Author/s:
Goli, R;Moffat, A;Buchanan, G

Title:
Refined Medical Search via Dense Retrieval and User Interaction

Date:
2025

Citation:
Goli, R., Moffat, A. & Buchanan, G. (2025). Refined Medical Search via Dense Retrieval and User Interaction. SIGIR '25: Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.3315-3324. Association for Computing Machinery. https://doi.org/10.1145/3726302.3730292.

Persistent Link:
https://hdl.handle.net/11343/361944

Refined Medical Search via Dense Retrieval and User Interaction

Reyhaneh Goli
The University of Melbourne
Melbourne, Australia
rgoli@student.unimelb.edu.au

Alistair Moffat
The University of Melbourne
Melbourne, Australia
ammoffat@unimelb.edu.au

George Buchanan
RMIT University
Melbourne, Australia
george.buchanan@rmit.edu.au

Abstract

Users formulate search queries that reflect an information need. Those queries are then submitted to a search service in the expectation that the retrieved results will allow the user to complete an external task, and align with their broader information context.

In this study, we reimplement and reproduce the log-augmented dense retrieval approach introduced by Jin, Shin, and Lu in 2023. As part of our study we extend the experimentation by: (1) using all of the available training data rather than a subset; (2) exploring a second dense retrieval model; and (3) enhancing the approach by incorporating user reformulation behavior into the dense retrieval computation so as to improve ranking effectiveness. To evaluate these modifications, we again utilize the TripClick IR benchmark, which comprises approximately four million click log entries from a health domain web search engine.

Although we were unable to exactly replicate the results of Jin, Shin, and Lu, our findings confirm the overall trends reported in their study. Specifically, our results support the conclusion that incorporating user interactions into dense retrieval models improves ranking effectiveness compared to when no user information is available. Moreover, our enhanced formulation allows further small gains to be made in retrieval effectiveness.

CCS Concepts

• **Information systems** → **Retrieval effectiveness**; *Query log analysis*; *Task models*; Query intent.

Keywords

Dense retriever; user behavior analysis; query log mining

ACM Reference Format:
Reyhaneh Goli, Alistair Moffat, and George Buchanan. 2025. Refined Medical Search via Dense Retrieval and User Interaction. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '25), July 13–18, 2025, Padua, Italy*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3726302.3730292

1 Introduction

Information retrieval systems play a crucial role in enabling users to efficiently access relevant information, with preference given to systems that accurately interpret user intent and return high-quality results. Traditional retrieval models, such as BM25, rely on lexical matching, with the predicted relevance of a document to the search query determined by exact term overlap and term weight. While effective in many cases, lexical-based retrieval models struggle in two key ways: (1) vocabulary mismatch, where relevant documents may not be retrieved if they do not contain the terms used in a query but instead use synonyms; and (2) semantic mismatch, where documents containing query terms may be ranked highly even if the meaning of term usage differs from that desired. As a result, these models may fail to prioritize semantically appropriate documents.

Dense retrieval models address these limitations, representing queries and documents as contextualized embeddings in a continuous vector space. As a consequence, dense retrievers better capture query and document semantics, generalizing beyond mere term matching. Most dense retrieval approaches employ a single-representation approach, with queries and documents mapped to same-dimension vectors, and with relevance estimated by the inner product of their corresponding embeddings. While dense retrieval models have demonstrated improvements over traditional methods [13], there remains an open challenge in effectively incorporating user interactions to further refine retrieval quality.

In this study, we reproduce and extend prior work by Jin et al. [11], exploring their *log-augmented dense retrieval* mechanism, and investigating the use of user interaction data. Specifically, we:

- seek to replicate the results of Jin et al.; then
- quantify the effect that training set size has on effectiveness;
- train and evaluate different dense retrieval models; and
- integrate user reformulation history to further improve rankings.

Figure 1 gives an overview of the Jin et al. [11] proposal, showing the key steps and workflow of the traditional (red arrows) and augmented (red plus green arrows) approaches, and highlighting how user interaction signals such as click logs and query reformulations can be integrated into the dense retrieval process.

To implement and evaluate these approaches, we make use of the *TripClick IR Benchmark* [23], collection, containing approximately 1.5 million health-focused documents; around 692,000 user queries; and approximately four million click log entries.

We were unable to exactly replicate the values reported by Jin et al. [11]. Nevertheless, our results align with their overall findings. Specifically, we confirm that trained dense retrieval models yield better rankings than corresponding untrained ones. Additionally, we show that incorporating user interactions enhances retrieval quality, with click logs playing a important role and query reformulations providing an additional layer of improvement. These findings reinforce the importance of user behavior signals, allowing more accurate rankings and an improved user experience.

The remainder of this paper is organized as follows. Section 2 reviews related work, discussing different retrieval approaches and dense retrieval models. Section 3 then details our methodology for reproducing log-augmented dense retrieval, covering data processing, model training, and evaluation metrics; and noting differences.

SIGIR '25, July 13–18, 2025, Padua, Italy

ACM ISBN 979-8-4007-1592-1/2025/07
https://doi.org/10.1145/3726302.3730292

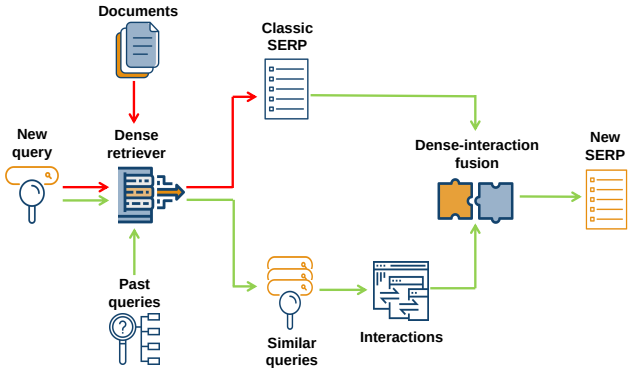

Figure 1: Integrating past interactions, including query click logs and query reformulation data, into the ranking process. The red path represents the standard dense retrieval approach, while the additional green elements depict the enhanced ranking system described by Jin et al. [11] and summarized in Section 3.

Next, Section 4 steps beyond reproduction, describing our extended approach and analyzing the results of further experiments. Section 5 discusses key findings, limitations, and potential future directions.

2 Background

Substantial research has been conducted to improve retrieval effectiveness, and has resulted in the development of a range of models and approaches. In this section, we begin by examining traditional retrieval models, which have been widely studied and applied in information retrieval. We then explore the evolution of dense retrieval models, which address key limitations of traditional methods by shifting to semantic representations of queries and documents. Following this, we briefly summarize the different types of dense retrieval models, highlighting their underlying mechanisms and the distinctions between them. Finally, we discuss advancements in dense retrieval that have further improved the accuracy and effectiveness of retrieval systems.

Traditional Retrieval Methods. Information retrieval systems are often built on top of retrieval models that are usually now referred to as *sparse* arrangements. In these models, queries and documents are represented as vectors in a high-dimensional term space based upon lexical token such as stopped/stemmed words. The "sparse" label refers to the fact that any particular query or document vector will contain only a tiny minority of the huge set of options provided by the collection vocabulary, meaning that most entries in most vectors will be zeros. In this approach document similarity scoring is based on term matching between query and document vectors, with scores influenced by the existence of overlapping terms, and then further ordered by a range of secondary factors including document length [2]. Sparse retrieval models, which frequently use inverted indexes for query processing (see, for example, Zobel and Moffat [36]) are both interpretable and computationally efficient.

The BM25 computation (see Robertson and Zaragoza [25] for details) is usually now regarded as the exemplar of this class of computation. It scores documents according to word frequencies, inverse document frequencies, and a document length normalization component. Because it performs well over a wide range of datasets and domains, and can be efficiently computed using an inverted index, BM25 has established itself as a standard baseline for many information retrieval tasks.

Refinement is then possible using *pseudo-relevance feedback* (PRF) methods like RM3, which analyze the retrieved documents, and re-issue expanded queries containing added terms semantically connected to those found in highly-scoring documents [18].

Pre-dating ranking approaches that make use of similarity computations, Boolean retrieval models rely on strict logical conditions, using operators such as AND/OR/NOT, to identify an unordered set of documents from the collection that match the stipulated conditions [29]. Boolean retrieval is still used in some applications such as legal discovery and medical systematic review, where repeatability and explainability are important.

Dense Retrieval. Dense retrieval models address the limitations of traditional retrieval approaches by encoding queries and documents into contextualized embeddings. These embeddings are learned representations that better capture the semantic meaning of text, thereby facilitating matching beyond exact term overlap [21]. In dense retrieval, the relevance of a document to a query is estimated by computing the inner product of their embeddings, making their retrieved results sets less dependent on queries and documents using exactly the same terms (rather than, for example, synonyms) than are traditional lexical retrieval systems. Dense retrieval models have consistently demonstrated significant improvements over sparse retrieval methods in terms of retrieval effectiveness [14, 33].

Dense retrieval models are commonly built on top of a general-purpose language model constructed from a large-scale non-specific corpus, and then fine-tuned for a specific retrieval task [34]. The general-purpose model incorporates foundational language knowledge acquired from the broad corpus; the fine-tuning then adds in task-specific datasets, thereby enhancing the system's ability to encode queries and documents into domain-relevant embeddings. That is, fine-tuning is essential if retrieval effectiveness is to be maximized in the context of specific domains or applications [13].

Types of Dense Retrieval. Dense retrieval models can be categorized into single-representation and multi-representation approaches, based on how they encode queries and documents in the embedding space [19]. Single-representation models, such as Dense Passage Retrieval (DPR) [13], Approximate nearest neighbor Negative Contrastive Estimation (ANCE) [32], and Tightly-Coupled Teachers ColBERT (TCT-ColBERT) [17], generate a single fixed-length embedding for each query and document. Relevance is then computed using the inner product between these embeddings. These approaches offer computational efficiency and scalability, making them particularly suitable for large-scale retrieval tasks. However, the simplicity of single-representation models comes at the cost of losing token-level information, which may limit their ability to handle complex or ambiguous queries effectively.

In contrast, multi-representation models retain token-level embeddings for both queries and documents, allowing them to incorporate interactions between individual tokens. Relevance is determined through token-wise comparisons, enabling these models

to capture richer semantic relationships. A prominent example is Contextualized Late Interaction over BERT (ColBERT) [14], which employs a two-stage scoring mechanism. In the first stage, an approximate nearest neighbor search identifies a set of candidate documents; then, in the second stage, those candidates are re-ranked using a late interaction mechanism to compute token-wise relevance scores. While multi-representation models offer improved effectiveness and are better suited for handling complex queries, they are computationally more expensive than single-representation models as collection size increases.

Developments in Dense Retrieval. Significant progress has been made recently in terms of improving both the efficiency and the effectiveness of dense retrieval models, addressing challenges such as scalability, and adaptability to domain-specific tasks. One major area of focus has been the development of better pretraining strategies. For instance, CoCondenser [6] eliminates the need for extensive data engineering steps, such as synthesis or filtering, and avoids reliance on large-batch training, thereby simplifying the pretraining process while maintaining effectiveness. Similarly, RocketQAv2 [24] introduces a joint training approach for dense passage retrieval and passage re-ranking. A key innovation is dynamic listwise distillation, a unified listwise training method that enables the retriever and re-ranker to adaptively improve based on each other's relevance information, enhancing overall effectiveness.

Hard negative sampling has also been employed to train dense retrievers more effectively. By incorporating challenging negative examples during training, models are forced to better distinguish between relevant and irrelevant documents, significantly improving retrieval accuracy [33].

Another area of advancement involves improving retrieval efficiency, seeking to reduce the computational costs associated with dense retrieval. Methods such as FAISS [12] and ScaNN [8] optimize approximate nearest neighbor searches, reducing query latency while preserving retrieval quality. These techniques make dense retrieval more practical for real-world, large-scale applications.

Exploiting User Interactions. Information retrieval is by its very nature an interactive task between user and retrieval systems. Researchers in IR, human-computer interaction and information science have all endeavored to uncover patterns in user interactions, including search strategies, methods for both formulating and reformulating queries, and the tactics with which users parse results and choose what to inspect at what level of detail.

There have been three major strands of work that examine query (re-) formulation: first, examining the use of query syntax and logical controls; second, the selection of query terms; and third shifts in the use of tactics across a search session. Researchers quickly established that the use of logic and syntax is relatively rare, and applied sparingly even by effective searchers. In contrast, the selection of query terms involved both domain expertise and general search skills. Across time, more attention has been paid to shifts in query tactics across a search session, and that research has similarly revealed that effective users combine domain expertise with transferable search skill.

Marchionini et al.'s pioneering work quickly revealed the importance of both a user's general search skill and their domain expertise [20]. Across time, different research methods have been used to build on this foundation, providing more detailed and concrete analysis. Wildemuth [31] built on Bates' theory of search tactics [4] to examine the influence of higher or lower domain expertise on query reformulation. She found that greater expertise led to better selection of domain concepts, and hence of query vocabulary. This in turn led to better search results and to shorter, more successful search sessions.

Others have followed these early researchers, confirming the importance of domain expertise [5] and also its limits. Domain expertise appears to have very limited transferability [30]. The underlying intuition of exploiting logs of user reformulations is that logs will uncover common, and likely more effective, reformulation strategies, particularly in regard to term selection. This should particularly benefit novices to a domain, who may choose suboptimal query terms. If these caveats hold, extracted reformulation patterns will likely be domain-specific.

3 Reproducibility

This section describes our work to reproduce the log-augmented retrieval approach proposed by Jin et al. [11]. We first provides a detailed explanation of Jin et al.'s proposal, outlining its motivation, key methodological aspects, and our rationale for reproducing it. We then describe the dataset used in our experiments, its structure, and its characteristics, and follow with an explanation of the training process, implementation decisions, model configurations, and other key assumptions. We then present the result of our reproduction efforts, comparing our observations with the findings reported by Jin et al., and discussing the similarities and discrepancies, and factors that may have contributed to the deviations observed.

Log-Augmented Retrieval Reproduction. The Log-Augmented Dense Retrieval (LADER) framework proposed by Jin et al. [11] is a plug-in module designed to enhance dense retrieval models by exploiting historical user interactions, specifically click logs from training queries that are assumed to be similar to those expected in an operational system. The LADER system operates in two main steps. First, given a query, it identifies both similar queries and similar documents using a dense retrieval model. The second step then improves the ranking by incorporating click data from any similar queries (the green edges in Figure 1), boosting the scores of any documents that have been previously clicked for related queries, weighting the increments based on their similarity to the current query. This approach is motivated by the observations that similar information needs tend to yield similar document-click patterns [35]; and that for any given information need a range of queries is possible [3, 22].

The dense retriever is a core component of LADER. It consists of two neural encoders: one for queries, and one for documents. These take query text and document text respectively, and return corresponding vector representations. Both encoders are initialized using PubMedBERT, a domain-specific BERT model designed for biomedical applications, pre-trained on PubMed articles [7]. The encoders are then fine-tuned on the TripClick dataset (details are provided shortly). In LADER's implementation, the last hidden states of PubMedBERT are used to generate query and document embeddings, mapping each input into a 768-dimensional space.

Once trained, the document encoder is used to encode all of the documents in the collection. When a query is received, it is likewise encoded. A maximum inner product search is then applied to retrieve the most similar documents from the collection, as well as the most similar queries from the log of retained query encodings.

By the end of this stage each document has an assigned similarity score based on its nearness to the query in the embedding space. The log-augmentation module is then applied to refine the ranking. Specifically, LADER identifies the documents that were clicked by previous users when shown as responses to one of more of the set of selected "most similar" previous queries; and computes an additional score component based on that historical exposure and the similarity of the retrieved query to the current query.

The final score for each document is computed as a weighted linear combination of two components: (1) the similarity score from the dense retriever; and (2) an additional score derived from log information the document was previously clicked for a similar query. The document ranking returned to the user is determined by that overall score. Details are provided shortly.

Dataset and Resources. Jin et al. [11] trained LADER and their further extensions using the TripClick IR benchmark [23], a collection of around 1.5 million documents, 692,000 queries, and approximately four million click log entries connected to those queries. It is derived from the Trip web site[1], a health web search service for clinical research support primarily used by health-care professionals. The raw dataset contains 5.2 million clicks collected between 2013 and 2020, spanning approximately 1.6 million search sessions. Further details of the dataset and the methodology for constructing the IR benchmark are available at the TripClick web site[2].

Both query encoder and document encoder are initialized with PubMedBERT and fine-tuned on a triplet-based training collection derived from the TripClick dataset. This triplets were refined by Hofstätter et al. [9], building on an earlier version released by Rekabsaz et al. [23], and contains 10 million triplets. Each triplet consists of a query text, a clicked document (positive example), and a non-relevant document (negative example). Hofstätter et al. refined the triplet dataset using negative sampling techniques, improving the training effectiveness of the dense retriever.

The TripClick IR benchmark queries are partitioned into three frequency-based groups. Queries that appear more than 44 times are classified as Head queries; those occurring between 6 and 44 times are in the Torso category; and queries appearing fewer than 6 times are Tail queries. This process, developed by Rekabsaz et al. [23], resulted in the 692,699 queries being split as 5,879 Head queries; 108,314 Torso queries; and 578,506 Tail queries. Each query group is further split into training, validation, and test sets, with all groups containing 1175 queries in both the validation and test sets.

In the absence of human annotations indicating the relevance of each document to each query, two surrogate approaches are employed. The first Raw option assigns a binary grade, with all clicked document for any instance of that identical query deemed relevant (score 1) and all previously-returned unclicked documents assigned a score of 0. The inferred qrels can then be employed with all three query groups (Head, Torso, and Tail). The second DCTR (Click-Through Rate-Based Relevance) method estimates query-document relevance as the ratio between the number of times a document was clicked for that exact query, divided by the total number of times it appeared in search results for that query; with those numeric ratios then mapped to the ordinal scale 0–3. In this DCTR surrogate, relevance grade 0 indicates non-relevance, and grade 3 represents high relevance. Note that DCTR-based effectiveness measurements are only applicable to Head queries because of the need for sufficient interaction data to compute meaningful click-through rates. Details of the threshold criteria used for assigning the DCTR relevance grades are provided by Rekabsaz et al. [23].

In the Raw qrels the average number of judged (relevant or non-relevant) documents per query for Head, Torso, and Tail queries is 41.9, 9.1, and 2.8, respectively. Of these 20.1, 4.0, and 1.3 documents are relevant. The DCTR labeling resulted in an average of 46.2 judgments per Head query of which 13.2 were at grade 1 or higher.

The reported results (see Table 1) are based on three top-weighted effectiveness measurements: Normalized Discounted Cumulative Gain (NDCG), Reciprocal Rank (RR), and Recall, all with the cutoff $k = 10$. The particular NDCG formulation employed is:

$$\text{NDCG@}k = \frac{\text{DCG@}k}{\text{IDCG@}k}\,, \tag{1}$$

with

$$\text{DCG@}k = \sum_{i=1}^{k} \frac{rel_i}{\log_2(i+1)}\,, \tag{2}$$

in which rel_i is the numeric relevance grade of the i th document in the ranking, that is, $rel_i \in \{0, 1\}$ for the binary Raw evaluations, and $rel_i \in \{0, 1, 2, 3\}$ for the DCTR evaluations; and in which IDCG refers to the highest possible DCG score at the stipulated ranking depth. In the RR and Recall evaluations using the Head queries and DCTR judgments, a document was deemed relevant if $rel_i \geq 1$.

Reimplementation. As did Jin et al. [11], we initialized the query encoder and document encoder with PubMedBERT, available from Hugging Face[3]. The encoders were then fine-tuned on the improved triplets dataset of Hofstätter et al. [9]. The improved version of the released triplets only includes query IDs, positive document IDs, and negative document IDs, but we were able to retrieve the corresponding documents with the support of the TripClick team.

Also following the setup employed in the original LADER paper [11], we used the AdamW optimizer [15] with a learning rate of 2×10^{-5} and a weight decay of 1×10^{-2}. The epsilon parameter was not explicitly noted in the original paper, so we assumed the default value of 1×10^{-8}. We also mimicked the scheduler of Jin et al., using a cosine schedule with 10,000 warmup steps. The number of training steps was specified as 20,000 with a batch size of 256, which, based on our understanding, suggests that the encoders were trained on approximately 5 million triplets.

To validate this setup, we first trained the encoders on 5 million triplets and then extended training to the entire dataset of 10 million triplets, to evaluate the extent to which increased the training data improved model performance. Jin et al. [11] do not provide information on the number of training epochs or the hardware

[1] https://www.tripdatabase.com/
[2] https://tripdatabase.github.io/tripclick/#tripclick-dataset
[3] https://huggingface.co/microsoft/BiomedNLP-BiomedBERT-base-uncased-abstract-fulltext

specifications used. In our experiments, we trained the encoders using four GPUs (H100), with a per-GPU batch size of 32, running for 40,000 steps in each epoch on the 5M dataset and for 80,000 steps on the 10M dataset.

For the loss function, we again followed the Jin et al. implementation, which combines two objectives: in-batch negative log-likelihood loss $\mathcal{L}_i$ [13] and triplet contrastive loss $\mathcal{L}_t$ [26]. The in-batch negative log-likelihood loss helps the dense retriever distinguish between positive documents and all other documents in the mini-batch, and is computed as:

$$\mathcal{L}_i = -\log \frac{\exp(E(q) \odot E(d^+))}{\sum_{i \in B} \exp(E(q) \odot E(d_i))}, \quad (3)$$

in which $E(q)$ and $E(d)$ represent the query embedding and document embedding respectively and $\odot$ is the scalar inner product of two vectors; B denotes the set of documents in the mini-batch; and d^+ is a positive example. The triplet loss further contrasts positive and hard negative documents and is computed as:

$$\mathcal{L}_t = \max\{0, dist(E(q), E(d^+)) - dist(E(q), E(d^-) + \alpha)\}, \quad (4)$$

with $dist(\cdot, \cdot)$ representing a distance computation, with α the margin between the positive and negative pairs, and with d^- a negative example. Based on the values reported in the original work, α was set to 5.0, and the Euclidean distance was used as the metric for triplet loss.

The final loss function is then computed as a combination of those two components:

$$\mathcal{L} = \beta \cdot \mathcal{L}_i + (1 - \beta) \cdot \mathcal{L}_t, \quad (5)$$

where β is a hyperparameter for loss weighting, and was set to 0.9 [11]. The model parameters were tuned using gradient-based optimizers, based on the training setup in the original work.

Once training was completed the query encoder and document encoder were used to generate the vector representations for the TripClick queries and documents. This process produced 685,649 query vectors and 1,523,871 document vectors. The validation set of queries and the test set of queries were then used to evaluate the performance of the dense retrieval model with the trained encoders.

For each incoming query, the model first converts it into its vector representation. Then, the maximum inner product between the query vector and all document vectors and (separately) query vectors is computed to identify sets of similar documents and queries. Jin et al. [11] chose to fetch 1000 similar documents and queries for each new query, and further normalized the similarity scores by a softmax function. We have replicated all of these configuration choices. The implementation of Jin et al. [11] made use of the FAISS FlatIP index [12], whereas our implementation directly computes inner products in an exhaustive manner. That difference does not alter effectiveness outcomes.

Once the top-1000 similar documents and top-1000 similar queries are retrieved, the clicked documents associated with each similar query q' are each assigned a score determined by the similarity between q and q'. This results in two sets of document scores:

- Retrieval Similarity Score: The score assigned to document d by the dense retriever when query q, calculated as $score_r(d, q) = softmax(E(d) \odot E(q))$, with the *softmax* calculated relative to the set of such values.
- Query Log-Derived Score: The score assigned to a document d as a result of being clicked when it was presented in connection with a similar query q', calculated as

$$score_q(d, q) = \sum_{q' \in Q_{1000}(q)} C(d, q') \cdot softmax(E(q') \odot E(q)), \quad (6)$$

where $Q_{1000}(q)$ is the set of 1000 most-similar previous queries to q, and $C(d, q') = 1$ if document d was both presented and clicked for any instance of query q', and $C(d, q') = 0$ otherwise; and where the *softmax* operator is applied holistically to the set of 1000 inner products.

The final score for document d in the context of query q is then:

$$score(d, q) = score_r(d, q) + \lambda \cdot score_q(d, q), \quad (7)$$

where λ is a hyperparameter that controls the extent of log augmentation, and varies with the query group, following the configuration describe by Jin et al. [11]. Specifically, $\lambda = 0.5$ for Head and Torso queries, and $\lambda = 0.2$ for Tail queries. Jin et al. did not specify the number of final retrieved results; here we follow typical practice and return a ranked list of 1000 documents to be assessed using a top-weighted effectiveness measurement.

Reproducibility Results. Table 1 lists three different system configurations: the complete LADER arrangement, which combines both dense retrieval and log augmentation; and then two simpler systems, LADER w/o DR, which evaluates the log-augmented module independently, and LADER w/o LA, which considers only the dense retrieval component without log augmentation. The first three data rows in Table 1 are taken directly from Table 1 of Jin et al. [11]. The next two blocks are our results, with DCTR relevance again used for Head queries, and Raw relevance again used for Torso and Tail queries, and with encoders trained using five million triplets. The first block shows the use of the query test set; the final group of three rows shows the effectiveness obtained on the validation set. (The impact of training with the full set of ten million triplet is considered in Section 4.)

The consistency of the LADER results between test and validation queries from our implementation demonstrates the stability of the models across different evaluation splits. A similar close alignment between the test results and the validation results is observed in the LADER w/o DR and LADER w/o LA rows in Table 1.

Comparing our LADER results and those of Jin et al., there is broad agreement, but in the headline LADER rows all of our effectiveness measurements were below the target values established by the first row of the table. Moreover, our LADER w/o LA values fall well short of the corresponding values of Jin et al.. This suggests that the difference in the LADER scores is arising in the dense retrieval computation, but were we were unable to find any issues in our code that would account for the difference.

We did not have access to the Jin et al. implementation, and were not able to undertake detailed diagnostics. Nevertheless, it seems likely that some other factor or component present in the software but not described in Jin et al.'s paper is causing the difference. Other possible sources of variance include the number of training epochs, the specific version of PubMedBERT that was used, and details of the mapping used to convert ordinal gain scores to numeric values used for top-weighted effectiveness calculation.

Model	Head (DCTR)			Torso (Raw)			Tail (Raw)		
	NDCG	RR	Recall	NDCG	RR	Recall	NDCG	RR	Recall
Reported results on test set (Jin et al. [11, Table 1])									
LADER (total)	0.338	0.664	0.304	0.303	0.427	0.353	0.310	0.306	0.449
LADER w/o DR	0.324	0.649	0.284	0.266	0.396	0.298	0.232	0.236	0.330
LADER w/o LA	0.247	0.532	0.237	0.241	0.350	0.293	0.260	0.257	0.394
Reproduced results on test set									
LADER (total)	0.313	0.591	0.266	0.286	0.384	0.307	0.249	0.234	0.360
LADER w/o DR	0.336	0.585	0.265	0.310	0.373	0.289	0.229	0.214	0.310
LADER w/o LA	0.105	0.220	0.094	0.138	0.189	0.157	0.129	0.114	0.212
Results on validation set									
LADER (total)	0.312	0.588	0.261	0.294	0.392	0.310	0.258	0.236	0.387
LADER w/o DR	0.336	0.587	0.257	0.320	0.390	0.294	0.238	0.215	0.330
LADER w/o LA	0.116	0.238	0.102	0.142	0.185	0.169	0.150	0.136	0.237

Table 1: Reported, reproduced, and validation results for LADER for three types of queries. The values in the "Reported" rows are taken directly from Jin et al. [11]; other values are from our reimplementation. All three effectiveness measures are evaluated at depth $k = 10$.

Overall, we can regard our reproduction of the experiments described by Jin et al. [11] as being only a moderate success. Our test code and experimental harness are available.

4 Extension Beyond Reproducibility

We now extend the investigation by exploring additional enhancements to dense retrieval performance, measuring the impact of increased training data, alternative pretrained models, untrained encoders, and of query reformulation behaviors. We also include results for Head queries using the Raw judgments, adding to the previous Head DCTR, Torso Raw, and Tail Raw measurements.

Scaling Training Data . The training configuration reported for the query encoder and document encoder specifies 20,000 training steps with a batch size of 256, that is, using approximately 5 million triplet records. The results presented in Table 1 follow that regime.

To assess the impact of increased training data, we also used the full 10 million triplet training records from the TripClick dataset. Results appear in rows i, j, and k of Table 2. When compared to the corresponding reference rows (f, g, and h, indicated by the row labels in the rightmost column) it is clear that adding further training records improves performance, with "*" annotations representing significant gains compared to the corresponding 5M training configuration (Student two-tailed t-test, $p < 0.05$, with the test applied independently for each effectiveness metric and each test environment, and with no multiple-comparison adjustment necessary in this usage). That is, doubling the number of training triples from 5M to 10M results in effectiveness gains for all metrics and all query categories (Head with both types of qrels; Torso; and Tail).

Alternative Pretrained Model. In the initial training setup the query encoder and document encoder were initialized with PubMedBERT and then fine-tuned on TripClick triplet-based training data. To confirm the need for domain-specific pretraining, we conducted an experiment in which we initialized the encoders with BERT, a general-purpose language model that has *not* been adapted to the biomedical domain. As expected, effectiveness markedly decreased, reinforcing the importance of domain-specific pretraining. For space reasons we do not include those results in Table 2.

To further explore the role of pretrained models in dense retrieval, we next experimented with Biomedical Contrastive Pretrained Transformers (MedCPT), a domain-specific model trained on 255 million query-article pairs collected from PubMed search logs between 2020 and 2022 [10]. MedCPT is initialized from PubMedBERT and subsequently fine-tuned on PubMed biomedical corpora, making it a strong candidate for improving dense retrieval performance on our health-related dataset.

The training configuration for MedCPT-initialized encoders follows the same setup as models trained with PubMedBERT, see Section 3. Table 2 lists a range of MedCPT-based results, including a block of measurements (rows l, m, and n) that employs the full 10M training triplets. As can be seen, the two models (one initialized with PubMedBERT, the other with MedCPT) yield very similar behaviors, suggesting that the exact training content does not affect final retrieval performance provided it is biomedical in nature.

Evaluating Untrained Models. We also measured untrained models, without fine-tuning them on the TripClick triples. In these experiments the untrained PubMedBERT model, and untrained MedCPT model, are used directly as query and document encoders. The pure dense retrieval performance (LADER w/o LA) for untrained PubMedBERT was close to zero across all measurement configurations, and are not included in Table 2. When comparing the performance of untrained PubMedBERT with fine-tuned PubMedBERT using 10M triples (that is, row i versus row a, and row j versus row b), it is clear that fine-tuning using the triplet data is a key contributor to the LADER performance, across all of the three query types (Head, Torso, and Tail), and across all evaluation scenarios. Similar outcomes occur for MedCPT (row l versus row c; and row m

Id.	Model	Head (Raw)			Head (DCTR)			Torso (Raw)			Tail (Raw)			Rf.
		NDCG	RR	Recall	NDCG	RR	Recall	NDCG	RR	Recall	NDCG	RR	Recall	
Untuned PubMed														
a	LADER (total)	0.098	0.140	0.039	0.083	0.122	0.050	0.006	0.005	0.008	0.017	0.014	0.024	–
b	LADER w/o DR	0.328	0.516	0.137	0.284	0.465	0.168	0.173	0.197	0.137	0.082	0.071	0.107	–
Untuned MedCPT														
c	LADER (total)	0.182	0.324	0.110	0.135	0.273	0.127	0.174	0.238	0.216	0.217	0.199	0.340	–
d	LADER w/o DR	0.376	0.571	0.207	0.298	0.510	0.241	0.290	0.345	0.279	0.212	0.198	0.285	–
e	LADER w/o LA	0.182	0.324	0.111	0.135	0.273	0.127	0.171	0.234	0.214	0.217	0.199	0.340	–
Fine-tuned PubMed – 5M														
f	LADER (total)	0.414*	0.661*	0.224*	0.313*	0.591*	0.266*	0.286*	0.384*	0.307*	0.249*	0.234*	0.360*	a
g	LADER w/o DR	0.428*	0.655*	0.223*	0.336*	0.585*	0.265*	0.310*	0.373*	0.289*	0.229*	0.214*	0.310*	b
h	LADER w/o LA	0.149	0.273	0.088	0.105	0.220	0.094	0.138	0.189	0.157	0.129	0.114	0.212	–
Fine-tuned PubMed – 10M														
i	LADER (total)	0.414	0.657	0.229*	0.312	0.591	0.271*	0.287	0.386	0.321*	0.266*	0.248*	0.393*	f
j	LADER w/o DR	0.429	0.653	0.227	0.337	0.589	0.267	0.311	0.373	0.292	0.231	0.214	0.310	g
k	LADER w/o LA	0.189*	0.329*	0.111*	0.129*	0.257*	0.121*	0.183*	0.250*	0.220*	0.180*	0.164*	0.283*	h
Fine-tuned MedCPT – 10M														
l	LADER (total)	0.415*	0.660*	0.228*	0.309*	0.591*	0.268*	0.283*	0.377*	0.320*	0.264*	0.254*	0.378*	c
m	LADER w/o DR	0.430*	0.653*	0.226*	0.338*	0.590*	0.267*	0.313*	0.371*	0.291*	0.237*	0.222*	0.320*	d
n	LADER w/o LA	0.178	0.309	0.104	0.124	0.254	0.115	0.166	0.236	0.199	0.164	0.148	0.265	e
Fine-tuned MedCPT + Reformulation Augmentation – 10M														
o	LADER (total)	0.416	0.663	0.229	0.312*	0.591	0.270	0.284	0.376	0.321	0.263	0.254	0.378	l
p	LADER w/o DR	0.433*	0.660*	0.226	0.341*	0.594	0.268	0.311	0.366	0.293	0.235	0.219	0.318	m
q	LADER w/o LA	0.178	0.309	0.104	0.124	0.254	0.115	0.166	0.236	0.199	0.164	0.148	0.265	n

Table 2: Extended results, adding to Table 1, with each section a further round of experimentation. Statistical significance tests were conducted independently for corresponding pairs of systems within columns. Values marked $*$ indicate statistically significant improvements ($p < 0.05$) compared to the corresponding value in the same column in the reference row, indicated for each target row by the label in column "Rf". Rows f, g, and h are the same system and experiments as are shown in the middle group in Table 1. All evaluations are to depth $k = 10$.

versus row d). However, in contrast to PubMedBERT, the performance of pure dense retrieval (LADER w/o LA) remains relatively unchanged. These findings suggest that using pretrained encoders alone, without adaptation to the target retrieval task, significantly underperforms compared to task-specific fine-tuning.

Comparing PubMedBERT-based and MedCPT-based arrangements, the improvement in retrieval performance from untrained PubMedBERT to its fine-tuned version is greater than the improvement seen from untrained MedCPT to its fine-tuned version. This can be attributed to the fact that MedCPT has already undergone extensive pretraining on PubMed articles, aligning it with the biomedical retrieval task even before fine-tuning using triples is carried out. In contrast, untrained PubMedBERT performs poorly, and should not be used without fine-training.

Incorporating Reformulation Behavior. The augmentation approach of Jin et al. [11] means that documents receive score boosts if they are clicked for any previous instance of the query. Our proposal here is to incorporate prior clicks via frequency counts, expecting that documents clicked multiple times are more likely to be appropriate responses. Moreover, users often iteratively refine their queries to better express their information needs, and proceed via *search sessions* rather than individual queries, adjusting terms until they retrieve suitable results. That is, we further hypothesize that incorporating query reformulation interactions – a second form of implicit feedback – will enhance retrieval effectiveness. Prior work has also explored the role of query reformulation in clinical information retrieval settings [1, 27], providing additional motivation for our use of such signals.

To allow exploration of this proposal, we turn to the click logs provided by Rekabsaz et al. [23], which contain anonymized user interactions. Each record in this log represents a click event on a search result page, and contains a clicked document identifier, a session identifier, and a timestamp. We define that dataset to be a collection of user search sessions, where each session represents a sequence of queries submitted by the same user during a continuous interaction period with the search system. Let $\mathcal{S} = [S_1, S_2, \ldots, S_i, \ldots, S_m]$ denote the set of m available sessions, with S_i

the i th one. Each session S_i consists of a sequence of user queries, ordered based on their timestamps: $S_i = [q_{i,1}, q_{i,2}, \ldots, q_{i,\ell(i)}]$, where $\ell(i)$ is the length of session S_i (number of queries), and $q_{i,j}$ is the j th query in session S_i. This notation explicitly represents queries based on their position in the session.

The dataset can also be thought of as individual queries, collated without regard to the session boundaries. Overloading the notation somewhat, we thus also have $\mathcal{Q} = \{q_1, q_2, \ldots, q_n\}$ as the set of n unique queries in the dataset. Any given query $q \in \mathcal{Q}$ might appear in multiple sessions, and in a different position in each. We define the set of sessions that contain query q as:

$$sessions(q) = \{S_i \mid S_i \in \mathcal{S} \wedge q \in S_i\}\,.$$

In the Jin et al. approach (Section 3) receipt of a query q leads to retrieval of similar documents using the document encoder and similar queries from $\mathcal{Q}$ using the query encoder. To incorporate session-level reformulation interactions we instead: (1) find $sessions(q')$, the set of sessions in which q' appears; (2) form the union of the sets of clicked documents across each of those sessions; and (3) weight those clicked documents by their relative click frequencies.

Implementing this first method did not yield observable improvements. To refine our approach, we then limited the click aggregation to focused contexts, only considering clicks from the queries immediately before and after occurrences of q' in any session. We then score documents (compare with Equation 6) by computing $score'_q(d, q)$ as

$$\sum_{q' \in Q_{1000}(q)} \log(1 + C'(d, q')) \cdot softmax(E(q') \odot E(q))\,, \qquad (8)$$

where $Q_{1000}(q)$ is again the set of 1000 most-similar queries to q, and where $C'(d, q')$ now represents the number of times that d was clicked for q' in the training data. We also define a new score, $score_s(d, q)$, to represent the scores derived from session interactions, denoted by $score_s(d, q)$ and calculated via:

$$\sum_{\substack{q' \in Q_{1000}(q) \wedge \\ q'' \in sessions(q')}} \gamma \cdot \log(1 + C'(d, q'')) \cdot softmax(E(q') \odot E(q))\,. \qquad (9)$$

We also explored an alternative in which q'' was restricted to the (at most) two queries adjacent to q' in the sessions that contain q', rather than ranging over all of the queries in those session. We set the balancing value to $\gamma = 0.5$ so that these "secondary" queries provide a reduced contribution. The final score is then similarly adjusted, and computed as:

$$score'(d, q) = score_r(d, q) + score'_q(d, q) + score_s(d, q)\,. \qquad (10)$$

The results of this session-based augmentation (using two neighboring queries from within each session) are the last group of results in Table 2. For these three rows (o, p, and q) the corresponding reference rows are the three 10M rows above, l, m, and n. Note that row q is, by definition, equal to row n. Comparing rows o and l, there are small (but not consistently statistically significant) gains in effectiveness across the various measurement scenarios, suggesting that extending the click-based augmentation across query sessions may indeed be surfacing new useful documents. On the other hand, the lack of consistency in the statistical relationships means that must regard these gains as being indicative rather than definitive. Even with that caveat, these results are encouraging.

Model	NDCG @10	RR @10	Recall @10	Recall @1000
BM25	0.199	0.347	0.128	0.806
RM3-PRF	0.198	0.350	0.124	0.804
LADER(total)	0.416*	0.663*	0.229*	0.829*

Table 3: Sparse retrieval versus dense retrieval. Values in the third row are significantly better than those in both other rows (Student t-test, $p < 0.05$).

5 Discussion and Conclusion

We now discuss the results of our LADER reproducibility study, and consider possible extensions to the log-augmentation process.

Reproduction. Our study sought to replicate and extend the findings of Jin et al. [11], utilizing various configurations of the LADER system. The reproducibility results highlighted in Table 1 indicate a broad agreement with the baseline established by Jin et al., but also expose some consistent differences. Specifically, our implementation of the dense retrieval module (LADER w/o LA) underperformed relative to the Jin et al. results, indicating that in some critical dimension our underpinning dense retriever is not an accurate reflection of their implementation. We have explored a wide range of alternative parameter settings while seeking to understand that difference, but been unable to arrive at a satisfactory explanation. We again note that the original LADER code was not available to us, meaning that there are multiple possible reasons for the differences in results, including the number of training epochs used, the version of PubMedBERT that was used, and specification of which 5M training triples were selected. There may also have been different ordinal-to-numeric mappings used as the qrels were prepared.

Extension. On the positive side, our study has shown that increasing the volume of training data enhances retrieval effectiveness across all measurement scenarios, emphasizing the importance of larger datasets in capturing subtle data patterns. Furthermore, employing different pretrained models, such as PubMedBERT and MedCPT, underscored the crucial role of domain-specific pretraining in model performance. Our findings suggest that the choice domain-specific pretraining (in our case, for the medical domain) does not markedly alter ultimate effectiveness, with PubMedBERT and MedCPT providing comparable retrieval quality.

We then explored the impact of reformulation augmentation on retrieval performance. This method is motivated by the observation that user intent tends to be consistent within a session, and queries formulated before and after a particular query are likely to be targeting the same information need. By incorporating these interactions, particularly the documents clicked in response to them, we can potentially capture a more holistic view of the user's intent. Our results indicate that incorporating these session-based interactions does indeed lead to small measurable effectiveness gains, but not to the level that they were consistently significant. Confirming those gains with further experiments is an ongoing objective.

We have also compared our results with sparse retrieval methods using Pyserini [16], specifically evaluating performance against BM25 and RM3-PRF. Retrieval effectiveness averages for Head

queries and Raw relevance labels are reported in Table 3, with the LADER version taken from row o of Table 2.

As expected, the dense retrieval models outperform traditional sparse approaches in terms of bringing useful answer documents towards the head of the ranking. Note that while Recall@1000 is high for all three approaches, it is highest for the augmented regime. We have not considered retrieval efficiency in any way in this work, focusing only on effectiveness, but the Recall@1000 scores in Table 3 suggests that a two-phase process that employs a sparse mechanism at first and then a dense-based re-ranker might be a useful practical combination.

Additional Empirical Findings. Given that the untrained PubMedBERT (without fine-tuning on the TripClick triplet dataset) demonstrated poor retrieval performance, we further investigated its behavior. Specifically, since PubMedBERT is a pretrained model in the medical domain yet fails to perform well on TripClick data, which is also health-related, we were interested in exploring the extent to which it is tightly coupled to health applications.

To analyze this, we conducted a small experiment evaluating PubMedBERT's scoring process on datasets from five further domains: politics, fashion and arts, sports, history, and geography. For each of those five domains we curated a small dataset of 100 documents, sourced using the NEWS API.[4] Additionally, we randomly selected 100 queries from the TripClick test set, and 100 documents that were confirmed to be absent from the TripClick relevance judgments for all of the 100 queries, thereby ensuring that those documents were (probably) not suitable responses to any of the 100 queries.

For each medical query, we computed inner-product scores separately across the 100 medical documents (TripClick domain) to get a set of 10,000 scores. We then took 100 documents from each of the five non-medical domains, using the PubMedBERT encoder throughout, to get five further sets of 10,000 scores. While inner-product scores have no innate meaning, and are used as a basis for ordering documents rather than as absolute values, we would nevertheless anticipate that medical documents – even when unrelated to the query – would score more highly than (say) fashion documents when evaluated in response to medical queries.

Figure 2 shows that the converse is what occurs. Each line represents the kernel density plot of the average score distribution across 100 queries and the 100 documents across a total of six different domains – medical, plus five non-medical. The plot reveals that PubMedBERT assigns lower inner-product scores to non-relevant documents in the medical domain compared to documents in other domains. That is, while the untrained PubMedBERT is capable of differentiating between "useful" and "not useful" medical documents, it treats non-medical documents more generously in terms of scoring. In turn, that behavior may mean that application to collections that are of mixed content might result in out-of-topic documents also being assigned higher scores than useful on-topic ones, diluting the result rankings and lowering the scores.

Limitations and Future Directions. There are a number of aspects of this work that will benefit from further exploration. One area of concern is the reliance that is placed on the inferred relevance judgments, which are derived from user clicking behavior, and hence relatively sparse. In particular, this approach means that only a small number – just dozens of the 1.5 million TripClick documents – were "judged" for each query, and that all others were assumed to be non-useful. While manual annotations at the scale that would be required are probably too costly to be feasible, we are encouraged by recent progress in automatic labeling, see, for example, the work of Thomas et al. [28].

There are also aspects of the log-augmentation process that present opportunities. For example, Jin et al. chose to select 1000 "nearby" queries to use as the basis for query-time click evidence. But there seems no reason for that to be a fixed value (nor for it to be so large), and it might well be that a more sensitive selection process based on the actual similarity score, or the use of decreasingly weighted multipliers so that the effect decays with distance from the underlying query q, will provide more nuanced signals. We look forward to exploring such areas as we continue this project.

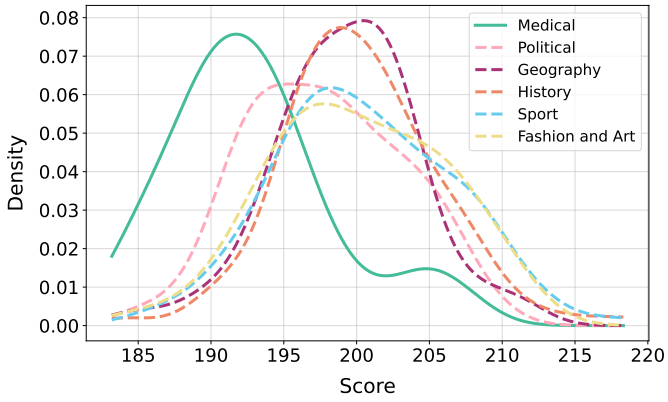

Figure 2: Average retrieval score distribution for 100 medical queries, scored first against 100 unrelated medical documents, and then against sets of 100 documents drawn from each of five other topical domains.

Acknowledgment. This work was supported by the Australian Research Council's *Discovery Projects* Scheme (project DP190101113) and by The University of Melbourne's Research Computing Services Petascale Campus Initiative. We are also grateful to the TripClick benchmark organizers for sharing the TripClick dataset. We thank the anonymous referees for their support.

Software. The source code developed in the course of this study is publicly available at https://github.com/reyhane-goli/SIGIR-2025.

References

[1] M. Agosti, G. M. D. Nunzio, and S. Marchesin. An analysis of query reformulation techniques for precision medicine. In *Proc. SIGIR*, pages 973–976, 2019.

[2] A. Aizawa. An information-theoretic perspective of TF–IDF measures. *Inf. Proc. & Man.*, 39(1):45–65, 2003.

[3] P. Bailey, A. Moffat, F. Scholer, and P. Thomas. Retrieval consistency in the presence of query variations. In *Proc. SIGIR*, pages 395–404, 2017.

[4] M. J. Bates. Information search tactics. *J. Am. Soc. Inf. Sc.*, 30(4):205–214, 1979.

[4] https://newsapi.org/docs/endpoints/everything

[5] G. B. Duggan and S. J. Payne. Knowledge in the head and on the web: Using topic expertise to aid search. In *Proc. CHI*, page 39–48, 2008.

[6] L. Gao and J. Callan. Unsupervised corpus aware language model pre-training for dense passage retrieval. In *Proc. ACL*, pages 2843–2853, 2022.

[7] Y. Gu, R. Tinn, H. Cheng, M. Lucas, N. Usuyama, X. Liu, T. Naumann, J. Gao, and H. Poon. Domain-specific language model pretraining for biomedical natural language processing. *J. Trans. Comput. Healthc.*, 3(1):1–23, 2021.

[8] S. Hassantabar, Z. Wang, and N. K. Jha. SCANN: Synthesis of compact and accurate neural networks. *IEEE Trans. Comp. Aided Des. Integr. Circuits Syst.*, 41 (9):3012–3025, 2021.

[9] S. Hofstätter, S. Althammer, M. Sertkan, and A. Hanbury. Establishing strong baselines for TripClick health retrieval. In *Proc. ECIR*, pages 144–152, 2022.

[10] Q. Jin, W. Kim, Q. Chen, D. C. Comeau, L. Yeganova, W. J. Wilbur, and Z. Lu. MedCPT: Contrastive pre-trained transformers with large-scale PubMed search logs for zero-shot biomedical information retrieval. *Bioinform.*, 39(11):btad651, 2023.

[11] Q. Jin, A. Shin, and Z. Lu. LADER: Log-augmented dense retrieval for biomedical literature search. In *Proc. SIGIR*, pages 2092–2097, 2023.

[12] J. Johnson, M. Douze, and H. Jégou. Billion-scale similarity search with GPUs. *IEEE Trans. Big Data*, 7(3):535–547, 2019.

[13] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W. Yih. Dense passage retrieval for open-domain question answering. In *Proc. EMNLP*, pages 6769–6781, 2020.

[14] O. Khattab and M. Zaharia. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proc. SIGIR*, pages 39–48, 2020.

[15] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arxiv.org/abs/1412.6980*, 2014.

[16] J. Lin, X. Ma, S. Lin, J. Yang, R. Pradeep, and R. Nogueira. Pyserini: A python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proc. SIGIR*, pages 2356–2362, 2021.

[17] S. Lin, J. Yang, and J. Lin. In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval. In *Proc. RepL4NLP*, pages 163–173, 2021.

[18] Y. Lv and C. Zhai. A comparative study of methods for estimating query language models with pseudo feedback. In *Proc. CIKM*, pages 1895–1898, 2009.

[19] C. Macdonald, N. Tonellotto, and I. Ounis. On single and multiple representations in dense passage retrieval. In *Proc. IIR*, 2021.

[20] G. Marchionini, X. Lin, and S. Dwiggins. Effects of search and subject expertise on information seeking in a hypertext environment. *Proc. Assoc. Inf. Sc.*, 27: 129–142, 1990.

[21] B. Mitra and N. Craswell. An introduction to neural information retrieval. *Found. Trnd. Inf. Retr.*, 13(1):1–126, 2018.

[22] A. Moffat, P. Bailey, F. Scholer, and P. Thomas. Incorporating user expectations and behavior into the measurement of search effectiveness. *ACM Trans. Inf. Sys.*, 35(3):24.1–24.38, 2017.

[23] N. Rekabsaz, O. Lesota, M. Schedl, J. Brassey, and C. Eickhoff. TripClick: The log files of a large health web search engine. In *Proc. SIGIR*, pages 2507–2513, 2021.

[24] R. Ren, Y. Qu, J. Liu, W. X. Zhao, Q. She, H. Wu, H. Wang, and J.-R. Wen. RocketQAv2: A joint training method for dense passage retrieval and passage re-ranking. In *Proc. EMNLP*, pages 2825–2835, 2021.

[25] S. Robertson and H. Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Found. Trnd. Inf. Retr.*, 3(4):333–389, 2009.

[26] F. Schroff, D. Kalenichenko, and J. Philbin. FaceNet: A unified embedding for face recognition and clustering. In *Proc. CVPR*, pages 815–823, 2015.

[27] L. Soldaini, A. Yates, and N. Goharian. Learning to reformulate long queries for clinical decision support. *J. Assoc. Inf. Sci. Technol.*, 68(11):2602–2619, 2017.

[28] P. Thomas, S. Spielman, N. Craswell, and B. Mitra. Large language models can accurately predict searcher preferences. In *Proc. SIGIR*, pages 1930–1940, 2024.

[29] W. Waller and D. Kraft. A mathematical model of a weighted Boolean retrieval system. *Inf. Proc. & Man.*, 15(5):235–245, 1979.

[30] R. W. White, S. T. Dumais, and J. Teevan. Characterizing the influence of domain expertise on web search behavior. In *Proc. WSDM*, page 132–141, 2009.

[31] B. M. Wildemuth. The effects of domain knowledge on search tactic formulation. *J. Am. Soc. Inf. Sc. Tech.*, 55(3):246–258, 2004.

[32] L. Xiong, C. Xiong, Y. Li, K.-F. Tang, J. Liu, P. N. Bennett, J. Ahmed, and A. Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *Proc. ICLR*, 2021.

[33] J. Zhan, J. Mao, Y. Liu, J. Guo, M. Zhang, and S. Ma. Optimizing dense retrieval model training with hard negatives. In *Proc. SIGIR*, pages 1503–1512, 2021.

[34] W. X. Zhao, J. Liu, R. Ren, and J. Wen. Dense text retrieval based on pretrained language models: A survey. *ACM Trans. Inf. Sys.*, 42(4):1–60, 2024.

[35] S. Zhuang, H. Li, and G. Zuccon. Implicit feedback for dense passage retrieval: A counterfactual approach. In *Proc. SIGIR*, pages 18–28, 2022.

[36] J. Zobel and A. Moffat. Inverted files for text search engines. *ACM Comp. Surv.*, 38(2):6.1–6.56, 2006.