Zero-shot Hybrid Retrieval and Reranking Models for Biomedical Literature

Jing Lu, Ji Ma and Keith Hall

Google Research

Abstract
We describe our participating system in the document retrieval sub-task (Task B Phase A) at the 10th BioASQ challenge. We designed and implemented a zero-shot hybrid model using only synthetic training data. The model consists of two stages: retrieval and reranking. The retrieval model is a hybrid of sparse and dense retrieval models, which is an extension of our participating system at 8th BioASQ challenge. We improved the dense retrieval model with a T5-based synthetic question generation model and an iterative training strategy involving techniques to filter low-quality synthetic data. In the second stage, we proposed a hybrid reranking model, which is trained using the candidates retrieved from the first stage. We further explored whether the knowledge from the hybrid reranking model can be transferred to the dense retrieval model through distillation. Our experiments show the proposed hybrid ranking model is effective even when applied to different first-stage retrieval models. Furthermore, we explored the combination of different systems via reciprocal rank fusion and achieved additional accuracy gains. Evaluation shows that our model compares favorably with the top participating system, achieving MAP scores of 0.4696, 0.3984, 0.4586, 0.4089, 0.4065 and 0.1704 on six batches.

Keywords
document retrieval, reranking, question generation, BioASQ

1. Introduction

We participated in the document retrieval sub-task (Task B Phase A) at the 10th BioASQ challenge. The task aims to retrieve relevant articles from PubMed[1] to biomedical questions constructed by a team of biomedical experts. In this paper, we present our system developed for this task. We designed and implemented a zero-shot hybrid model which consists of two stages: retrieval and reranking, and uses only synthetic training data. Our contributions in this paper are three-fold. First, we show the effectiveness of a zero-shot model which doesn't need any labeled data from the biomedical domain. Second, many prior works explore the two-stage models[1, 2, 3], but they use either a sparse model (e.g., BM25) or a dense model (e.g., a dense neural retriever) to generate training data for training the reranker. We show that by training a reranker from a hybrid retriever, the reranker not only performs better when presented with results from a hybrid retriever, but it performs very well with results from other retrievers. The observed effect is that the reranker is able to capture both term matching as well

CLEF 2022: Conference and L of the Evaluation Forum, September 5–8, 2022, Bologna, Italy
EMAIL: ljwinnie@google.com (J. Lu); maji@google.com (J. Ma); kbhall@google.com (K. Hall)
ORCID: 0000-0003-1076-7662 (J. Lu); 0000-0001-8577-175X (J. Ma); 0000-0002-0977-6572 (K. Hall)

[1]https://pubmed.ncbi.nlm.nih.gov/

as semantic matching. Results on 3 out of 6 test batches outperform other participating systems showing the effectiveness of our proposed model. Third, we investigate the distillation of the hybrid reranking model to a dense retrieval model. The distilled dense model outperforms the non-distilled dense model and sparse model on the development set.

2. Model

In this section, we describe our system, which consists of two stages: retrieval and reranking. In both stages, our model relies on language models pretrained on PubMed articles (Section 2.1). In the retrieval stage, we use a model which is a hybrid of BM25 and a dual encoder model (Section 2.2). In the reranking stage, we use a cross-attention model with ranking loss, which is trained using the candidates retrieved from the first stage (Section 2.3). In addition, we explore transferring the knowledge from the reranking model to the dual encoder model through distillation (Section 2.4).

2.1. Pretrained Language Model

Language models such as GPT[4, 5, 6], BERT [7] and T5[8] pre-trained on large scale corpora provide rich knowledge for many downstream NLP tasks. For the biomedical domain, previous research shows that language models adapted on biomedical corpora can bring further improvement on biomedical NLP tasks [9]. In this work, we use domain adapted BERT-based and T5-based models for retrieval and reranking models.

The BERT-based model is pre-trained from scratch using PubMed abstracts along with the commercial Open-Access PubMed Central corpus as distributed by the National Library of Medicine; we refer it as PubMed_BERT. We created a specialized wordpiece vocabulary from the training corpus containing 107137 entries. The PubMed_BERT model consists of 12 transformer [10] layers, each with hidden size 1024 and 16 attention heads. We use the same sentence sampling procedure as reported in the original BERT paper, e.g., the combined sequence has length no longer than 512 tokens, and we uniformly mask 15% of the tokens from each sequence for masked language model prediction. We update the next sentence prediction task with softmax cross-entropy loss. We use the same hyper-parameter values for BERT pretraining except that the learning rate is set 2e-5, and the model is trained for 300,000 steps.

We also fine-tune a T5 model on PubMed abstracts using the span corruption task [8]. We refer it as PubMed_T5. Specifically, we set the max input and target length to 512 and 114 respectively. We tune the T5.1.1.xl model[2] for 1 million steps with a learning rate of 0.01 and dropout rate 0.1.

2.2. Hybrid Retrieval Model

We extend our hybrid first-stage retrieval model used in the 8th BioASQ challenge [11]. Specifically, we use a BM25 model as the sparse retrieval model and a dual encoder model as the dense retrieval model. We cast both models as vector similarity via nearest neighbor search and create

[2]gs://t5-data/pretrained_models/t5.1.1.xl

hybrid encodings which are a concatenation of the BM25 and dual encoder encodings. We refer the reader to the original paper for more details. We briefly describe the model and focus on the extension in this section.

For the BM25 model, we represent each question as a $|V|$-dimensional binary encoding $\mathbf{q}^{\text{bm25}}$, where $\mathbf{q}^{\text{bm25}}[i]$ is 1 if the i-th entry of vocabulary V is in the question, 0 otherwise. We represent each passage as a sparse real-valued vector $\mathbf{p}^{\text{bm25}}$:

$$\mathbf{p}_i^{\text{bm25}} = \frac{\text{IDF}(p_i) * \text{cnt}(p_i, P) * (k+1)}{\text{cnt}(p_i, P) + k * (1 - b + b * \frac{m}{m_{\text{avg}}})}.$$

where p_i are tokens from passage P, cnt(p_i, P) is p_i's term frequency in P, k/b are BM25 hyperparameters, IDF is the term's inverse document frequency from the document collection, m are the number of tokens in P, and m_{avg} is the collection's average passage length. We use vector dot-product to measure the question and passage relevance.

Our dual encoder model is based on BERT [7]. To encode a question, we feed the question text to the BERT model and apply a fully-connected (FC) layer of size 768 to the [CLS] token embedding. The output of the FC layer is used as the question encoding $\mathbf{q}^{\text{de}}$. A passage encoding $\mathbf{p}^{\text{de}}$ is generated in a similar way but we concatenate the document and corresponding document title as the input to the BERT model: [CLS] *title* [SEP] *passage* [SEP]. The question to passage relevance is computed by the cosine similarity of their vectors.

Following our previous work [12], we train our dual encoder models using synthetically generated questions. We apply a question generation model (QGen) to the abstracts of PubMed articles to generate (synthetic question, passage) pairs. We then use these data to train a dual encoder model DE_0. To filter low quality questions, we adapt the roundtrip consistency [13, 14] idea to retrieval. Given a synthetic question in the training data, we run 1-nearest neighbor search based on scores between the question and all passages using DE_0. If the neighbor is the one from which the question is generated, we keep that (question, passage). Otherwise, the (question, passage) pair is filtered. With the filtered data, we continue fine tune DE_0 to get the final dual encoder model DE_1.

We use two QGen models in this work, namely NQ_QGen and SQuAD_QGen, created by fine-tuning a general T5 model using the question and passage pairs from Natural Question(NQ)[15] and SQuAD[16], respectively. Particularly, we form the input as "Generate question >>> title.passage >>> target sentence", and the output is the corresponding question. Here "target sentence" is the sentence that contains the short answer span, and "passage" corresponds to long answer and the passage of NQ and SQuAD respectively. At inference time, given a PubMed abstract, we iterate over every sentence as the target to generate diverse questions. We generate synthetic questions from each of the QGen models and our preliminary experiment shows that mixing them as the training data for dual encoder models result in better performance than using synthetic training data from each individual model.

To benefit from both sparse model and dense neural model, we create the hybrid model by combining the encodings from two models in a principled way:

$$\begin{aligned}\text{sim}(\mathbf{q}^{\text{hyb}}, \mathbf{p}^{\text{hyb}}) &= \langle \mathbf{q}^{\text{hyb}}, \mathbf{p}^{\text{hyb}} \rangle \\ &= \langle [\mathbf{q}^{\text{bm25}}, \lambda \mathbf{q}^{\text{de}}], [\mathbf{p}^{\text{bm25}}, \mathbf{p}^{\text{de}}] \rangle\end{aligned}$$

$$= \langle \mathbf{q}^{\mathrm{bm25}}, \mathbf{p}^{\mathrm{bm25}} \rangle + \lambda \langle \mathbf{q}^{\mathrm{de}}, \mathbf{p}^{\mathrm{de}} \rangle,$$

where $\mathbf{q}^{\mathrm{hyb}}$ and $\mathbf{p}^{\mathrm{hyb}}$ are the hybrid encodings that concatenate the BM25 ($\mathbf{q}^{\mathrm{bm25}}/\mathbf{p}^{\mathrm{bm25}}$) and the dual encoder encodings ($\mathbf{q}^{\mathrm{de}}/\mathbf{p}^{\mathrm{de}}$ described above; and λ is an interpolation hyperparameter that trades-off the relative weight of BM25 versus the dual encoder models.

2.3. Hybrid Reranking Model

Our reranking model is a listwise model. To train the model, for each question, we generate a list of 1 positive example and N negative examples. We randomly sample negative examples from top retrieved results from rank i to rank j using the hybrid retrieval model described in Section 2.2. We skip the first i results to avoid false negatives. In addition, we ignore questions whose gold passage is below rank k. We use 15, 100, 5 for i, j and k in this work. We experiment with two ranking models: BERT-based reranking model and T5-based reranking model.

BERT-based reranking model is based on TFR-BERT model [17]. The passage for a given question is represented as "[CLS] question [SEP] passage [SEP]". The pooled BERT output is used as the ranking score. For all passages retrieved for a question, the ranking result is obtained by sorting the passages based on their ranking scores. The model is trained with softmax loss of list size 50. The model is implemented using the TFR-ranking package[3].

For T5-based reranking model, we only use the encoder and discard the decoder. We represent the question-passage pair as input sequence "Query: {question} Document: {passage}" and feed it into the encoder. The output of the encoder is the encodings of the input sequence. We then apply a projection layer on the encoding of the first token and the output is used as the ranking score. As with the BERT-based model, the ranking result of a given question is obtained by sorting the passages based on their ranking scores. We optimize the model using the listwise softmax cross entropy loss function. We implement the model using T5X[4] and we also use RAX [18], a learning-to-rank framework for implementing the ranking loss.

2.4. Distillation

The hybrid reranking model learns both the term matching and semantic similarity between question and passage pairs. We apply distillation to transfer the knowledge learned by the reranking model to the dual encoder model. We first train a T5-based dual encoder model [19] using synthetic training data and then run inference over the same synthetic dataset. We sample negative examples from the top K retrieved passages for each question. We then use the T5-based reranking model as the teacher and score each question-passage pair. To train the student dual encoder model, we use the weighted sum of batch softmax loss computed on in-batch negatives and ranking cross-entropy loss computed on sampled negatives using scores from the teacher model as soft labels.

[3]https://github.com/tensorflow/ranking
[4]https://github.com/google-research/t5x

3. Evaluation

3.1. Data

We use the articles from the PubMed Annual Baseline Repository for 2021[5] as the document collection. During preprocessing, we break each article into passages of 300 tokens. For each passage in the article, we iterate over every sentence in the passage as target and generate one synthetic question from each target. We generate around 25 million unique synthetic questions in total. After filtering, around 7 million synthetic questions are left. We use the 5 test batches from BioASQ9b as the development set for hyperparameter tuning. It has 497 questions in total. We use the official 6 test batches for evaluation and each test batch is released every two weeks. Batch 1–5 contain 90 questions in each batch and batch 6 contains 37 questions.

3.2. Systems

We experiment with five retrieval models: **BM25**, which is a unigram model using the wordpiece tokenizer and the vocabulary of PubMed_BERT as described in section 2.1. Its IDF values are computed on the document collection. **BERT DE**, which is the BERT-based dual encoder model described in section 2.2. It is initialized from the pre-trained PubMed_BERT model. **Hybrid**, which is the hybrid of **BM25** and **BERT DE** with $\lambda = 50$ which is achieved by running a grid search on the development set. **Distill T5 DE** is the distilled T5-based dual encoder model described in section 2.4. It is initialized from the PubMed_T5. **Distill Hybrid** is also a hybrid model of **BM25** and **Distill T5 DE** with $\lambda = 100$.

In order to understand how the hybrid reranking model **HybridRR** performs on different retrieval models, we submitted three systems that use the same T5-based **HybridRR** model trained with the candidates generated from **Hybrid** retrieval model. We apply this reranking model on **BM25**, **BERT DE** and **Hybrid** respectively. We also submitted one system that uses BERT-based **HybridRR** model to understand how different pre-trained models affect the reranking model. In the later batches, we submitted two systems that use the reranking model **Distill HybridRR** trained from **Distill Hybrid** to understand the effectiveness of distillation. We apply this reranking model on **Hybrid** and **Distill Hybrid** separately. Finally, we submitted an ensemble system following the reciprocal rank fusion method (RRF)[20]. We compute the RRF score for a document d as follows:

$$RRF(d) = \sum_{r \in R} \frac{1}{k + r(d)}$$

where $r(d)$ is the rank of document d from system r, and we set $k = 0$, after searching from [0, 100] with a step size 10.

We tried different combinations of above mentioned systems, and the ensemble of three systems that use the same T5-based **HybridRR** model outperforms other combinations when we evaluate on the development set.

[5]https://ftp.ncbi.nlm.nih.gov/pubmed/baseline/

Table 1
Mean average precision (MAP) official results for batches 1–6. “-” indicates that we did not submit those systems for official evaluations.

	Systems	Batch 1	Batch 2	Batch 3	Batch 4	Batch 5	Batch 6
1	BM25 + T5 HybridRR	-	0.3629	0.4087	0.3782	0.3675	0.1677
2	BERT DE + T5 HybridRR	-	0.3647	0.4151	0.3864	0.3593	0.1698
3	Hybrid + T5 HybridRR	-	**0.3666**	0.4256	0.3904	0.3687	**0.1704**
4	Hybrid + BERT HybridRR	0.4154	0.3506	-	-	-	-
5	Hybrid + Distill HybridRR	-	-	-	-	0.3588	0.1553
6	Distill Hybrid + Distill HybridRR	-	-	-	0.3778	0.3572	0.1551
7	RRF (1, 2, 3)	-	-	**0.4304**	**0.3913**	**0.3757**	0.1657
	Best Reporting System	0.4805	0.3977	0.5063	0.4058	0.4154	0.1704

Table 2
Mean average precision (MAP) results for development set (DEV) and updated results for test batches 1–5 and their AVG MAP.

	Systems	DEV	Batch 1	Batch 2	Batch 3	Batch 4	Batch 5	AVG
1	BM25	0.3514	0.3722	0.3054	0.3558	0.2852	0.3214	0.3280
2	BERT DE	0.3262	0.3357	0.2740	0.3432	0.2977	0.3172	0.3136
3	Hybrid	0.4113	0.4121	0.3197	0.3820	0.3522	0.3615	0.3655
4	Distill T5 DE	0.3524	0.3167	0.2917	0.3141	0.3394	0.2972	0.3118
5	Distill Hybrid	0.4163	0.3944	0.3291	0.3885	0.3544	0.3620	0.3657
6	BM25 + T5 HybridRR	0.4563	0.4451	0.3946	0.4493	0.3963	0.3972	0.4165
7	BERT DE + T5 HybridRR	0.4492	0.4524	0.3940	0.4506	0.4054	0.3996	0.4204
8	Hybrid + T5 HybridRR	0.4633	0.4542	**0.3984**	0.4568	**0.4093**	0.3994	0.4236
9	Hybrid + DistillHybRR	**0.4661**	**0.4696**	0.3918	0.4439	0.4089	0.3898	0.4208
10	DistillHyb + DistillHybRR	0.4609	0.4685	0.3902	0.4480	0.4070	0.3911	0.4210
11	RRF (6, 7, 8)	0.4643	0.4571	0.3973	**0.4586**	0.4087	**0.4065**	**0.4256**
	Best Reporting System		0.4805	0.3977	0.5063	0.4058	0.4154	0.4411

3.3. Results and Analysis

Table 1 shows the official results of submitted systems. Not all systems were submitted to all batches as we updated models while the challenge was underway. We report the mean average precision (MAP) which is the official metric used in the task. The ensemble model showed in row 7 always outperforms the single models except on Batch 6. Our two-stage hybrid model in row 3 outperforms other single models; it achieves the best result among all participating systems in Batch 6.

After Batch 5, we noticed that some abstracts were missing after the prepossessing. We fixed the issue and re-evaluated Batch 1–5. Table 2 shows the updated results. Row 1-5 shows the results of first-stage retrieval systems. Row 6-10 shows the reranking results of different combinations of retrieval models and reranking models. As we can see, our two-stage hybrid model in row 8 outperforms the best reporting system and ensemble model (row 11). Comparing row 1,2 and 3, we can see that the hybrid retrieval model outperforms single retrieval models by 0.0375-0.052 points in average MAP. After applying the hybrid reranking model as shown in

Table 3
MAP and Recall@10 of each question type for the best model in test batches 1–6.

	MAP					Recall@10				
	Factoid	List	Summary	Yes/No	All	Factoid	List	Summary	Yes/No	All
Batch 1	0.4972	0.6498	0.3135	0.4480	0.4696	0.5573	0.7471	0.4654	0.6416	0.5890
Batch 2	0.2823	0.3297	0.5822	0.4403	0.3984	0.4319	0.4711	0.6472	0.5274	0.5125
Batch 3	0.4061	0.4796	0.6032	0.3895	0.4586	0.6406	0.5857	0.7091	0.4967	0.6107
Batch 4	0.2844	0.3845	0.4763	0.5187	0.4093	0.3742	0.4988	0.5848	0.6944	0.5300
Batch 5	0.3982	0.3871	0.4798	0.3837	0.4065	0.4825	0.4652	0.6578	0.5042	0.5150
Batch 6	0.2875	0.1178	0.1620	0.1986	0.1704	0.3333	0.2315	0.2317	0.2556	0.2520
WAVG	0.3710	0.3843	0.4678	0.4222	0.4086	0.4923	0.4918	0.5800	0.5563	0.5287

row 6,7, and 8, the difference is only 0.0032-0.0071 in average MAP, which shows the robustness of the hybrid reranking model.

To understand the errors made by our model, we further evaluate the best model from each batch as indicated in Table 2 and report the performance of each question type and weighted average (WAVG) since the number of questions for each question type is different in each batch. There are four question types, namely factoid, list, Yes/No and summary. For factoid type questions, an entity name, a number or a short answer span is expected in the retrieved passages. For list type questions, a list of entity names, numbers or answer spans is expected in the retrieved passages. For Yes/No type questions, "Yes" or "No" answers are expected to be derived from the retrieved passages. For summary type questions, it contains questions that can not be categorized as the above three types. As we can see from Table 3, the performance for each type varies between different batches. In general, the set achieving higher recall always achieves higher MAP. When examining the set that has the worst performance, summary questions from Batch 1, factoid questions from Batch 2 and 4, Yes/No questions from Batch 3 and 5, and List questions from Batch 6, we notice that List type questions are more difficult than the other three types as knowledge aggregation from different documents is usually expected. From the example in Table 4 it shows that the expected passages should contain answer snippets that are names of specific tools to predict protein structure and they are from different documents. While the top retrieved passages except **P4** contain only high level descriptions instead of specific tools. In addition, we also note that for many questions, predictions are actually correct, but due to the annotation sparsity, they are counted as incorrect, especially for Factoid and Yes/No types questions (i.e., there are many false-negatives in the gold dataset). A metric that can evaluate whether the retrieved passages contain the answer spans may be more ideal for those two types of questions.

4. Conclusion

We presented our participating system in the 10th BioASQ challenge document retrieval subtask. We implemented a zero-shot two-stage hybrid model that includes a hybrid retrieval model consisting of BM25 and a dual encoder model, and a reranking model trained with the candidates retrieved from the first stage. Our hybrid retrieval model combined with T5-based reranking model outperforms best reporting system on test Batch 2, 4, and 6.

Table 4
An example of “List” type question in Batch 6 and top predictions from our model.

Question	Are there any tools that could predict protein structure considering amino acid sequence?
Answer Snippets	**A1:** PredictProtein (https://predictprotein.org) is a one-stop online resource for protein sequence analysis **A2:** Jpred (http://www.compbio.dundee.ac.uk/jpred) is a secondary structure prediction server **A3:** The recently updated Jnet algorithm provides a three-state (alpha-helix, beta-strand and coil) prediction of secondary structure at an accuracy of 81.5% **A4:** Porter 4.0 and PaleAle 4.0. Porter 4.0 predicts secondary structure correctly for 82.2% of residues **A5:** The PSIPRED Workbench is a web server offering a range of predictive methods
Predictions	**P1:** ... To deal with the overwhelming data, a collection of automated methods as bioinformatics tools which determine the structure of a protein from its amino acid sequence have emerged. The aim of this paper is to provide the experimental biologists with a set of cutting-edge, carefully evaluated, user-friendly computational tools for protein structure prediction ... **P2:** In principle, it is possible to predict theoretically the three-dimensional structure of a protein from its amino acid sequence. Recently substantial progress towards this goal has been made by the use of simple models to ... **P3:** Methods of predicting protein conformation from amino acid sequence are reviewed. Several widely used algorithms to predict local secondary structure are first discussed. Four general approaches to predict the tertiary structure are then described: ... Throughout this review, the likely success of these methods is considered. **P4:** Protein tertiary structure prediction algorithms aim to predict, from amino acid sequence, the tertiary structure of a protein. ... Here we briefly discuss protein tertiary structure prediction, the biennial competition for the Critical Assessment of Techniques for Protein Structure Prediction (CASP) and its role in shaping the field. We also discuss, in detail, our cutting-edge web-server method IntFOLD2-TS for tertiary structure prediction. Furthermore, we provide a step-by-step guide on using the IntFOLD2-TS web server....

References

[1] R. Nogueira, W. Yang, K. Cho, J. Lin, Multi-stage document ranking with BERT, 2019. URL: https://arxiv.org/abs/1910.14424.

[2] L. Gao, Z. Dai, J. Callan, Rethink training of BERT rerankers in multi-stage retrieval pipeline, in: Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 – April 1, 2021, Proceedings, Part II, 2021, p. 280–286. URL: https://doi.org/10.1007/978-3-030-72240-1_26.

[3] R. Pradeep, R. Nogueira, J. Lin, The expando-mono-duo design pattern for text ranking with pretrained sequence-to-sequence models, 2021. URL: https://arxiv.org/abs/2101.05667.

[4] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, Improving language understanding by generative pre-training, 2018. URL: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

[5] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language models are unsupervised multitask learners, 2019. URL: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

[6] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, in: Advances in neural information processing systems, volume 33, 2020, pp. 1877–1901. URL: https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[7] J. Devlin, M.-W. Chang, K. Lee, 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), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. URL: https://www.aclweb.org/anthology/N19-1423.

[8] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, Journal of Machine Learning Research 21 (2020) 1–67. URL: http://jmlr.org/papers/v21/20-074.html.

[9] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, J. Kang, BioBERT: a pre-trained biomedical language representation model for biomedical text mining, Bioinformatics 36 (2019) 1234–1240. URL: https://doi.org/10.1093/bioinformatics/btz682.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in neural information processing systems, Curran Associates, Inc., 2017, pp. 5998–6008. URL: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

[11] J. Ma, I. Korotkov, K. Hall, R. T. McDonald, Hybrid first-stage retrieval models for biomedical literature., in: Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum, 2020. URL: http://ceur-ws.org/Vol-2696/paper_92.pdf.

[12] J. Ma, I. Korotkov, Y. Yang, K. B. Hall, R. T. McDonald, Zero-shot neural passage retrieval via domain-targeted synthetic question generation, in: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, 2021, pp. 1075–1088. URL: https://aclanthology.org/2021.eacl-main.92/.

[13] C. Alberti, D. Andor, E. Pitler, J. Devlin, M. Collins, Synthetic QA corpora generation with roundtrip consistency, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 2019, pp. 6168–6173. URL: https://aclanthology.org/P19-1620.

[14] P. Lewis, Y. Wu, L. Liu, P. Minervini, H. Küttler, A. Piktus, P. Stenetorp, S. Riedel, PAQ: 65 million probably-asked questions and what you can do with them, Transactions of the Association for Computational Linguistics 9 (2021) 1098–1115. URL: https://aclanthology.org/2021.tacl-1.65. doi:10.1162/tacl_a_00415.

[15] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, et al., Natural questions: a benchmark for question answering research, Transactions of the Association for Computational Linguistics 7

(2019) 453–466. URL: https://aclanthology.org/Q19-1026.

[16] P. Rajpurkar, J. Zhang, K. Lopyrev, P. Liang, SQuAD: 100,000+ questions for machine comprehension of text, in: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Austin, Texas, 2016, pp. 2383–2392. URL: https://aclanthology.org/D16-1264.

[17] S. Han, X. Wang, M. Bendersky, M. Najork, Learning-to-rank with BERT in TF-Ranking, 2020. URL: https://arxiv.org/abs/2004.08476.

[18] R. Jagerman, X. Wang, H. Zhuang, Z. Qin, M. Bendersky, M. Najork, Rax: Composable learning-to-rank using JAX, in: Proceedings of the 28th ACM SIGKDD International Conference on Knowledge Discovery Data Mining, 2022. URL: https://storage.googleapis.com/pub-tools-public-publication-data/pdf/585d520959bb08dbb25b8dc60ff6aeb6eadc59dd.pdf.

[19] J. Ni, C. Qu, J. Lu, Z. Dai, G. H. Ábrego, J. Ma, V. Y. Zhao, Y. Luan, K. B. Hall, M.-W. Chang, Y. Yang, Large dual encoders are generalizable retrievers, 2021. URL: https://arxiv.org/abs/2112.07899.

[20] G. V. Cormack, C. L. A. Clarke, S. Buettcher, 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, SIGIR ’09, Association for Computing Machinery, New York, NY, USA, 2009, p. 758–759. URL: https://doi.org/10.1145/1571941.1572114.