PSU at CLEF-2020 ARQMath Track: Unsupervised Re-ranking using Pretraining

Shaurya Rohatgi[1], **Jian Wu**[3], **C. Lee Giles**[1]
[1] Pennsylvania State University, [3] Old Dominion University
[1]{szr207,clg20}@psu.edu, [3]jwu@cs.odu.edu

Abstract. This paper elaborates on our submission to the ARQMath track at CLEF 2020. Our primary run for the main Task-1: Question Answering uses a two-stage retrieval technique in which the first stage is a fusion of traditional BM25 scoring and tf-idf with cosine similarity-based retrieval while the second stage is a finer re-ranking technique using contextualized embeddings. For the re-ranking we use a pre-trained roberta-base model (110 million parameters) to make the language model more math-aware. Our approach achieves a higher NDCG′ score than the baseline, while our MAP and P@10 scores are competitive, performing better than the best submission (MathDowsers) for text and text+formula dependent topics.

Keywords: Math Information Retrieval · Math Embeddings · Math-aware search · Math formula search

1 Introduction

The Intelligent Information Systems Research Laboratory at Pennsylvania State University participated in the CLEF-2020 ARQMath track to contribute and develop new techniques for math-aware information retrieval. One of the main goals [9] of this track was to push mathematical question answering into an informal language scenario, specifically using data from Math Stack Exchange (MSE). We participated in Task-1: Answer Retrieval track, the goal of which was to return ranked lists of past answers given an actual recent post containing math formulas and text.

Question answering has elements of both relevance matching and semantic matching but remains a different task from document retrieval [16]. We use a two phase retrieval and re-ranking approach. In the first phase, retrieval is based on two runs: tf-idf representation using BM25 scoring and cosine similarity. Once we obtain the top 1000 candidates from the first phase, we re-rank them using contextualized BERT-based embeddings from the math-aware language model trained on the MSE dataset. This language model makes these embeddings semantically rich. Finally, we fuse the results from BM25, cosine, and BERT-based re-ranking runs. Our contributions are:

– Achieve a competitive score that beats the best baseline in terms of the NDCG′ score.
– Achieve better results than the best submission for Text and Text+Math dependent posts.
– Propose a Masked Language Model[1] used for re-ranking candidate answers containing both text and math formulas.

Our paper has the following Sections. Section 2 discusses our two-stage retrieval approach which includes indexing and re-ranking phases. Section 3 describes our experimental setup, system configuration, and a detailed comparison of our results with the best submission. Conclusions are in Section 4.

2 Our Approach

Here we describe our two-stage cascade candidate retrieval and ranking approach. tf-idf with cosine similarity and an off-the-shelf BM25-based search platform [15] are used for the first stage. This is relatively computationally cheaper than the second more expensive re-ranking stage. For the second stage, we use a pre-trained language model to obtain semantically rich embeddings for the candidates obtained from the first stage. We then use these embeddings to rank the candidates using a cosine distance to the topic/query embedding. This ensures that the ranking captures semantics between the query and the documents. We only rely on the content of the post, which contain raw text and math formulas, not using external information such as votes/scores, user_id, user score, post tags, and linked duplicate posts.

2.1 First-Phase: Retrieval

The aim of this phase is to get a sufficient number of relevant candidates to be re-ranked in the next stage. Past work has used BM25 scoring for this but we add cosine similarity ranking as well.

Indexing: We convert the MSE dataset from XML to JSON format so that it can be ingested by the search platforms we use. Because indexing all answers will potentially lose information in the questions, we generate a document by concatenating the question (Q) post text (title and body) with their corresponding answer (A) posts text (body). As such, the question body and title concatenated with the answer body become a document - one unit of retrieval. This allows us to remove questions without corresponding answers and represent the relevant posts as one large document because the relevant information the user is seeking could be in either the answer or the question post. This results in a total of 1,435,643 Q+A posts to be indexed.

We index the data using two off-the-shelf libraries - Elasticsearch (ES) and Anserini. We use two different libraries because each has its strengths and weaknesses. Anserini seems to perform better than ES in terms of relevance ranking

[1] `https://huggingface.co/shauryr/arqmath-roberta-base-1.5M`

and recall, which we observed when searching and evaluating the three training queries for the Question Answering Task provided by the organizers. Elasticsearch has a more scalable implementation of tf-idf with cosine similarity which is slow for a large dataset when using Anserini. For the formulas, we use the raw LaTeX strings. We re-rank answers based on contextualized text and formulas embeddings using RoBERTa.

Retrieval: Once all posts in Elasticsearch and Anserini have been indexed, we query the topics/queries for Task-1 to get the top-1000 candidates. For each task, each index is queried independently and later fused using Reciprocal Rank Fusion (RRF) [2] which then ranks the documents using a naive scoring formula. Given a set of posts P to be ranked and a set of rankings R from different scoring schemes, for each permutation on $1 \cdots |P|$, we compute

$$RRFscore(p \in P) = \sum_{r \in R} \frac{1}{k + r(p)}, \tag{1}$$

where the original work [2] suggests $k = 60$, a hyper-parameter which we keep constant. The fusion of results from two different search platforms above significantly increases the performance numbers as demonstrated in Section 3.

2.2 Second-Phase: Re-Ranking

Here we describe how we pretrain our language model and re-rank candidates obtained in the last phase.

BERT [5] is a self-supervised approach for fine-tuning a deep transformer encoder [14]. Given a sequence, BERT learns a contextualized vector representation for each token. The input representations are fed into a stack of multi-layer bidirectional transformer blocks, which uses self-attention to compute semantically rich text representations by considering the whole input sequence.

BERT-based systems [4][17][11][10] have shown significant performance in the recent tasks of TREC-2019 deep learning track [3] with the Microsoft MARCO dataset [1]. Participants for these tasks used query-document pairs from the training data to train transformer-based models to predict relevance. However, such models rely on a massive amount of data, which is not available in our task. Therefore, instead of training for relevance, we leveraged an unsupervised model by calculating the semantic similarities of queries and documents and later using them for re-ranking.

We choose the RoBERTa model [8] over BERT because in our experiments Vanilla RoBERTa achieves better NDCG′ scores than Vanilla BERT for the three preliminary training posts provided by the task organizers. Also, RoBERTa converges in fewer training steps than BERT as it gets rid of the computationally expensive next sentence prediction task. RoBERTa attains better performance on various NLP tasks than BERT [7]. We start with the initial weights of the *roberta-base* model and further pretrain the model using MSE data. The language model is trained for a mask prediction task. Once we are done training we use

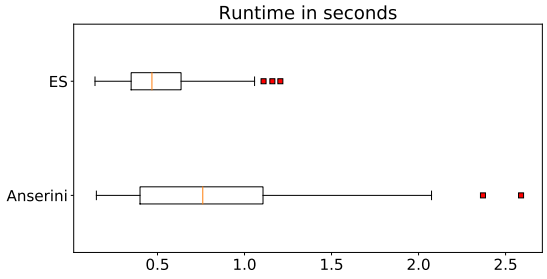


Fig. 1: Comparison of run-times for all the topics averaged over 10 runs. Fastest topics to retrieve top-1000 for ES and Anserini were A.39 and A.88 respectively while the slowest topics were A.87 and A.47 respectively.

the Masked Language Model (MLM) to get contextualized token embeddings averaged over the sequence length to represent the candidates from Phase-1 into a 768 dimension vector. Similarly, we obtain topic/query embeddings and rank the candidates using their cosine distance.

3 Experiments and Results

Our experiments were conducted on a 24 core machine with Intel(R) Xeon(R) Silver 4116 CPU @ 2.10GHz with 256GB of RAM with 4 RTX 2080 Ti GPUs. Default configurations were adopted in ES (cosine similarity; tf-idf) and Anserini (BM25). Anserini runs on a single thread while ES uses a multi-threaded function. In Figure 1 we compare runtimes of BM25 ranking using Anserini and tf-idf based cosine similarity ranking using ES. The size of ES Q+A index is 3.6GB whereas for Anserini the size is 2.2GB.

Pretraining RoBERTa: We use the transformers[2] library's implementation of RoBERTa to train the MLM. We start with the original weights released by the authors for *roberta-base* and then further pretrain the model on the MSE dataset. Fortunately, BASE-vocabulary used in the original was able to cover the whole MSE dataset so did not train from scratch. Further, we reduce the batch size to 4 per GPU and increase step size to facilitate gradient accumulation. We kept the maximum sequence length of 512 to accommodate longer posts.[3] Q+A posts are usually longer than 512 tokens so we had to break the posts into chunks before feeding them to our system.

Once we are done with pretraining our MLM, we use it to extract the embeddings for each token in the candidate posts from the first-phase. For longer Q+A posts, we had to find a way of truncating them so that the sequence can fit

[2] `https://github.com/huggingface/transformers`
[3] Link to training details

in the 512 token window. We experimented with the $head - tail$ approach [13]. The $head - tail$ approach gave a lower NDCG′ score for the three preliminary train topics. Therefore, we use the *head* approach, in which we keep the first 510 tokens from the Q+A post and leave two token spaces for $[CLS]$ and $[SEP]$. This gives better results for the trained topics. This is likely because our Q+A posts include question text, so if you can find a similar question to a topic, then the answer to the similar question has a higher chance of being an answer to the topic.

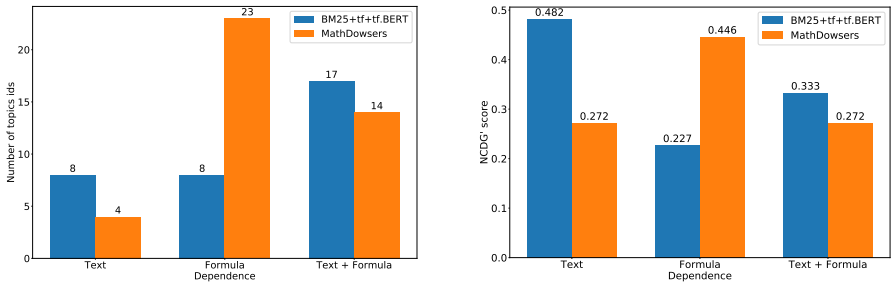
Table 1: Results for Task-1 compared with other submissions and best baselines. **tf** is tf-idf representation and cosine similarity-based ranking and **tf.BERT** signifies re-ranking using RoBERTa embeddings of the candidates selected by **tf**. Reciprocal rank fusion is used to fuse the runs separated by + sign. (* represents our best run)

	Run Tag	Evaluation Measures			
		NDCG′	MAP′	P@10	Rel_Ret
Baselines	Linked MSE Posts	0.303	**0.210**	**0.417**	395
	Approach-0	0.250	0.100	0.062	441
Official Runs	PSU1 (tf+tf.BERT)	0.263	0.082	0.116	761
	PSU2 (tf)	0.228	0.054	0.055	761
	PSU3 (tf.BERT)	0.221	0.046	0.026	761
Other Comparable Systems	MIRMU	0.238	0.064	0.139	708
	MathDowsers (Task-1 Best)	**0.345**	0.139	0.161	804
Unsubmitted Runs	BM25+tf+tf.BERT*	0.314	0.097	0.149	**902**
	BM25+tf	0.304	0.098	0.151	875
	BM25	0.246	0.078	0.139	660

We show in Table 1 how our system compares with other submissions and the baselines. We only include the best submissions for baselines and other systems. Our system BM25+tf+tf.BERT clearly beats the best baseline in terms of NDCG′. Before the challenge, we had only three train topics provided by the organizers on which we could test our approach. For these three topics tf-idf with cosine similarity was running better than BM25 scoring, so we did not include BM25 in our final submission. Later we added BM25 scoring which showed a greater increase in the number of relevant retrieved documents. It can be seen that **tf.BERT** that selects candidates using tf-idf similarity and re-ranking using our pretrained RoBERTa model does not achieve the best performance. But when the rankings of tf.BERT and tf-idf are fused, NDCG′ [12] score is substantially boosted. This is because the BERT ranking solely relies on semantic similarity whereas the tf-idf replies only on word frequency. Fusing these two runs seems reasonable since the posts which have a higher rank in both the runs are boosted even higher in the final ranking list. Note that BM25+tf+tf.BERT achieves the maximum number of relevant retrieved posts. It is important to

note that the tf runs are not as consistent since ES does sharding and round-robins between different shard searching. Thus, we did multiple runs to achieve the above scores.

Comparison with Other submissions: Our best unsubmitted run, BM25+tf+tf.BERT was compared with other submissions, among which MathDowsers is the system that achieved one of the best results in Task 1. Submissions by Approach0 and MathDowsers leveraged Tangent-S [18] and Tangent-L [6], which are Symbol Layout and Operator Tree-based systems giving more attention to the math formula in the text. Our post-hoc system does not use a different representation for formula and text, and therefore seems to suffer for formula dependent topics. We believe representing math content as trees and separately from the text content could have benefited our scores.



(a) Number of topics for which each system had higher NCDG′ than the other system.

(b) Average NCDG′ over total topics depending on different types of topics.

Fig. 2: Comparison of runs for the 77 topics in Task-1 based on dependence class of the topics. We clearly see that for the topics that depend on Text and Text+Formula our system performs better.

In Figure 2 we see that our system is better for the topics that depend on text and text+formula. For Text dependent topics MathDowsers had 4 topics for which there NDCG′ score was higher than our best run, while our system performed better in 8 topics. For the 31 text+formula dependent topics our system had better NDCG′ score for 17 topics. MathDowsers achieves a higher ($\approx 96.4\%$ higher) NDCG′ score for the topics which are only formula dependent. In contrast, our system is better ($\approx 22.42\%$) at ranking posts containing both formula and text. This is attributed to the contextualized embeddings which our pretrained MLM can produce. It models equations with surrounding text and hence has better performance in text+formula topics. The difference ($\approx 77.2\%$) is even more when we compare text dependant topics.

4 Conclusion

Overall, our participation in the ARQMath Track helped in our understanding of how to improve multi-modal search (text+formula) by exploiting state-of-the-art text embedding and information retrieval models. In terms of effectiveness, our most effective post-hoc run BM25+tf+tf.BERT was able to outperform the baselines and all submissions except Mathdowsers in terms of NDCG′. Our post-hoc system achieved the best results for queries that depend on text and text+formula.

Future work would include the use of the MSE dataset to get question-answer post pairs to train a better ranking model. It would also be helpful to investigate how important a formula is in a question post to retrieve relevant answers.

Acknowledgements

We would like to thank members of the ARQMath lab at the Department of Computer Science in Rochester Institute of Technology for organizing this track. Special thanks to Behrooz Mansouri for providing the dataset, initial analysis of topics, and starter code to all the participants of the task; it made it easier for us to pre-process the data and jump directly to the experiments which have been presented in this work.

References

1. Bajaj, P., Campos, D., Craswell, N., Deng, L., Gao, J., Liu, X., Majumder, R., McNamara, A., Mitra, B., Nguyen, T., et al.: Ms marco: A human generated machine reading comprehension dataset. arXiv preprint arXiv:1611.09268 (2016)
2. Cormack, G.V., Clarke, C.L., Buettcher, S.: Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 758–759 (2009)
3. Craswell, N., Mitra, B., Yilmaz, E., Campos, D., Voorhees, E.M.: Overview of the trec 2019 deep learning track. arXiv preprint arXiv:2003.07820 (2020)
4. Dai, Z., Callan, J.: Deeper text understanding for ir with contextual neural language modeling. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 985–988 (2019)
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
6. Fraser, D., Kane, A., Tompa, F.W.: Choosing math features for BM25 ranking with tangent-l. In: Proceedings of the ACM Symposium on Document Engineering 2018. pp. 1–10 (2018)
7. Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., Smith, N.A.: Don't stop pretraining: Adapt language models to domains and tasks. arXiv preprint arXiv:2004.10964 (2020)

8. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
9. Mansouri, B., Agarwal, A., Oard, D., Zanibbi, R.: Finding old answers to new math questions: The arqmath lab at clef 2020. In: Jose, J.M., Yilmaz, E., Magalhães, J., Castells, P., Ferro, N., Silva, M.J., Martins, F. (eds.) Advances in Information Retrieval. pp. 564–571. Springer International Publishing, Cham (2020)
10. Nogueira, R., Cho, K.: Passage re-ranking with bert. arXiv preprint arXiv:1901.04085 (2019)
11. Nogueira, R., Yang, W., Cho, K., Lin, J.: Multi-stage document ranking with bert. arXiv preprint arXiv:1910.14424 (2019)
12. Sakai, T., Kando, N.: On information retrieval metrics designed for evaluation with incomplete relevance assessments. Information Retrieval **11**(5), 447–470 (2008)
13. Sun, C., Qiu, X., Xu, Y., Huang, X.: How to fine-tune bert for text classification? In: China National Conference on Chinese Computational Linguistics. pp. 194–206. Springer (2019)
14. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in neural Information processing systems. pp. 5998–6008 (2017)
15. Yang, P., Fang, H., Lin, J.: Anserini: Enabling the use of lucene for information retrieval research. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 1253–1256 (2017)
16. Yang, W., Zhang, H., Lin, J.: Simple applications of bert for ad hoc document retrieval. arXiv preprint arXiv:1903.10972 (2019)
17. Yilmaz, Z.A., Wang, S., Yang, W., Zhang, H., Lin, J.: Applying bert to document retrieval with birch. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations. pp. 19–24 (2019)
18. Zhong, W., Rohatgi, S., Wu, J., Giles, C.L., Zanibbi, R.: Accelerating substructure similarity search for formula retrieval. In: Jose, J.M., Yilmaz, E., Magalhães, J., Castells, P., Ferro, N., Silva, M.J., Martins, F. (eds.) Advances in Information Retrieval. pp. 714–727. Springer International Publishing, Cham (2020)