis on a larger test collection (with more queries and deeper depth) is required to explain this behavior of the model.

5.2. Submitted Runs

Table 4 shows the performance of the submitted runs on the training set using queries. As can be seen, different variations of the *MS-DB-v4* model fine-tuned by GPL an the CS data using queries has the best performance in terms of most metrics. The main variation in the performance of different versions of this model comes from the number of top ranked documents, retrieved by the Elastic Search system, used for re-ranking. Increasing the number of ES documents from 100 to 500 has a negative impact on $nDCG@50$ and $nDCG@100$. This result again shows a possible pooling bias towards the ES model in the training set.

Table 4
Evaluated per Query: Performance of Official Runs on `Simpletext_2023_task1_train.qrels`

Run	P@10	R@10	MRR@10	nDCG@5	nDCG@10	nDCG@50	nDCG@100
1	0.217	0.134	**0.520**	**0.231**	0.218	0.278	0.345
2	**0.259**	0.150	0.516	0.230	0.234	**0.389**	**0.512**
3	0.217	0.134	0.484	0.221	0.210	0.259	0.306
4	0.262	**0.160**	0.439	0.227	**0.243**	0.380	0.511
5	0.207	0.132	0.413	0.211	0.209	0.270	0.332
6	0.197	0.122	0.393	0.200	0.199	0.255	0.300
7	0.214	0.149	0.344	0.149	0.183	0.238	0.288
8	0.172	0.119	0.277	0.114	0.146	0.206	0.262
9	0.141	0.063	0.319	0.132	0.129	0.147	0.167
10	0.153	0.034	0.193	0.096	0.101	0.126	0.143

Table 5 shows the performance of the submitted runs on the training set using topics. Performance of the models based on topics is similar to their query-based performance. However, the model used to re-rank top 5000 documents of the ES system achieves the higher performance in topic-based evaluation.

Table 5
Evaluated per Topic: Performance of official runs on `Simpletext_2023_task1_train.qrels`

Run	P@10	R@10	RR@10	nDCG@5	nDCG@10	nDCG@50	nDCG@100
1	0.160	0.050	**0.593**	0.194	0.166	0.180	0.224
2	0.220	0.070	0.554	0.180	0.187	0.267	0.385
3	0.160	0.050	0.526	0.177	0.153	0.158	0.194
4	**0.287**	**0.097**	0.555	**0.251**	**0.250**	**0.312**	**0.422**
5	0.213	0.064	0.486	0.211	0.195	0.218	0.268
6	0.193	0.053	0.461	0.206	0.182	0.193	0.240
7	0.227	0.073	0.451	0.178	0.190	0.228	0.269
8	0.200	0.059	0.326	0.123	0.135	0.183	0.222
9	0.167	0.050	0.555	0.196	0.164	0.141	0.155
10	0.153	0.034	0.193	0.096	0.101	0.126	0.143

5.3. Official Results

As per Table 6, where the results are sorted on the primary measure, nDCG@10, we see that our submitted runs (e.g. *Elsevier*) dominate the top of the scoreboard.

In particular, the highest performing result, run 8, was obtained by re-ranking top-500 passages retrieved by ElasticSearch when searching for "query, topic" with *MS-DB-v4-GPL-CS*, again searching with query, topic. The selection of configurations(see Table 2) for our submissions were based on our own evaluation on the set of qrels provided to us, which indicated that searching only for the query with *MS-DB-v4-GPL-CS* outperformed our best model for the KAPR task: *MS-DB-tas-b-GPL-all*. However, this set might not have been representative of SimpleText's official evaluation set as most of the other high-ranking results, were obtained with *MS-DB-tas-b-GPL-all*. For instance, run 7 can directly be compared with run 3 as they use the same type of query input and the same type of corpus (i.e. top-1000 ElasticSearch results). This also applies for run 5 versus run 2 and run 6 versus run 1. In each of these settings, the tas-b model fine-tuned on our entire benchmark set outperformed the v4 model fine-tuned on only the Computer Science portion of our test collection.

This indicates that even for the SimpleText task, *MS-DB-tas-b-GPL-all* performs better than *MS-DB-v4-GPL-CS*, and that the success of run 8 could thus be partly attributed to the fact that it was the only run that used "query, topic" as its query input. Using *MS-DB-tas-b-GPL-all* with "query, topic" might thus have outperformed our winning run. Nonetheless, these results show that the model fine-tuned for our specific scientific passage retrieval task still generalizes well to other datasets.

Table 6
Official Results of Simple Text Task 1 - CLEF 2023

Run	MRR	P@10	P@20	P@30	nDCG@10	nDCG@20	nDCG@30	BPREF	MAP
ElsevierSimpleText_run8	0.8082	0.5618	0.3515	0.2696	0.5881	0.4422	0.3803	0.2371	0.1633
ElsevierSimpleText_run7	0.7136	0.5618	0.4103	0.3441	0.5704	0.4627	0.4158	0.2626	0.1915
maine_CrossEncoder1	0.7309	0.5265	0.4500	0.4216	0.5455	0.4841	0.4687	0.3337	0.2754
maine_CrossEncoderFinetuned1	0.7338	0.4971	0.4000	0.3529	0.4859	0.4295	0.4062	0.3443	0.2385
ElsevierSimpleText_run5	0.6600	0.4765	0.3838	0.3314	0.4826	0.4186	0.3834	0.2542	0.1828
ElsevierSimpleText_run2	0.7010	0.4676	0.4059	0.3480	0.4791	0.4282	0.3912	0.2528	0.1942
ElsevierSimpleText_run6	0.6402	0.4676	0.3853	0.3284	0.4723	0.4185	0.3828	0.2557	0.1809
ElsevierSimpleText_run4	0.6774	0.4529	0.3794	0.3422	0.4721	0.4116	0.3876	0.2485	0.1898
ElsevierSimpleText_run9	0.5933	0.4735	0.3176	0.2500	0.4655	0.3595	0.3102	0.1758	0.1238
ElsevierSimpleText_run1	0.6821	0.4588	0.3824	0.3353	0.4626	0.4071	0.3786	0.2573	0.1823
maine_CrossEncoderFinetuned2	0.7082	0.4706	0.3926	0.3637	0.4617	0.4089	0.3969	0.3259	0.2253
UAms_CE1k_Filter	0.6403	0.4765	0.3559	0.2941	0.4533	0.3743	0.3334	0.2727	0.1936
ElsevierSimpleText_run3	0.6502	0.4471	0.3779	0.3324	0.4460	0.3994	0.3709	0.2558	0.1785
UAms_ElF_Cred44	0.6888	0.4324	0.3338	0.2951	0.4103	0.3499	0.3300	0.2395	0.1719
UAms_CE100	0.6779	0.3971	0.3456	0.3137	0.4016	0.3642	0.3483	0.2658	0.1792
maine_Pl2TFIDF	0.5626	0.4176	0.2809	0.2206	0.4014	0.3218	0.2887	0.2155	0.1364
UAms_Elastic	0.6424	0.4059	0.3456	0.2990	0.3910	0.3541	0.3314	0.2501	0.1895
UAms_ElF_Cred53	0.6429	0.4088	0.3382	0.3010	0.3883	0.3468	0.3292	0.2454	0.1833
UAms_ElF_Cred44Read	0.6625	0.3971	0.3147	0.2775	0.3723	0.3282	0.3101	0.2123	0.1403
UAms_CE1k	0.5880	0.4147	0.3515	0.3098	0.3706	0.3398	0.3250	0.2700	0.1865
UAms_CE1k_Combine	0.5880	0.4147	0.3515	0.3098	0.3706	0.3398	0.3250	0.2700	0.1865
UAms_ElF_Read25	0.6076	0.3735	0.3074	0.2833	0.3539	0.3190	0.3105	0.2194	0.1522
UAms_ElF_Cred53Read	0.6088	0.3676	0.3059	0.2784	0.3469	0.3153	0.3042	0.2133	0.1456
maine_tripletloss	0.5502	0.3382	0.2176	0.1608	0.3353	0.2561	0.2145	0.1335	0.0696
uninib_DoSSIER_2	0.5201	0.2853	0.2515	0.2118	0.2980	0.2683	0.2403	0.1898	0.1141
uninib_DoSSIER_4	0.5202	0.2853	0.2441	0.2108	0.2972	0.2632	0.2392	0.1873	0.1111
run-LIA.bm25	0.4536	0.1912	0.1338	0.1108	0.2192	0.1700	0.1505	0.1384	0.0515
run-LIA.all-MiniLM-L6-v2.query	0.3505	0.2000	0.1662	0.1353	0.2019	0.1767	0.1540	0.1956	0.0667
run-LIA.all-MiniLM-L6-v2.query-topic	0.3655	0.1765	0.1485	0.1245	0.1912	0.1647	0.1476	0.2043	0.0591
run-LIA.all-mpnet-base-v2.query-topic	0.3506	0.1647	0.1294	0.1098	0.1835	0.1517	0.1357	0.2073	0.0523
run-LIA.all-mpnet-base-v2.query	0.3302	0.1647	0.1529	0.1294	0.1802	0.1644	0.1462	0.1956	0.0602
run-LIA.lda	0.3138	0.1824	0.1456	0.1245	0.1666	0.1488	0.1387	0.1402	0.0521
run-LIA.es	0.3056	0.1118	0.0912	0.0804	0.1277	0.1080	0.0989	0.1935	0.0342

6. Conclusion

In this paper, we designed several ranking models to address the document retrieval task of the SimpleText lab. To this end, we first built a test collection containing 5000 query-document pairs annotated by relevance labels. The documents in this test collection are extracted from scientific documents which makes it suitable to evaluate performance of ranking models on the scientific document retrieval task. We, then evaluated the performance of existing ranking models on this test collection and selected a few models based on their performance to build our ranking models (used to create our SimpleText submissions). Since these models are trained on generic datasets created for the ad-hoc document retrieval task, they might not have a strong performance on the specific task of scientific document retrieval. To address this issue, we used a domain adaptation technique, namely Generative Pseudo Labeling (GPL) to fine-tune the selected ranking models to the scientific document retrieval task by means of a large collection of unlabeled scientific documents. Our results on the SimpleText training dataset shows the effectiveness of fine-tuning on the performance of our best ranking model. The *distilbert-base-v4* model fine-tuned using GPL on a large collection of documents in Computer Science domain which is used to re-rank top-500 documents retrieved by a Elastic Search system using "topic,

query" as the query input has the highest performance compared to the other fine-tuned models. Using the relevance labels from Computer Science-related domains to fine-tune state-of-the-art ranking models proved successful. However, as only a small portion of our test collection consisted of Computer Science queries, future work could explore labeling a larger set of queries in Computer Science-related domains to fine-tune a model in the same fashion.

References

[1] P. Plavén-Sigray, G. J. Matheson, B. C. Schiffler, W. H. Thompson, The readability of scientific texts is decreasing over time, Elife 6 (2017) e27725.

[2] L. Ermakova, E. SanJuan, S. Huet, O. Augereau, H. Azarbonyad, J. Kamps, Overview of simpletext - clef-2023 track on automatic simplification of scientific texts., in: Avi Arampatzis, Evangelos Kanoulas, Theodora Tsikrika, Stefanos Vrochidis, Anastasia Giachanou, Dan Li, Mohammad Aliannejadi, Michalis Vlachos, Guglielmo Faggioli, Nicola Ferro (Eds.) Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Fourteenth International Conference of the CLEF Association, 2023.

[3] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, I. Gurevych, Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models, arXiv preprint arXiv:2104.08663 (2021).

[4] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, L. Deng, Ms marco: A human generated machine reading comprehension dataset, choice 2640 (2016) 660.

[5] X. Wang, C. Macdonald, I. Ounis, Improving zero-shot retrieval using dense external expansion, Information Processing & Management 59 (2022) 103026.

[6] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, W.-t. Yih, Dense passage retrieval for open-domain question answering, arXiv preprint arXiv:2004.04906 (2020).

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

[8] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, 2019.

[9] R. Nogueira, K. Cho, Passage re-ranking with bert, arXiv preprint arXiv:1901.04085 (2019).

[10] R. Nogueira, W. Yang, J. Lin, K. Cho, Document expansion by query prediction, arXiv preprint arXiv:1904.08375 (2019).

[11] R. Nogueira, Z. Jiang, J. Lin, Document ranking with a pretrained sequence-to-sequence model, arXiv preprint arXiv:2003.06713 (2020).

[12] S. MacAvaney, A. Yates, A. Cohan, N. Goharian, Cedr: Contextualized embeddings for document ranking, in: Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, 2019, pp. 1101–1104.

[13] S. MacAvaney, F. M. Nardini, R. Perego, N. Tonellotto, N. Goharian, O. Frieder, Efficient document re-ranking for transformers by precomputing term representations, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 49–58.

[14] C. Li, A. Yates, S. MacAvaney, B. He, Y. Sun, Parade: Passage representation aggregation for document reranking, arXiv preprint arXiv:2008.09093 (2020).
[15] K. Wang, N. Thakur, N. Reimers, I. Gurevych, Gpl: Generative pseudo labeling for unsupervised domain adaptation of dense retrieval, arXiv preprint arXiv:2112.07577 (2021).
[16] S. Hofstätter, S.-C. Lin, J.-H. Yang, J. Lin, A. Hanbury, Efficiently teaching an effective dense retriever with balanced topic aware sampling, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 113–122.
[17] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).