

# Multi-stage Literature Retrieval System Trained by PubMed Search Logs for Biomedical Question Answering

Ashley Shin<sup>1</sup>, Qiao Jin<sup>1</sup> and Zhiyong Lu<sup>1</sup>

<sup>1</sup>National Library of Medicine (NLM), NIH

## Abstract

This paper discusses our submission to the 2023 BioASQ challenge, document retrieval subtask (subtask B, phase A). In the subtask, systems must return top 10 most relevant PubMed articles for each natural language query. Our multi-stage system incorporates a bi-encoder model in the retrieval stage, and a cross-encoder model at the reranking stage. At the retrieval stage, we use a hybrid retriever that combines dense and sparse retrieval, where the dense retrieval is implemented with the bi-encoder and the sparse retrieval is implemented with BM25. The bi-encoder and cross-encoder are initialized with PubMedBERT and further trained on PubMed query-article search logs of unprecedented scale, with about 255 million relevant query-article pairs. Each pair consists of a user query and a document clicked on by the user during their search on PubMed. Our system ranks second place in the second batch, and third place on the first and third batches at the 2023 BioASQ challenge, demonstrating the strength of models trained on PubMed search logs.

## Keywords

document retrieval, biomedical literature search, reranking, information retrieval, question answering, BioASQ

## 1. Introduction

Automatic question answering (QA) plays an important role in biomedical knowledge acquisition and clinical decision support [1]. For many retrieval-augmented QA systems, the first stage is to retrieve relevant documents for the given question. The BioASQ Biomedical Semantic QA (Task 11B), particularly the document retrieval subtask (Phase A), provides a way to evaluate QA systems on datasets built by biomedical experts [2]. We focus on the document retrieval aspect of the broader biomedical QA process. Lexical and semantic retrieval are two major approaches to textual information retrieval. Lexical retrieval is commonly implemented with sparse retrievers, while semantic retrieval is implemented with dense retrievers. Conventionally, document retrieval relied on sparse retrievers such as BM25 [3, 4], especially before the rise of transformer-based deep learning models [5]. Sparse retrievers, by design, are capable of efficient

---

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

✉ ashley.shin@nih.gov (A. Shin); qiao.jin@nih.gov (Q. Jin); zhiyong.lu@nih.gov (Z. Lu)


🌐 <https://www.ashleyshin.org> (A. Shin); <https://andy-jqa.github.io/> (Q. Jin);

<https://www.ncbi.nlm.nih.gov/research/bionlp/> (Z. Lu)

🆔 0000-0002-0531-9040 (A. Shin); 0000-0002-1268-7239 (Q. Jin); 0000-0001-9998-916X (Z. Lu)



© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

lexical term matching but are weak at semantic matching due to the vocabulary mismatch problem [6, 7].

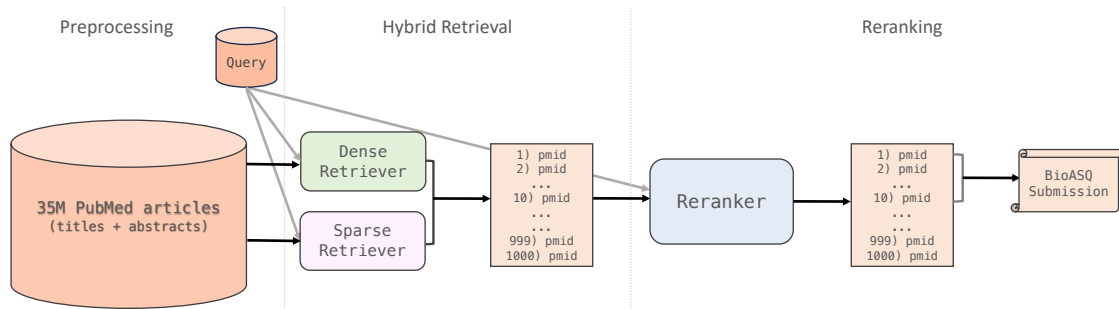
In this regard, the recent developments in dense retrieval led to great performance improvements over traditional sparse retrieval [8, 9]. However, dense retrievers have their drawbacks. Unlike sparse retrievers, dense retrievers often underperform on datasets that are outside the distributions of the training dataset [10], requiring large amounts of data and compute to train, in addition to longer latency at inference time. Thus, term-based sparse retrieval still has a place in document retrieval [11, 12]. A lexical retriever is much more cost-efficient and practical at scale, so it is still widely used for early retrieval stages. And while transformer architectures are good at capturing semantic meaning, they often struggle with lexical matching [13].

Both lexical and semantic retrieval models have clear strengths and weaknesses. Building on this idea, one prominent research direction is incorporating different types of retrievers in an attempt to combine their strengths [14, 15, 16]. In particular, combining sparse retrievers and BERT-based bi-encoders has achieved notable results [17, 18]. Especially for zero-shot and out-of-domain tasks, hybrid models that incorporate both dense and sparse retrieval outperform dense retriever models alone. In addition to the retrieval stage, multi-stage document retrieval systems have further reranking stages. Sparse retrievers and bi-encoders are often used for early-stage retrieval. For reranking, a popular reranking architecture is the cross-encoder [19]. Cross-encoders generally outperform bi-encoders – albeit much slower during inference – making cross-encoders apt for reranking stages, where there are fewer candidate documents to rank [8].

In this work, we present a multi-stage retrieval system, where the first stage retrieval uses a hybrid approach, consisting of a dense retriever and a sparse retriever. In the second stage, we rerank with a cross-encoder model. Our contribution is that we pre-train both the dense retriever and the cross-encoder reranker on large-scale PubMed search logs, which contain 255 million relevant article-article pairs [20] and have not been used by any prior work. Figure 1 summarizes our overall architecture.

In the retrieval stage, we use BM25 and a bi-encoder. Since BM25 is a static model that is not trained with data, improving the dense retrieval component is essential to improving the hybrid retriever overall. The PubMed articles are processed into a Pyserini index [21] and vector embeddings for sparse and dense retrieval, respectively. The hybrid retriever selects the top 1000 most relevant articles for each query. Then, the scores of the retrieved list are discarded and just the ranks are fed to the cross-encoder for the reranking stage. The reranker returns an updated top 1000 list. The top 10 of the list are submitted for the BioASQ challenge.

We evaluate our results both amongst different variants of our system (e.g., with differing hyperparameter values) and against other high-performing systems by participating in the 2023 BioASQ challenge, document retrieval task [22]. We place in the top 3 for the first and the third batches, and top 2 in the second batch, demonstrating the strength of our bi-encoder and cross-encoder, and thus the value of training models with user query-click logs [23, 24, 25].



**Figure 1:** Outline of the overall system; the pipeline has 3 major parts – preprocessing, retrieval, and reranking – which are separated by the dotted vertical lines. The PubMed corpus is preprocessed and filtered to only include articles with both a title and an abstract. The hybrid retriever consists of a dense retriever and a sparse retriever, and returns an initial list of 1000 candidate documents, which is further reranked by the reranker. The top 10 documents of the final returned list is submitted as part of the BioASQ challenge.

## 2. Hybrid Retrieval

**Dataset** We use query-article pairs from PubMed search logs to train the Bi-Encoder [20]. Each unique query is associated with a list of articles. Each time a user searches for a query on PubMed and clicks on an article in their search results counts as a click count for that query-article pair. To build the training set, we start with raw PubMed search logs from 2020 to 2022. The raw logs consist of 168 million (M) unique queries and 24M unique articles. There are multiple stages of filtering, including removing navigational queries. Articles that don't have a title or an abstract are also excluded. After preprocessing, the logs are reduced to 255M relevant query-article pairs, generated from 87M queries and 18M articles.

**Dense Retrieval with Bi-Encoder** To implement the dense retrieval portion of the hybrid retriever architecture, we use a bi-encoder model. The dense retriever (DR) is a bi-encoder comprised of a query encoder and a document encoder, both initialized with PubMedBERT-base weights [27]. In a bi-encoder architecture, two independent encoders encode a query and a document [28, 29]. The relevance score is then the dot product of the two embeddings. The relevance scores are used to rank the most relevant documents.

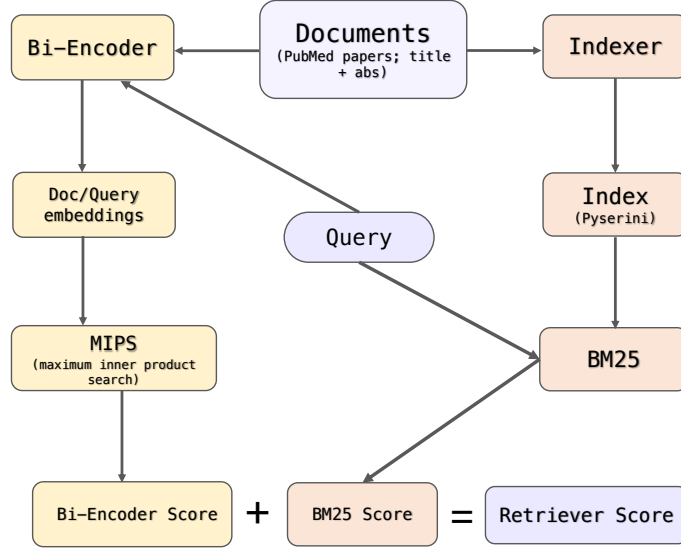
DR is comprised of a query encoder (QEnc) and a document encoder (DEnc). They are both 12-layer transformer architectures and are implemented with Pytorch [30] and Hugging Face Transformers library [31]. Given a query  $q$  and a document  $d$ , the relevance score is computed by:

$$DR(q, d) = \text{QEnc}(q)^T \text{DEnc}(d) \in \mathbb{R}$$

QEnc and DEnc are as follows:

$$\text{QEnc}(q) = \text{Transformer}([\text{CLS}]q[\text{SEP}])_{[\text{CLS}]} \in \mathbb{R}^{768}$$

$$\text{DEnc}(d) = \text{Transformer}([\text{CLS}]d_{\text{title}}[\text{SEP}]d_{\text{abstract}}[\text{SEP}])_{[\text{CLS}]} \in \mathbb{R}^{768}$$



**Figure 2:** Overview of the hybrid retriever. The yellow boxes represent dense retrieval, and orange denotes sparse retrieval. The scores from each retriever is summed to generate a final score for each candidate document. The output of the hybrid retriever are the candidate documents, ranked by their final scores. We train our own bi-encoder for dense retrieval, and use BM25 for sparse retrieval. FAISS [26] is used to perform nearest neighbor search for document and query embeddings, and Pyserini [21] is used to implement BM25.

Embeddings are taken from the last [CLS] hidden states. We use contrastive loss with in-batch negatives [28]. For a training instance containing a query  $q$  and clicked document  $d$ , query-to-document loss  $Loss_{q2d}(q, d)$  and document-to-query loss  $Loss_{d2q}(q, d)$  are described by:

$$Loss_{q2d}(q, d) = -\log \frac{\exp(\text{QEnc}(q)^T \text{DEnc}(d))}{\sum_{i \in B} \exp(\text{QEnc}(q)^T \text{DEnc}(d_i))}$$

$$Loss_{d2q}(q, d) = -\log \frac{\exp(\text{QEnc}(q)^T \text{DEnc}(d))}{\sum_{i \in B} \exp(\text{QEnc}(q_i)^T \text{DEnc}(d))}$$

where  $B$  denotes all documents from the mini-batch except the input training instance.

Batch-level losses,  $Loss_{q2d}^B$  and  $Loss_{d2q}^B$  are as follows:

$$Loss_{q2d}^B = \sum_{i \in B} W_i Loss_{q2d}(q_i, d_i)$$

$$Loss_{d2q}^B = \sum_{i \in B} W_i Loss_{d2q}(q_i, d_i)$$

where  $W_i$  is a parameter that gives higher weight to instances with more clicks. The final loss  $Loss_B$  for the mini-batch is then:

$$Loss_B = \alpha Loss_{q2d}^B + (1 - \alpha) Loss_{d2q}^B$$

where  $\alpha$  is a trained hyper-parameter. We optimize parameters in QEnc and DEnc end-to-end by gradient-based optimizers. The models use the Adam optimizer [32] without weight decay, with learning rate of  $2e-5$  and epsilon  $1e-8$ . We encode all queries and documents using QEnc and DEnc, respectively. We use the FAISS library to build an index and perform nearest N search [26]. The query and document embeddings are added to a FlatIP FAISS index, and maximum inner-product search is used to generate the a list of 1000 most relevant documents for each query.

**Sparse Retrieval with BM25** Sparse retrievers such as BM25 continue to be quite useful despite the recent rise of dense retrieval methods [33]. BM25 frequently serves as a nice baseline, but it can also be used to complement dense retrievers’ weaknesses, such as lexical matching. We use the Pyserini library to implement BM25 [21].

**Hybrid Retriever** Our retriever combines results from the dense retriever and the sparse retriever. The document scores returned by the retrievers are normalized with L1 norm. A hyperparameter  $\lambda$  controls the weight of the BM25 scores in the final summed scores:

$$\text{Score}(q, d) = \tilde{S}_{dr}(q, d) + \lambda \tilde{S}_{sr}(q, d)$$

where  $\text{Score}(q, d)$  is the final hybrid retrieval score for some query  $q$  and some document  $d$ ,  $\tilde{S}_{dr}$  the L1-normed score from the dense retriever, and  $\tilde{S}_{sr}(q, d)$  the L1-normed score from the sparse retriever. The final output list of the hybrid retriever for some query is simply the top 1000 relevant documents, ranked by the above summed scores.

### 3. Reranking with Cross-Encoder

The top 1000 list returned by the hybrid retriever is then reranked by our cross-encoder model. Unlike the bi-encoder, a single cross-encoder model (CrossEnc) is fed both the query and document, and predicts their relevance directly, without having to use nearest neighbors search. CrossEnc, like QEnc and DEnc, is initialized with PubMedBERT weights [27], and implemented with Pytorch [30] and Hugging Face Transformers libraries [31]. After filtering and preprocessing from the original 35 million PubMed papers, we feed the concatenations to (CrossEnc). Each instance  $d$  corresponding to a paper is formatted as [title]+[abstract]. The overall input to the CrossEnc is [CLS] $q$ [SEP] $d$ [SEP], as in:

$$\text{CrossEnc}(fcvq, d) = W^T \text{Transformer}([\text{CLS}]q[\text{SEP}]d[\text{SEP}])_{[\text{CLS}]} + b \in \mathbb{R}$$

where [CLS] and [SEP] are special tokens in BERT,  $W \in \mathbb{R}^{768}$  and  $b \in \mathbb{R}$  are learned paramters, and  $q$  denotes query. Unlike QEnc and DEnc, which are trained with in-batch negatives, CrossEnc uses local negatives [34]. For a training instance with a query  $q$  and a relevant (clicked) document  $d$ , the negative log-likelihood loss is:

$$\text{Loss}(q, d) = -\log \frac{\exp(\text{CrossEnc}(q, d))}{\exp(\text{CrossEnc}(q, d)) + \sum_{j \in M} \exp(\text{CrossEnc}(q, d_j))}$$

where  $M$  is a list of documents that are irrelevant (not clicked).

We pre-train the CrossEnc on PubMed search log data, and further train using the training set provided by BioASQ, which consist of over 4000 queries [35]. As for inference, we apply CrossEnc to every document in the list returned by the hybrid retriever for each test query, and the list of documents ranked by CrossEnc scores is the reranked list. Each team can submit up to 5 systems for official evaluation. For submission, we use two different versions of CrossEnc – one at 1200 optimization steps, and another at 1800.

## 4. Evaluation

The official BioASQ evaluation offers a way to compare our systems against other high performing systems. In the official BioASQ 2023 results, our systems rank at third, second, and third place for batches 1-3, respectively. We experiment with many different combinations of hyperparameters, which are shown in Table 1. The official results are on Table 2. Finally, we perform additional experiments on an internal development set for ablation purposes, shown on Table 3.

### 4.1. Configurations

Table 1 shows the various systems we submitted for official evaluation. Retriever1 and Retriever2 are hybrid retrievers combining our bi-encoder and BM25. The lambda values show how much weight was given to the BM25 scores in producing the overall retriever score. These are not reranked. Rerankers 1 through 5 represent submissions that are reranked by our cross-encoder after the initial retrieval by our hybrid retriever. We tried two different versions of the cross-encoder – which are checkpoints saved at 1200 and 1800 steps. For example, Ranker3 starts with the list returned by the hybrid retriever with lambda value of 1.25, and reranks with the cross-encoder checkpoint at 1800 steps.

**Table 1**

System names and their respective hyperparameters, for Table 2.  $\lambda_{BM25}$  refers to the hyperparameter for controlling the weight of sparse retrieval at the retrieval stage. Steps are optimization steps at which the checkpoint of the cross-encoder was saved. Retrievers 1 and 2 are retriever only, while Rerankers 1-5 use both retriever and reranker. For example, Reranker3 takes the results from Retriever2, and reranks with the cross-encoder checkpoint at 1800 steps.

System Name	$\lambda_{BM25}$	Steps
Retriever 1	1	N/A
Retriever 2	1.25	N/A
Reranker 1	1	1800
Reranker 2	1	1200
Reranker 3	1.25	1800
Reranker 4	2	1200
Reranker 5	0.6	1800

## 4.2. Official BioASQ Evaluation

Teams can submit up to 5 systems for official evaluation in the challenge [2]. Table 2 indicates which 5 systems we use for each batch. Retriever2 outperforms Retriever1 in batch 1, but Retriever1 outperforms Retriever2 in batch 2 by 66%. Retriever2 scores 0.3401 in batch 1 but 0.1725 in batch 2. The performance differences both among the two retrievers and among the first two batches is quite large. There is high variance amongst the batches which highly influences the two retrievers’ performance, considering that there’s only a 0.25  $\lambda_{BM25}$  difference between them. Nevertheless, it is clear that the rerankers consistently outperform the retriever-only systems. For example, the best reranker outperforms the best retriever by 29% and 30% on batches 1 and 2 respectively, which demonstrates the strength of our cross-encoder trained on PubMed search logs. And accordingly, we no longer submit retriever-only systems for batch 3.

Rerankers 1, 3, and 5 are reranked with cross-encoder checkpoint at 1800 steps, but Reranker1 and Reranker3 outperform Reranker5 by 24%. The only difference is that Reranker5 has a significantly lower  $\lambda_{BM25}$  of 0.6, as opposed to 1 or 1.25, showing that the hyperparameter weight of the sparse retriever versus dense retriever in the hybrid retriever can have a large impact. We use two different versions of the reranker (1200 and 1800) in an attempt to capture different aspects of the test data, which is only somewhat successful. In batch 1, Reranker1 and Reranker2 have the same MAP score despite using different versions of the cross-encoder.

In batch 3, Reranker2 and Reranker4, which use the 1200 version, outperform Reranker1 and Reranker3, which use the 1800 version, by about 1%. In comparison with other top teams, our systems fare pretty well. In batch 1, our top system scores 0.4404, comparable to the top score of 0.4590. In batch 2, the top system outperforms our best system by 3%. We again rank top 3 in batch 3.

**Table 2**

Mean average precision (MAP) official evaluation results from 2023 BioASQ task B phase A, batches 1-3. ”-” denotes systems that weren’t submitted for that batch for official evaluation. Bolded are the best performing amongst our submitted systems.

Systems	Batch 1	Batch 2	Batch 3
Retriever 1	0.2569	0.2860	-
Retriever 2	0.3401	0.1725	-
Reranker 1	<b>0.4404</b>	<b>0.3743</b>	0.2905
Reranker 2	<b>0.4404</b>	-	<b>0.2939</b>
Reranker 3	0.4397	<b>0.3743</b>	0.2902
Reranker 4	-	-	<b>0.2939</b>
Reranker 5	-	0.3014	0.2712
1st Place System	0.4590	0.3852	0.3185
2nd Place System	0.4462	<b>(ours)</b>	0.3042
3rd Place System	<b>(ours)</b>	0.3720	<b>(ours)</b>



### 4.3. Further Analysis

We randomly select about 200 queries from the BioASQ training dataset to form a development set [35]. We perform ablation experiments on the development set to further test the merit of our design choices. The results are summarized in Table 3. We evaluate the systems on the development set with three metrics: mean average precision (MAP), recall at 10 (R@10), and recall at 1000 (R@1k). We test BM25 (sparse retrieval only), Bi-Encoder (dense retrieval only), Hybrid Retriever, Reranker, and systems that combine the normed scores from Reranker and the previous 3 systems (i.e. BM25, Bi-Encoder, Hybrid Retriever). The plus sign in table 3 indicates that the normed scores from the "operand" systems were summed to generate the final score for the system.

The trends show the relative strengths of BM25 and BERT-based models. BM25 and Reranker2 + BM25 show strong performance for R@1k, perhaps showing that sparse retrieval is effective at recall but not as strong in precision, especially for low ranks. Hybrid Retriever outperforms BM25 and Bi-Encoder in MAP and R@10, justifying the case for hybrid retrieval. And although Bi-Encoder on its own performs poorly, it still improves retrieval when incorporated. In addition, Reranker outperforms Hybrid Retriever, confirming the importance of reranking and justifying the need for multi-stage systems. Finally, we test the possibility that reranking, similar to retrieval, also benefits from hybrid models – i.e. that the reranker might gain from augmentation with either BM25, Bi-Encoder, or Hybrid Retriever. The experiments show that our cross-encoder reranker is just as effective or better on its own as it is when combined with another retriever into a "hybrid" version, with the exception of Reranker + BM25 in R@1k, which further confirms the aforementioned strength of BM25 in recall. Overall, our internal evaluations further demonstrate the effectiveness of hybrid retrieval and multi-stage ranking systems.

**Table 3**

Results of ablation experiments on our development set, which is a subset of about 200 queries from the BioASQ training set [35]. MAP denotes mean average precision, R@10 is recall at 10, and R@1k is recall at 1000. Bold denotes the best performing system(s) with respect to the metric. The plus sign indicates that the ranks are scored by summing the normed scores from the "operands." Hybrid Retriever and Reranker in this table are the same systems as Retriever 1 and Reranker 2 from table 2, respectively.

System Name	MAP	R@10	R@1k
BM25 (Sparse Retriever)	0.235	0.331	0.806
Bi-Encoder (Dense Retriever)	0.112	0.183	0.636
Hybrid Retriever (BM25 + Bi-Encoder)	0.235	0.340	0.793
Reranker	<b>0.319</b>	<b>0.416</b>	0.793
Reranker + BM25	0.306	0.405	<b>0.826</b>
Reranker + Bi-Encoder	0.312	0.408	0.778
Reranker + Hybrid Retriever	<b>0.319</b>	<b>0.416</b>	0.793



## 5. Conclusion

We present a multi-stage system with a hybrid retriever and a reranker. The hybrid retriever is implemented with a bi-encoder model and BM25, and the reranker is a cross-encoder model. Our system rank at third, second, and third place for batches 1-3 respectively in the 2023 BioASQ challenge, document retrieval subtask, showing the strength of training on PubMed search logs. As for potential future directions, one simple addition would be to use reciprocal rank fusion to combine different versions of the trained reranker [36]. Another promising approach is to use large language models such as ChatGPT<sup>1</sup> as the reranker [37, 38, 39].

## Acknowledgments

This research was supported by the Intramural Research Program of the National Library of Medicine (NLM), National Institutes of Health.

## References

- [1] Q. Jin, Z. Yuan, G. Xiong, Q. Yu, H. Ying, C. Tan, M. Chen, S. Huang, X. Liu, S. Yu, Biomedical question answering: A survey of approaches and challenges, *ACM Comput. Surv.* 55 (2022). URL: <https://doi.org/10.1145/3490238>. doi:10.1145/3490238.
- [2] A. Nentidis, A. Krithara, G. Paliouras, E. Farre-Maduell, S. Lima-Lopez, M. Krallinger, Bioasq at clef2023: The eleventh edition of the large-scale biomedical semantic indexing and question answering challenge, in: *Advances in Information Retrieval: 45th European Conference on Information Retrieval, ECIR 2023, Proceedings, Part III, 2023*, p. 577–584. doi:10.1007/978-3-031-28241-6\_66.
- [3] S. E. Robertson, S. Walker, Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval, in: *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1994.
- [4] S. E. Robertson, H. Zaragoza, The probabilistic relevance framework: BM25 and beyond, *Found. Trends Inf. Retr.* 3 (2009) 333–389. URL: <https://doi.org/10.1561/1500000019>. doi:10.1561/1500000019.
- [5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, I. Polosukhin, Attention is all you need, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), *Advances in Neural Information Processing Systems*, volume 30, Curran Associates, Inc., 2017. URL: [https://proceedings.neurips.cc/paper\\_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf).
- [6] A. Berger, R. Caruana, D. Cohn, D. Freitag, V. Mittal, Bridging the lexical chasm: Statistical approaches to answer-finding, in: *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '00*, Association for Computing Machinery, New York, NY, USA, 2000, p. 192–199. URL: <https://doi.org/10.1145/345508.345576>. doi:10.1145/345508.345576.

---

<sup>1</sup><https://openai.com/blog/chatgpt>

- [7] H. Fang, C. Zhai, Semantic term matching in axiomatic approaches to information retrieval, in: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '06, Association for Computing Machinery, New York, NY, USA, 2006, p. 115–122. URL: <https://doi.org/10.1145/1148170.1148193>. doi:10.1145/1148170.1148193.
- [8] S. Humeau, K. Shuster, M.-A. Lachaux, J. Weston, Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring, in: International Conference on Learning Representations, 2020. URL: <https://openreview.net/forum?id=SkxgmnNFvH>.
- [9] L. Soldaini, A. Moschitti, The cascade transformer: an application for efficient answer sentence selection, in: Annual Meeting of the Association for Computational Linguistics, 2020.
- [10] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, I. Gurevych, Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models, in: J. Vanschoren, S. Yeung (Eds.), Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks, volume 1, Curran, 2021. URL: [https://datasets-benchmarks-proceedings.neurips.cc/paper\\_files/paper/2021/file/65b9eea6e1cc6bb9f0cd2a47751a186f-Paper-round2.pdf](https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/65b9eea6e1cc6bb9f0cd2a47751a186f-Paper-round2.pdf).
- [11] J. Guo, Y. Fan, Q. Ai, W. B. Croft, A deep relevance matching model for ad-hoc retrieval, Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (2016).
- [12] X. Ma, K. Sun, R. Pradeep, M. Li, J. Lin, Another look at dpr: Reproduction of training and replication of retrieval, in: European Conference on Information Retrieval, 2022.
- [13] X. Chen, K. Lakhotia, B. Oğuz, A. Gupta, P. Lewis, S. Peshterliev, Y. Mehdad, S. Gupta, W. tau Yih, Salient phrase aware dense retrieval: Can a dense retriever imitate a sparse one?, in: Conference on Empirical Methods in Natural Language Processing, 2021.
- [14] M. Seo, J. Lee, T. Kwiatkowski, A. Parikh, A. Farhadi, H. Hajishirzi, Real-time open-domain question answering with dense-sparse phrase index, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Florence, Italy, 2019, pp. 4430–4441. URL: <https://aclanthology.org/P19-1436>. doi:10.18653/v1/P19-1436.
- [15] L. Gao, Z. Dai, Z. Fan, J. Callan, Complementing lexical retrieval with semantic residual embedding, CoRR abs/2004.13969 (2020). URL: <https://arxiv.org/abs/2004.13969>. arXiv:2004.13969.
- [16] Y. Luan, J. Eisenstein, K. Toutanova, M. Collins, Sparse, dense, and attentional representations for text retrieval, Transactions of the Association for Computational Linguistics 9 (2021) 329–345. URL: <https://aclanthology.org/2021.tacl-1.20>. doi:10.1162/tacl\_a\_00369.
- [17] J. Lu, J. Ma, K. Hall, Zero-shot hybrid retrieval and reranking models for biomedical literature (2022).
- [18] T. Chen, M. Zhang, J. Lu, M. Bendersky, M. Najork, Out-of-domain semantics to the rescue! zero-shot hybrid retrieval models, in: Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part I, Springer, 2022, pp. 95–110.
- [19] R. F. Nogueira, K. Cho, Passage re-ranking with BERT, CoRR abs/1901.04085 (2019). URL:

<http://arxiv.org/abs/1901.04085>. arXiv:1901.04085.

- [20] R. I. Dogan, G. C. Murray, A. Névél, Z. Lu, Understanding pubmed® user search behavior through log analysis, *Database: The Journal of Biological Databases and Curation* 2009 (2009).
- [21] J. J. Lin, X. Ma, S.-C. Lin, J.-H. Yang, R. Pradeep, R. Nogueira, D. R. Cheriton, Pyserini: A python toolkit for reproducible information retrieval research with sparse and dense representations, *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2021).
- [22] G. Tsatsaronis, G. Balikas, P. Malakasiotis, I. Partalas, M. Zschunke, M. R. Alvers, D. Weisenborn, A. Krithara, S. Petridis, D. Polychronopoulos, Y. Almirantis, J. Pavlopoulos, N. Baskiotis, P. Gallinari, T. Artières, A.-C. N. Ngomo, N. Heino, É. Gaussier, L. Barrio-Alvers, M. Schroeder, I. Androutsopoulos, G. Paliouras, An overview of the bioasq large-scale biomedical semantic indexing and question answering competition, *BMC Bioinformatics* 16 (2015).
- [23] Q. Jin, W. Kim, Q. Chen, D. C. Comeau, L. Yeganova, J. Wilbur, Z. Lu, Biocpt: Contrastive pre-trained transformers with large-scale pubmed search logs for zero-shot biomedical information retrieval, 2023. arXiv:2307.00589.
- [24] Q. Jin, A. Shin, Z. Lu, Lader: Log-augmented dense retrieval for biomedical literature search, 2023. arXiv:2304.04590.
- [25] N. Rekabsaz, O. Lesota, M. Schedl, J. Brassey, C. Eickhoff, Tripclick: The log files of a large health web search engine, in: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '21*, Association for Computing Machinery, New York, NY, USA, 2021, p. 2507–2513. URL: <https://doi.org/10.1145/3404835.3463242>. doi:10.1145/3404835.3463242.
- [26] J. Johnson, M. Douze, H. Jégou, Billion-scale similarity search with GPUs, *IEEE Transactions on Big Data* 7 (2019) 535–547.
- [27] Y. Gu, R. Tinn, H. Cheng, M. Lucas, N. Usuyama, X. Liu, T. Naumann, J. Gao, H. Poon, Domain-specific language model pretraining for biomedical natural language processing, *ACM Transactions on Computing for Healthcare (HEALTH)* 3 (2021) 1–23.
- [28] V. Karpukhin, B. Oguz, S. Min, P. S. H. Lewis, L. Wu, S. Edunov, D. Chen, W. Yih, Dense passage retrieval for open-domain question answering, in: B. Webber, T. Cohn, Y. He, Y. Liu (Eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*, Online, November 16-20, 2020, Association for Computational Linguistics, 2020, pp. 6769–6781. URL: <https://doi.org/10.18653/v1/2020.emnlp-main.550>. doi:10.18653/v1/2020.emnlp-main.550.
- [29] L. Xiong, C. Xiong, Y. Li, K.-F. Tang, J. Liu, P. N. Bennett, J. Ahmed, A. Overwijk, Approximate nearest neighbor negative contrastive learning for dense text retrieval, in: *International Conference on Learning Representations*, 2021. URL: <https://openreview.net/forum?id=zeFrfgyZln>.
- [30] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-performance deep learning library, *CoRR abs/1912.01703* (2019). URL: <http://arxiv.org/abs/1912.01703>. arXiv:1912.01703.

- [31] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, A. Rush, Transformers: State-of-the-art natural language processing, in: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Association for Computational Linguistics, Online, 2020, pp. 38–45. URL: <https://aclanthology.org/2020.emnlp-demos.6>. doi:10.18653/v1/2020.emnlp-demos.6.
- [32] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, *CoRR* abs/1412.6980 (2014).
- [33] J. Lin, The neural hype and comparisons against weak baselines, *SIGIR Forum* 52 (2019) 40–51. doi:10.1145/3308774.3308781.
- [34] 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*, Springer-Verlag, Berlin, Heidelberg, 2021, p. 280–286. URL: [https://doi.org/10.1007/978-3-030-72240-1\\_26](https://doi.org/10.1007/978-3-030-72240-1_26). doi:10.1007/978-3-030-72240-1\_26.
- [35] A. Krithara, A. Nentidis, K. Bougiatiotis, G. Paliouras, BioASQ-QA: A manually curated corpus for Biomedical Question Answering, *Scientific Data* 10 (2023) 170.
- [36] 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. doi:10.1145/1571941.1572114.
- [37] W. Sun, L. Yan, X. Ma, P. Ren, D. Yin, Z. Ren, Is chatgpt good at search? investigating large language models as re-ranking agent, *arXiv preprint arXiv:2304.09542* (2023).
- [38] S. Tian, Q. Jin, L. Yeganova, P.-T. Lai, Q. Zhu, X. Chen, Y. Yang, Q. Chen, W. Kim, D. C. Comeau, et al., Opportunities and challenges for chatgpt and large language models in biomedicine and health, *arXiv preprint arXiv:2306.10070* (2023).
- [39] Q. Jin, R. Leaman, Z. Lu, Retrieve, summarize, and verify: How will chatgpt impact information seeking from the medical literature?, *Journal of the American Society of Nephrology* (2023) 10–1681.