LADER: Log-Augmented DEnse Retrieval for Biomedical Literature Search

Qiao Jin
National Library of Medicine
National Institutes of Health
Bethesda, Maryland, USA
qiao.jin@nih.gov

Ashley Shin
National Library of Medicine
National Institutes of Health
Bethesda, Maryland, USA
ashley.shin@nih.gov

Zhiyong Lu
National Library of Medicine
National Institutes of Health
Bethesda, Maryland, USA
zhiyong.lu@nih.gov

ABSTRACT

Queries with similar information needs tend to have similar document clicks, especially in biomedical literature search engines where queries are generally short and top documents account for most of the total clicks. Motivated by this, we present a novel architecture for biomedical literature search, namely Log-Augmented DEnse Retrieval (LADER), which is a simple plug-in module that augments a dense retriever with the click logs retrieved from similar training queries. Specifically, LADER finds both similar documents and queries to the given query by a dense retriever. Then, LADER scores relevant (clicked) documents of similar queries weighted by their similarity to the input query. The final document scores by LADER are the average of (1) the document similarity scores from the dense retriever and (2) the aggregated document scores from the click logs of similar queries. Despite its simplicity, LADER achieves new state-of-the-art (SOTA) performance on TripClick, a recently released benchmark for biomedical literature retrieval. On the frequent ("HEAD") queries, LADER largely outperforms the best retrieval model by 39% relative NDCG@10 (0.338 v.s. 0.243). LADER also achieves better performance on the less frequent ("TORSO") queries with 11% relative NDCG@10 improvement over the previous SOTA (0.303 v.s. 0.272). On the rare ("TAIL") queries where similar queries are scarce, LADER still compares favorably to the previous SOTA method (NDCG@10: 0.310 v.s. 0.295). On all queries, LADER can improve the performance of a dense retriever by 24%-37% relative NDCG@10 while not requiring additional training, and further performance improvement is expected from more logs. Our regression analysis has shown that queries that are more frequent, have higher entropy of query similarity and lower entropy of document similarity, tend to benefit more from log augmentation.

CCS CONCEPTS

• **Information systems → Language models**; • **Computing methodologies → Natural language processing**; • **Applied computing → Life and medical sciences**.

KEYWORDS

TripClick; biomedical literature search; dense retrieval

SIGIR '23, July 23–27, 2023, Taipei, Taiwan

ACM ISBN 978-1-4503-9408-6/23/07.
https://doi.org/10.1145/3539618.3592005

ACM Reference Format:
Qiao Jin, Ashley Shin, and Zhiyong Lu. 2023. LADER: Log-Augmented DEnse Retrieval for Biomedical Literature Search. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3539618.3592005

1 INTRODUCTION

Biomedical literature search is an essential step for knowledge discovery and clinical decision support [8, 12, 17]. It has several distinct properties from other information retrieval (IR) tasks: (1) Most queries are short. The average length of a query is about 3.5 tokens in PubMed[1], a widely used biomedical literature search engine [7, 9, 10]; (2) Large-scale relevant query-document pairs can be easily collected from the user click logs. TripClick [29], a recently released benchmark for biomedical literature retrieval, contains 1.3 million query-document relevance signals collected from the Trip Database[2] logs; (3) Users mostly browse the documents on the first page [7], and the top 31% most clicked documents account for 80% clicks in the released Trip Database logs. These unique characteristics motivate the augmentation of biomedical literature search by directly retrieving from the click logs of similar queries, since queries with similar information needs tend to have similar document clicks [1, 38, 40]. This approach is essentially similar to recent retrieval augmentation methods [11, 20, 39]. Fig 1 shows an example query and clicked documents from its similar queries in the training set of TripClick. The objective of this work is to augment biomedical literature search with such documents.

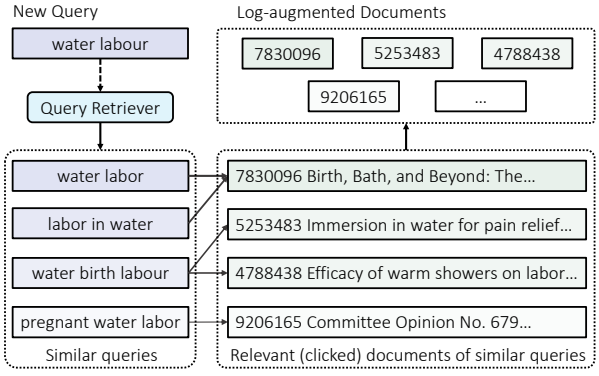

Figure 1: An example of using click logs of retrieved similar queries to augment biomedical literature search.

[1]https://pubmed.ncbi.nlm.nih.gov/
[2]https://www.tripdatabase.com/

One model can simply return clicked documents from the logs of similar queries as the retrieval results, which is analogous to memory-based collaborative filtering in recommendation systems [33]. However, this might miss relevant documents that have not been clicked before. Therefore, clicked documents from logs are better used to augment an existing retriever, and they are theoretically retriever-agnostic. We choose to use dense retrievers, where queries and documents are encoded and matched in low-dimensional embedding space, because recent work has shown that dense retrievers based on pre-trained language models such as BERT [6] outperform traditional sparse retrievers on various tasks [19, 21, 37].

In this paper, we introduce Log-Agumented DEnse Retrieval (LADER), which is a novel and simple plug-in module that augments dense retrievers by interpolating the document scores aggregated from logs of similar queries and the original scores from a dense retriever. We conduct experiments on the recently introduced TripClick benchmark [29] and show that: (1) LADER outperforms previous state-of-the-art (SOTA) models on queries of all frequency groups, with close to 40% relative NDCG@10 improvement on the frequent ("HEAD") queries; (2) LADER improves the backbone dense retriever by 24%-37% relative NDCG@10 while not requiring additional training, and it is expected to have further performance improvements with more logs. Our regression analysis shows that queries with higher frequency, lower entropy of similar query scores, and higher entropy of similar document scores, tend to benefit more from log augmentation.

2 METHODS

We describe the training of our backbone dense retriever in §2.1, and introduce how to do inference with the LADER module in §2.2.

2.1 Dense Retriever Training

We train a dense retriever that contains a query encoder (QEnc) and a document encoder (DEnc), both are 12-layer transformer (Trm) encoders [34]. The encoders are initialized with PubMedBERT-base [13], which is a biomedical domain-specific BERT model. During training, each instance contains a triple of a query q, a relevant document d^+ for the query, and a non-relevant document d^- for the query. They are fed to their respective encoders and the last [CLS] hidden states are used as their embeddings:

$$E(q) = \text{QEnc}(q) = \text{Trm}([\text{CLS}]\ q\ [\text{SEP}])_{[\text{CLS}]}$$

$$E(d) = \text{DEnc}(d) = \text{Trm}([\text{CLS}]\ d_{\text{title}}\ [\text{SEP}]\ d_{\text{abstract}}\ [\text{SEP}])_{[\text{CLS}]}$$

where [CLS] and [SEP] are special tokens used in BERT.

We optimize the model with a combination of two loss functions: an in-batch negative log-likelihood loss $\mathcal{L}^i$ [19] and a triplet contrastive loss $\mathcal{L}^t$ [32]. The in-batch negative log-likelihood loss helps the dense retriever distinguish between positive documents and all other documents in the mini-batch:

$$\mathcal{L}^i = -\log \frac{\exp(E(q)^T E(d^+))}{\sum_{i \in B} \exp(E(q)^T E(d_i))}$$

where B denotes all documents from the mini-batch. The triplet loss further contrasts positive documents and hard negative documents:

$$\mathcal{L}^t = \max(0, \text{Dist}(E(q), E(d^+)) - \text{Dist}(E(q), E(d^-) + \alpha)$$

where Dist denotes a distance metric and α is the target margin between the positive and negative pairs.

The final loss is a weighted sum of the two loss functions:

$$\mathcal{L} = \beta \mathcal{L}^i + (1 - \beta)\mathcal{L}^t$$

where β is a hyper-parameter for loss weighting. We train parameters in QEnc and DEnc end-to-end by gradient-based optimizers.

2.2 LADER Inference

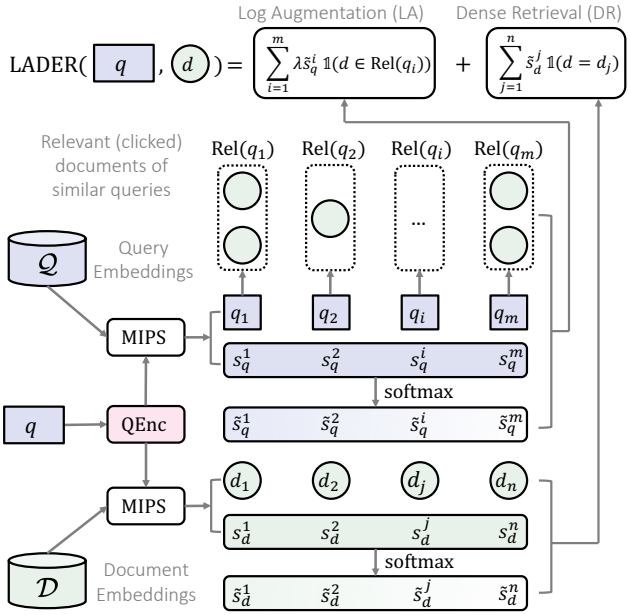

Figure 2: Overall architecture of the LADER model.

The overall architecture of LADER is shown in Figure 2. We use the trained QEnc to encode all queries in the training dataset and use the trained DEnc to encode all documents in the collection, getting $\mathcal{Q} \in \mathbb{R}^{N_q \times 768}$ and $\mathcal{D} \in \mathbb{R}^{N_d \times 768}$, respectively. N_q denotes the number of training queries and N_d denotes the number of documents in the collection.

During inference, we first encode a new query q by QEnc. Then, we conduct a maximum inner-product search (MIPS) to get the top-m most similar queries $(q_1, q_2, ..., q_m)$ and their inner-product similarities $(s_q^1, s_q^2, ..., s_q^m)$ from the training set, as well as the top-n most similar documents $(d_1, d_2, ..., d_n)$ and their similarities $(s_d^1, s_d^2, ..., s_d^n)$ from the collection:

$$(q_1, s_q^1), (q_2, s_q^2), ..., (q_m, s_q^m) = \text{MIPS}(\text{QEnc}(q), \mathcal{Q})$$

$$(d_1, s_d^1), (d_2, s_d^2), ..., (d_n, s_d^n) = \text{MIPS}(\text{QEnc}(q), \mathcal{D})$$

The similarity scores are further normalized by a softmax function:

$$\tilde{s}_q^1, \tilde{s}_q^2, ..., \tilde{s}_q^m = \text{softmax}(s_q^1, s_q^2, ..., s_q^m)$$

$$\tilde{s}_d^1, \tilde{s}_d^2, ..., \tilde{s}_d^n = \text{softmax}(s_d^1, s_d^2, ..., s_d^n)$$

We denote the mapping from a query to its relevant documents in the training set as Rel : $q \mapsto \{d\}$. The final score of a document d

Table 1: Model performance on the TripClick test sets. Baseline results are from [29]. LA: log augmentation; DR: dense retrieval.

Model	HEAD (DCTR relevance)			TORSO (RAW relevance)			TAIL (RAW relevance)		
	NDCG@10	MRR	Recall@10	NDCG@10	MRR	Recall@10	NDCG@10	MRR	Recall@10
BM25 [30]	0.140	0.290	0.138	0.206	0.283	0.262	0.267	0.258	0.409
RM3 PRF [23, 26]	0.141	0.300	0.136	0.194	0.261	0.254	0.242	0.227	0.384
PACRR [16]	0.175	0.356	0.162	0.212	0.302	0.262	0.267	0.261	0.409
MP [27]	0.183	0.372	0.173	0.243	0.347	0.297	0.281	0.280	0.409
KNRM [36]	0.191	0.393	0.173	0.235	0.338	0.283	0.272	0.265	0.409
ConvKNRM [5]	0.198	0.420	0.178	0.243	0.358	0.288	0.271	0.265	0.409
TK [15]	0.208	0.434	0.189	0.272	0.381	0.321	0.295	0.279	**0.459**
LADER (ours)	**0.338**	**0.664**	**0.304**	**0.303**	**0.427**	**0.353**	**0.310**	**0.306**	0.449
LADER w/o LA	0.247	0.532	0.237	0.241	0.350	0.293	0.260	0.257	0.394
LADER w/o DR	0.324	0.649	0.284	0.266	0.396	0.298	0.232	0.236	0.330

in the candidate set $\{d_1, d_2, ..., d_n\} \cup \{\text{Rel}(q_1), \text{Rel}(q_2), ..., \text{Rel}(q_m)\}$ is a weighted sum computed as follows:

$$\text{Score}(d) = \sum_{j \in [1,...,n]} \tilde{s}_d^j \, \mathbf{1}(d = d_j) + \sum_{i \in [1,...,m]} \lambda \tilde{s}_q^i \, \mathbf{1}(d \in \text{Rel}(q_i))$$

where the former part is the dense retrieval score and the latter part is the log-augmentation score from similar queries, $\mathbf{1}$ is the indicator function, and λ is a hyper-parameter to control the extent of log augmentation. We return the candidate documents ranked by their final scores to the input query.

3 EXPERIMENTS

3.1 Settings

Dataset. We evaluate our LADER method on the TripClick benchmark [29], which contains 692k unique queries and 1.5M documents (PubMed abstracts). Based on their frequencies, the queries are divided into HEAD (>44), TORSO (6–44), and TAIL (<6) subsets. Both the validation and test sets contain 1,175 queries for each HEAD, TORSO and TAIL subset. Following [14, 29], we use two sets of relevance scores: The "RAW" relevance is used to judge TORSO and TAIL queries, where clicked documents have a score of 1 and other documents have a score of 0. The Graded Document Click-Through Rate [3, 4] ("DCTR") relevance is used to judge the HEAD queries, where the document score is defined as the number of clicks divided by the number of exposure. To train the dense retriever (§2.1), we use 10M triples released by [14], where each triple contains a TripClick training query, a clicked document, and a non-relevant document sampled from BM25 negatives.

Configuration. We implement LADER with PyTorch [28] and HuggingFace's libraries [35]. For training the dense retriever, we use the AdamW optimizer [22, 25] for 20k steps with the learning rate of 2e-5 and weight decay of 1e-2, batch size of 256, 10k warmup steps, the cosine learning rate decay schedule, $\alpha = 5.0$, $\beta = 0.9$, and the Euclidean distance for the triplet loss. For inference, we implement MIPS with FAISS's `FlatIP` index [18], $m = n = 1000$, $N_q = 685,649$, and $N_d = 1,523,871$. We use $\lambda = 0.5$ for the HEAD and TORSO queries, and $\lambda = 0.2$ for the TAIL queries. We also experiment with two ablations: LADER w/o Log Augmentation (LA) where LA scores are set to 0, and LADER w/o Dense Retrieval (DR) where DR scores are set to 0.

Baseline methods for comparison. We compare LADER with various baselines in [29] on all queries, including BM25 [30], RM3 Pseudo Relevance Feedback (PRF) [23, 26], Position Aware Convolutional Recurrent Relevance Matching (PACRR) [16], Match Pyramid (MP) [27], Kernel-based Neural Ranking Model (KNRM) [36], Convolutional KNRM (ConvKNRM) [5], and Transformer-Kernel (TK) [15]. For the HEAD queries, we further compare with current SOTA methods [14] based on pre-trained language models, including bi-encoders initialized by different BERT models [2, 13, 31]. Following their respective metrics, we compare with benchmark baselines [29] using NDCG@10, MRR, and Recall@10, and compare with HEAD query SOTA [14] using NDCG@10, MRR@10, and also Recall@1k.

3.2 Main Results

Comparison with benchmark baselines. Table 1 shows comprehensive comparisons of LADER with a variety of benchmark baselines [29] on all queries. LADER outperforms all benchmark baselines on each query subset, and on all metrics except the Recall@10 for the TAIL queries. On the most frequent HEAD queries, LADER outperforms the best benchmark baseline method (TK) by large margins, showing 63% (0.338 v.s. 0.208), 53% (0.664 v.s. 0.434), and 61% (0.304 v.s. 0.189) relative gains on NDCG@10, MRR, and Recall@10, respectively. On the less frequent TORSO queries, the performance improvements over the previous SOTA are decent with 10% to 12% relative gains on all metrics. On the rare TAIL queries, LADER still performs favorably than previous SOTA on NDCG@10 (0.310 v.s. 0.295) and MRR (0.306 v.s. 0.280), but the performance is slightly lower on Recall@10 (0.449 v.s. 0.459).

Comparison with BERT DOT. In Table 2, we compare LADER with more recent BERT-based retrievers (BERT DOT) [14], whose results are only available on the HEAD queries. Compared with the BERT DOT dense retriever, LADER has 39% (0.338 v.s. 0.243), 24% (0.659 v.s. 0.530), and 7% (0.893 v.s. 0.828) relative improvement on NDCG@10, MRR@10, and Recall@1k, respectively. This shows the effectiveness of log augmentation since our backbone dense retriever (LADER w/o LA) performs similarly to their counterparts.

Table 2: LADER results on the HEAD queries compared to BERT DOT [14]; LA: log augmentation; DR: dense retrieval.

Model	NDCG@10	MRR@10	R@1k
BM25 [30]	0.140	0.276	0.834
BERT DOT [14]			
w/ DistillBERT [31]	0.236	0.512	0.813
w/ SciBERT [2]	0.243	0.530	0.793
w/ PubMedBERT [13]	0.235	0.509	0.828
LADER (ours)	**0.338**	**0.659**	**0.893**
LADER w/o LA	0.247	0.526	0.878
LADER w/o DR	0.324	0.644	0.670
Log-Augmented (LA-)			
Sparse retriever (BM25)	0.312	0.598	0.889
Raw PubMedBERT [13]	0.137	0.288	0.453

Improvement over the backbone dense retriever. On all queries, LADER improves the performance of the backbone dense retriever by 24%-37% relative NDCG@10 (LADER v.s. LADER w/o LA) while not requiring additional training. The performance improvement is more significant on the HEAD queries than on the TORSO or TAIL queries, which will be further analyzed in the next section.

3.3 Analysis

Importance of trained dense retrievers. In Table 2, we show the results of log augmentation with different backbone retrievers. Log-augmented raw PubMedBERT performs much worse than LADER (0.137 v.s. 0.338 NDCG@10), suggesting the importance of the backbone dense retriever to be trained on retrieval tasks (§2.1). We also replace the dense retriever in LADER with BM25, which we denote as LABM25 and implement with Pyserini [24]. LABM25 greatly improves the original BM25 baseline by 122% (0.312 v.s. 0.140 NDCG@10), which proves that the effectiveness of log augmentation is retriever-agnostic. However, LADER still outperforms LABM25 (0.338 v.s. 0.312 NDCG@10), indicating that the potential of log augmentation is better harnessed by dense retrievers.

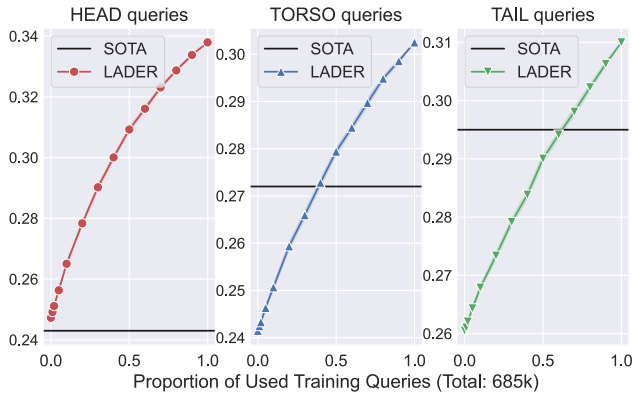

Figure 3: NDCG@10 of LADER with different proportions of training queries to retrieve in log-augmentation.

Effects of log size. In Figure 3, we show the performance of LADER using different proportions of queries sampled from the training set for the log augmentation. For all queries, the performance improves with the number of queries to retrieve, and since these curves have not saturated yet, more performance gains are expected with more logs. We also find that on lower frequency queries, more training queries are required to retrieve from for LADER to outperform current SOTA methods.

What queries gain more from log-augmentation? We collect 9 features for each query, including query length (QL), query group (HEAD, TORSO or TAIL), ENT(Q): the entropy of $[\tilde{s}_q^1, \tilde{s}_q^2, ..., \tilde{s}_q^m]$, ENT(D): the entropy of $[\tilde{s}_d^1, \tilde{s}_d^2, ..., \tilde{s}_d^n]$, E(REL): the expectation of relevant document number, and the average number of relevant documents in the top-1 and top-5 similar queries (REL1 and REL5). We fit a linear regression model to predict the gain of NDCG@10 by log-augmentation (LADER v.s. LADER w/o LA) using min-max normalized query features. The feature coefficients shown in Figure 4 indicate that: (1) Query frequency is the most important feature: more frequent (HEAD group) queries benefit more than average from log augmentation, while rare (TAIL group) queries benefit less than average; (2) Queries without very similar documents (higher ENT(D)), benefit more than queries with very similar documents; (3) Queries with very similar queries in the log (lower ENT(Q)), benefit more from log augmentation than queries without.

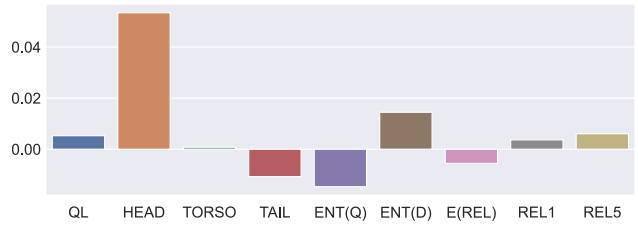

Figure 4: Feature coefficients in the regression analysis. Positive values indicate more gains from log augmentation.

4 CONCLUSIONS AND LIMITATIONS

We present LADER, a simple and novel plug-in module that uses search logs to augment dense retrievers. Our results show that LADER achieves new SOTA on TripClick, and can largely improve the performance of its backbone retriever without additional training. We also provide thorough analyses of its characteristics.

One limitation of LADER is that it increases the search latency by about N_q/N_d (45% for TripClick) due to the additional query-to-query retrieval step. Another drawback of this study is that we only use data from one search engine. It remains interesting to test the generalizability of LADER to a different search engine, which will be beneficial for cold-starting new literature search initiatives.

ACKNOWLEDGMENTS

We are grateful to the TripClick benchmark organizers for sharing the data. We also thank the SIGIR reviewers for their constructive comments. This research was supported by the NIH Intramural Research Program, National Library of Medicine.

REFERENCES

[1] Ricardo A. Baeza-Yates, Carlos A. Hurtado, and Marcelo Mendoza. 2007. Improving search engines by query clustering. *J. Assoc. Inf. Sci. Technol.* 58, 12 (2007), 1793–1804. https://doi.org/10.1002/asi.20627

[2] Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A Pretrained Language Model for Scientific Text. 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 2019, Hong Kong, China, November 3-7, 2019*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, 3613–3618. https://doi.org/10.18653/v1/D19-1371

[3] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. 2015. *Click Models for Web Search*. Morgan & Claypool Publishers. https://doi.org/10.2200/S00654ED1V01Y201507ICR043

[4] Nick Craswell, Onno Zoeter, Michael J. Taylor, and Bill Ramsey. 2008. An experimental comparison of click position-bias models. In *Proceedings of the International Conference on Web Search and Web Data Mining, WSDM 2008, Palo Alto, California, USA, February 11-12, 2008*, Marc Najork, Andrei Z. Broder, and Soumen Chakrabarti (Eds.). ACM, 87–94. https://doi.org/10.1145/1341531.1341545

[5] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional Neural Networks for Soft-Matching N-Grams in Ad-hoc Search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5-9, 2018*, Yi Chang, Chengxiang Zhai, Yan Liu, and Yoelle Maarek (Eds.). ACM, 126–134. https://doi.org/10.1145/3159652.3159659

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, 4171–4186. https://doi.org/10.18653/v1/n19-1423

[7] Rezarta Islamaj Dogan, G. Craig Murray, Aurélie Névéol, and Zhiyong Lu. 2009. Understanding PubMed® user search behavior through log analysis. *Database* 2009 (11 2009). https://doi.org/10.1093/database/bap018 arXiv:https://academic.oup.com/database/article-pdf/doi/10.1093/database/bap018/1022874/bap018.pdf bap018.

[8] John W. Ely, Jerome A. Osheroff, M. Lee Chambliss, Mark H. Ebell, and Marcy E. Rosenbaum. 2005. Research Paper: Answering Physicians' Clinical Questions: Obstacles and Potential Solutions. *J. Am. Medical Informatics Assoc.* 12, 2 (2005), 217–224. https://doi.org/10.1197/jamia.M1608

[9] Nicolas Fiorini, Kathi Canese, Grisha Starchenko, Evgeny Kireev, Won Kim, Vadim Miller, Maxim Osipov, Michael Kholodov, Rafis Ismagilov, Sunil Mohan, et al. 2018. Best match: new relevance search for PubMed. *PLoS biology* 16, 8 (2018), e2005343.

[10] Nicolas Fiorini, Robert Leaman, David J Lipman, and Zhiyong Lu. 2018. How user intelligence is improving PubMed. *Nature biotechnology* 36, 10 (2018), 937–945.

[11] Giacomo Frisoni, Miki Mizutani, Gianluca Moro, and Lorenzo Valgimigli. 2022. BioReader: a Retrieval-Enhanced Text-to-Text Transformer for Biomedical Literature. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 5770–5793. https://aclanthology.org/2022.emnlp-main.390

[12] Vishrawas Gopalakrishnan, Kishlay Jha, Wei Jin, and Aidong Zhang. 2019. A survey on literature based discovery approaches in biomedical domain. *J. Biomed. Informatics* 93 (2019). https://doi.org/10.1016/j.jbi.2019.103141

[13] Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing. *ACM Trans. Comput. Healthcare* 3, 1, Article 2 (oct 2021), 23 pages. https://doi.org/10.1145/3458754

[14] Sebastian Hofstätter, Sophia Althammer, Mete Sertkan, and Allan Hanbury. 2022. Establishing Strong Baselines For TripClick Health Retrieval. In *Advances in Information Retrieval - 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10-14, 2022, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 13186)*, Matthias Hagen, Suzan Verberne, Craig Macdonald, Christin Seifert, Krisztian Balog, Kjetil Nørvåg, and Vinay Setty (Eds.). Springer, 144–152. https://doi.org/10.1007/978-3-030-99739-7_17

[15] Sebastian Hofstätter, Hamed Zamani, Bhaskar Mitra, Nick Craswell, and Allan Hanbury. 2020. Local Self-Attention over Long Text for Efficient Document Retrieval. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy X. Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 2021–2024. https://doi.org/10.1145/3397271.3401224

[16] Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. 2017. PACRR: A Position-Aware Neural IR Model for Relevance Matching. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, Martha Palmer, Rebecca Hwa, and Sebastian Riedel (Eds.). Association for Computational Linguistics, 1049–1058. https://doi.org/10.18653/v1/d17-1110

[17] Qiao Jin, Chuanqi Tan, Mosha Chen, Ming Yan, Ningyu Zhang, Songfang Huang, Xiaozhong Liu, et al. 2022. State-of-the-Art Evidence Retriever for Precision Medicine: Algorithm Development and Validation. *JMIR Medical Informatics* 10, 12 (2022), e40743.

[18] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data* 7, 3 (2019), 535–547.

[19] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, 6769–6781. https://doi.org/10.18653/v1/2020.emnlp-main.550

[20] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through Memorization: Nearest Neighbor Language Models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. https://openreview.net/forum?id=HklBjCEKvH

[21] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy X. Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 39–48. https://doi.org/10.1145/3397271.3401075

[22] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1412.6980

[23] Victor Lavrenko and W. Bruce Croft. 2001. Relevance-Based Language Models. In *SIGIR 2001: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, September 9-13, 2001, New Orleans, Louisiana, USA*, W. Bruce Croft, David J. Harper, Donald H. Kraft, and Justin Zobel (Eds.). ACM, 120–127. https://doi.org/10.1145/383952.383972

[24] Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python Toolkit for Reproducible Information Retrieval Research with Sparse and Dense Representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*. 2356–2362.

[25] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=Bkg6RiCqY7

[26] Yuanhua Lv and ChengXiang Zhai. 2009. A comparative study of methods for estimating query language models with pseudo feedback. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009*, David Wai-Lok Cheung, Il-Yeol Song, Wesley W. Chu, Xiaohua Hu, and Jimmy Lin (Eds.). ACM, 1895–1898. https://doi.org/10.1145/1645953.1646259

[27] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text Matching as Image Recognition. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, Dale Schuurmans and Michael P. Wellman (Eds.). AAAI Press, 2793–2799. http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11895

[28] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (Eds.). 8024–8035. https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html

[29] Navid Rekabsaz, Oleg Lesota, Markus Schedl, Jon Brassey, and Carsten Eickhoff. 2021. 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* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 2507–2513. https://doi.org/10.1145/3404835.3463242

[30] Stephen E. Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* 3, 4 (2009), 333–389. https://doi.org/10.1561/1500000019

[31] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR* abs/1910.01108 (2019). arXiv:1910.01108 http://arxiv.org/abs/1910.01108

[32] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. FaceNet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*. IEEE Computer Society, 815–823. https://doi.org/10.1109/CVPR.2015.7298682

[33] Xiaoyuan Su and Taghi M. Khoshgoftaar. 2009. A Survey of Collaborative Filtering Techniques. *Adv. Artif. Intell.* 2009 (2009), 421425:1–421425:19. https://doi.org/10.1155/2009/421425

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 5998–6008. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[35] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. 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, 38–45. https://www.aclweb.org/anthology/2020.emnlp-demos.6

[36] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-End Neural Ad-hoc Ranking with Kernel Pooling. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White (Eds.). ACM, 55–64. https://doi.org/10.1145/3077136.3080809

[37] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=zeFrfgyZln

[38] Zhijun Yin, Milad Shokouhi, and Nick Craswell. 2009. Query Expansion Using External Evidence. In *Advances in Information Retrieval, 31th European Conference on IR Research, ECIR 2009, Toulouse, France, April 6-9, 2009. Proceedings (Lecture Notes in Computer Science, Vol. 5478)*, Mohand Boughanem, Catherine Berrut, Josiane Mothe, and Chantal Soulé-Dupuy (Eds.). Springer, 362–374. https://doi.org/10.1007/978-3-642-00958-7_33

[39] Zheng Yuan, Qiao Jin, Chuanqi Tan, Zhengyun Zhao, Hongyi Yuan, Fei Huang, and Songfang Huang. 2023. RAMM: Retrieval-augmented Biomedical Visual Question Answering with Multi-modal Pre-training. *CoRR* abs/2303.00534 (2023). https://doi.org/10.48550/arXiv.2303.00534 arXiv:2303.00534

[40] Shengyao Zhuang, Hang Li, and Guido Zuccon. 2022. Implicit Feedback for Dense Passage Retrieval: A Counterfactual Approach. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai (Eds.). ACM, 18–28. https://doi.org/10.1145/3477495.3531994