Context-Aware Query Term Difficulty Estimation for Performance Prediction

Abbas Saleminezhad [1], Negar Arabzadeh [2], Soosan beheshti [1], and
Ebrahim Bagheri [1]

[1] Toronto Metropolitan University, Toronto ON, Canada
{abbas.saleminezhad, soosan ,bagheri}@torontomu.ca
[2] University of Waterloo, Toronto ON, Canada
narabzad@uwaterloo.ca

Abstract. Research has already found that many retrieval methods are sensitive to the choice and order of terms that appear in a query, which can significantly impact retrieval effectiveness. We capitalize on this finding in order to predict the performance of a query. More specifically, we propose to learn query term difficulty weights specifically within the context of each query, which could then be used as indicators of whether each query term has the likelihood of making the query more effective or not. We show how such difficulty weights can be learnt through the finetuning of a language model. In addition, we propose an approach to integrate the learnt weights into a cross-encoder architecture to predict query performance. We show that our proposed approach shows a consistently strong performance prediction on the MSMARCO collection and its associated widely used Trec Deep Learning tracks query sets. Our findings demonstrate that our method is able to show consistently strong performance prediction over different query sets (MSMARCO Dev, TREC DL'19, '20, Hard) and a range of evaluation metrics (Kendall, Spearman, sMARE).

1 Introduction

With the diverse range of user queries, a single Information Retrieval (IR) method faces challenges in effectively addressing all query types. Certain retrieval methods excel for specific queries but may fall short for others [6,12,39,7,1]. To assess how well a retrieval method can meet a query's needs, researchers have delved into the realm of *Query Performance Prediction (QPP)*. The primary aim of QPP is to predict the potential retrieval effectiveness of a method for a given query [11,22,41,33,36,4,3,27,24,23]. Numerous QPP methods exist in the literature, broadly categorized as *pre-retrieval* and *post-retrieval* methods. Post-retrieval methods, despite their superior performance, incur additional overhead by necessitating the complete retrieval of the query for estimating effectiveness [11]. On the other hand, pre-retrieval methods are lightweight, concentrating solely on query and document collection characteristics [20]. Most recent pre-retrieval methods focus on injecting external knowledge into performance estimation by benefiting from contextual language models. For instance, the work by

Arabzadeh et al. [8,9] advocates for the use of neural-embedding representation of the query to determine query *specificity*, as an indicator of query performance. Roy et al. [35] also promote the idea of using contextual embeddings to measure the *ambiguity* of a query by estimating the number of senses each query term is associated with.

Similar to the works by Arabzadeh et al. and Roy et al. [10,9,35,5], we also benefit from contextual embeddings; however, in contrast rather than estimating *query specificity* or *ambiguity*, we are interested in using contextual embeddings to learn the impact of query terms on the overall query performance. Our proposed approach builds on the foundational premise that certain terms in the query and document spaces have a higher impact on retrieval effectiveness [15,16]. These terms are often more discriminative and can hence more effectively discern between relevant and irrelevant documents to a given query. For this reason and during the retrieval process, such terms would need to play a more important role and have a higher weight. On the same basis, there are terms that negatively impact the performance of a query and would hence need to receive a lower weight. We build on this premise and propose to learn weights for terms in the query space so as to understand which terms have the potential to contribute positively or negatively to query performance. A query with a large number of terms that can positively contribute to retrieval effectiveness is more likely to be an easier query, whereas conversely, a query with many terms with a negative prospect would be harder queries [2,37]. We have extensively evaluated our proposed approach based on four widely-used MSMARCO query sets, namely Dev set [28], TREC DL 2019 [14], DL 2020 [13], and DL Hard [26]. We show that our approach has strong performance compared to the baselines on various evaluation metrics and query sets. For reproducibility purposes, we made our code and models publicly available at https://github.com/Saleminezhad/context-aware-qpp.git.

2 Proposed Approach

Objective. The goal of our work in this paper is to propose a *pre-retrieval query performance predictor*, denoted by $\mu(q, C)$, where q and C represent a query, and a collection of documents, respectively. This predictor would need to estimate the performance of query q with respect to a specific IR evaluation metric M, resulting in an estimated performance score represented as $\widehat{M_q}$ [11].

Approach Overview. We are interested in estimating which query terms are likely to impact the performance of a query positively or negatively. Terms that positively impact the performance of a query can be seen as softer terms, while terms with a negative impact would be harder. To distinguish between soft and hard terms, we can compare pairs of queries expressing the same information need but differently. In essence, such two queries could be the same from an information seeking objective perspective, but in practice, their retrieval effectiveness could (potentially vastly) differ from each other. Given two such queries and their retrieval effectiveness, it would be possible to determine which query is softer and which is harder. Consequently, depending on the terms in each query and how overlapping they are between the two queries, one could also make inferences about whether and to what extent query terms can impact query performance and hence be considered soft or hard. Our work in this paper offers a systematic approach to identify a collection of comparable query pairs based on which the likelihood of soft or hard query terms are learnt. Once the likelihood of a query term contributing to query performance is learnt, we incorporate this information to predict the performance of the query on a pre-retrieval basis.

Methodology. Our proposed work consists of three main steps, namely: (1) developing a collection of comparable pairs of queries that address the same information need but have varying retrieval effectiveness, (2) given the pairs of queries, learning the likelihood of query terms contributing to the softness or hardness of the query, and (3) adopting the learnt term likelihood information to estimate query performance. We provide the details of each step in the following.

Developing Comparable Query Pairs. Here, the objective is to develop a collection of comparable query pairs that address the same information need but the performance of the queries in each pair is not the same. To achieve this goal, we are inspired by the DocT5Query [30,31], which has suggested that a translation function can be learned based on the T5 transformer to map documents to queries [32]. The idea is simple yet intuitive: given a collection of relevance judgements, one can finetune a T5 transformer to learn to generate the query from its relevant document. Once the transformer is finetuned, one could then use it to generate queries from any given document. In the context of our work, we adopt a similar strategy where we finetune a T5 transformer architecture based on a large relevance judgment collection. Using the finetuned T5 transformer and for the documents in the MSMARCO collection, we generate multiple queries per document. The queries generated for each document would be addressing the same information need as they have been generated for the same document but their degree of retrieval effectiveness is not the same. We create pairs of such queries where queries in each pair have differing effectiveness.

More formally, let $C = \{d_1, d_2, ..., d_m\}$ be a collection of m documents. Let us assume that a T5 transformer architecture can be finetuned, as outlined in [30,29], to serve as translation function $\mathcal{T} : \mathcal{D} \rightarrow Q$, to facilitate the transition from the document space to the query space. With $\mathcal{T}$, we can generate queries for documents in $\mathcal{D}$ where each query q_d^q is seeking information from d. Succinctly, given a document d and the translation function $\mathcal{T}$, we generate q_d^g as $\mathcal{T}(d) = q_d^g$. Now, for any query q and its relevant judged document d_q, it is possible to generate alternative variations for q through $\mathcal{T}(d_q)$. Based on $\mathcal{T}(d_q)$, we develop query pairs (q, q') where $q' \in \mathcal{T}(d_q)$ and $M'_q \neq M_q$.

Learning Query Term Weights. Using the created pairs of queries, we aim to learn query term weights that signify the likelihood of terms influencing the query's *softness* or *hardness*. We utilize contextualized word embeddings for q and q', facilitating the prediction of term difficulty weights via linear regression. To discern a term's difficulty within a given query, we employ an attention mechanism, allowing the term to gradually incorporate contextual information from its interactions with other terms in the same query. Let us define $TD(q_t)$ to denote the term difficulty weight of query term q_t as follows when $M'_q > M_q$:

$$\mathrm{TD(q}_t\mathrm{)} = \begin{cases} -1 & \text{if } q_t \in q \text{ and } q_t \notin q' \\ 1 & \text{if } q_t \notin q \text{ and } q_t \in q' \\ 0 & \text{if } q_t \in q \text{ and } q_t \in q' \end{cases} \tag{1}$$

Equation 1 illustrates that terms present in more challenging queries have a higher likelihood of lowering query performance, whereas terms contributing to improved query performance could be considered to be easier query terms. We utilize contextualized word embeddings representing query terms and their associated weights to train a linear regression model. This model predicts $TD(q_t)$ for each query term q_t in query q. We train a model via per-token regression, aiming to minimize the Mean Squared Error (MSE) between predicted weights $\widehat{TD}(q_t)$ and target weights $TD(q_t)$ derived from Equation 1. In other words, the model's goal is to minimize loss MSE = $\sum_{q_t \in q} \left(TD(q_t) - \widehat{TD}(q_t)\right)^2$. The regression model will be able to predict $\widehat{TD}(q_t)$ for any term within a query.

Pre-Retrieval Predictor. The objective of the final step is to incorporate term difficulty weights for performance prediction. To do so, we develop two *term sets* based on the term difficulty weights predicted for each term in the query. The two term sets represent soft, $\phi^+(q)$, and hard, $\phi^-(q)$, terms, respectively:

$$\phi^+(q) = \{\Omega(q_t) \,|\, T\hat{D}(q_t) > 0\}, \qquad \phi^-(q) = \{\Omega(q_t) \,|\, T\hat{D}(q_t) < 0\} \tag{2}$$

where $\Omega()$ is a weighting function based on term difficulty weight. We adopt the weighting function in [15] to implement Ω. Given the two sets $\phi^+(q)$ and $\phi^-(q)$ for each query q, we utilize a cross-encoder architecture to estimate the performance of a query directly. To achieve this, we feed both the weighted concatenated representations of easy query terms ($\phi^+(q)$) and hard query terms ($\phi^-(q)$) into a cross-encoder network and train this network for the efficient development of μ . The goal of this network is to learn a continuous difficulty score M_q by examining the relationship between the weighted representation of the predicted easy query terms $\phi^+(q)$ and hard query terms $\phi^-(q)$. This is achieved by concatenating the query terms w.r.t their weights as suggested in [15,16]. Subsequently, we apply a linear layer to the initial vector generated by the transformer, resulting in a scalar value as $\mu(\phi^+(q), \phi^-(q))$. To further refine the network's performance, we employ a sigmoid layer σ in conjunction with a one-class Binary cross-entropy loss function l. More formally, the loss function for our cross-encoder network can be defined as follows:

$$\begin{aligned} l\Big(\mu(\phi^+(q), \phi^-(q)), M(q)\Big) = -\,\big[&M(q) \cdot \log\ \sigma(\mu(\phi^+(q), \phi^-(q))) \\ &+ (1 - M(q)) \cdot \log\ (1 - \sigma(\mu(\phi^+(q), \phi^-(q))))\big] \end{aligned}$$

where μ is our proposed pre-retrieval query performance predictor.

Table 1: Performance comparison between our proposed approach and other pre-retrieval QPP baselines over MS MARCO Dev on MRR@10 and TREC DL 2019,Trec DL 2020 and TREC DL HARD on ndcg@10 in terms of sMARE, Kendall τ and Spearman ρ. The highest value in each column is in bold.

QPP	**MS MARCO Dev**			**TREC DL 2019**			**TREC DL 2020**			**DL Hard**		
Method	$K-\tau$	$S-\rho$	sMARE	$K-\tau$	$S-\rho$	sMARE	$K-\tau$	$S-\rho$	sMARE	$K-\tau$	$S-\rho$	sMARE
IDF	0.116	0.154	0.330	0.158	0.245	0.321	0.245	0.353	0.374	0.116	0.152	0.342
VAR	0.062	0.083	0.333	0.107	0.152	0.290	0.059	0.077	0.318	0.016	0.035	0.349
PMI	0.017	0.023	0.323	0.009	0.017	0.341	0.040	0.056	0.344	0.022	0.031	0.349
SCS	0.037	0.049	0.333	**0.194**	**0.287**	0.316	0.272	0.397	0.333	0.106	0.140	0.326
SCQ	0.011	0.014	0.334	0.116	162	0.387	0.076	0.132	0.365	0.127	0.179	0.369
ICTF	0.114	0.152	0.330	0.153	0.240	0.360	0.345	0.330	0.330	0.107	0.115	0.314
DC	0.107	0.144	0.333	0.095	0.053	0.293	0.091	0.035	0.327	0.123	0.165	0.335
CC	0.065	0.085	0.333	0.099	0.055	0.319	0.106	0.026	0.327	0.103	0.141	0.310
IEF	0.094	0.104	0.330	0.187	0.166	0.387	0.064	0.081	0.334	0.140	0.191	0.377
our model	**0.303**	**0.401**	**0.321**	0.158	0.224	0.320	**0.290**	**0.423**	**0.314**	**0.355**	**0.492**	**0.284**
ours - $\phi^{-}(q)$	0.297	0.402	0.335	0.114	0.176	0.334	0.189	0.356	0.327	0.305	0.424	0.305
ours - $\phi^{+}(q)$	0.288	0.400	0.336	0.105	0.124	0.352	0.204	0.350	0.383	0.300	0.427	0.310

3 Experiments

3.1 Data

We employ the MSMARCO passage collection dataset [28], featuring 8.8 million passages and over 500,000 queries, each associated with at least one relevance-judged document. Following [30], we fine-tune the T5 transformer with default settings to create the translation function $\mathcal{T}$ [32] from the MSMARCO collection. Using $\mathcal{T}$, we generate queries for passages with conditions $M'_q > M_q$ and $M'_q = 1$, where q' is a generated query for q based on the relevant document. This results in 188,398 query pairs, used to train the regression model for predicting $TD()$. We evaluate our approach on four widely used query sets: MSMARCO Development set (Dev set, 6,980 queries), TREC DL 2019 [14] (43 queries), TREC DL 2020 [13] (53 queries), and DL-Hard [26] (50 queries).

3.2 Evaluation Metrics

The QPP evaluation involves correlating predicted and actual query performance [22,11] using Kendall's τ and Spearman ρ coefficients, and the scaled Mean Absolute Relative Error (sMARE) [18]. Higher Spearman and Kendall and lower sMARE values indicate better prediction accuracy. We assess based on predicting BM25 performance using official metrics MRR@10 for MS MARCO dev set and nDCG@10 for the other datasets [38].

3.3 Baselines

For the sake of comparative analysis with the state of the art, we adopt the following *pre-retrieval QPP* baselines including term-frequency baselines which utilize index statistics, including such as IDF [25] and ICTF [25]. The Simplified Clarity Score (SCS) metric [21] measures query specificity through Kullback-Leibler divergence, while SCQ [40] introduces vector-space-based query and collection similarity. Pointwise Mutual Information (PMI) analyzes term co-occurrence [19], and VAR assesses coherency based on term weight distributions [40]. From the neural-embedding based baselines, we include CC , DC , and IEF [8] that operate on term specificity. We note that for metrics that require an aggregation function, the best aggregator (average, maximum or minimum) was chosen based on the best performance of another set (DL 2019 for DL 2020, Dev for DL Hard, and vice versa).

3.4 Experimental Setup

We adopted the BERT-base-uncased [17] and got it fine-tuned for the weight prediction as the regression task for 10 epochs with a learning rate of 2e-5 and the maximum input length for queries was set to 9 (covering the maximum query length of 90% of queries in MSMARCO). For the Cross-encoder training, we employed the SentenceTransformer library [34]. This architecture underwent one epoch of training on the query pairs generated based on the finetuned T5 transformer ($\mathcal{T}$), with a batch size of 8.

3.5 Findings

We make the following observations based on the reported results in Table 1 : **(1)** we find that our approach shows the best performance on all three metrics over three of the query sets, namely MSMARCO Dev, DL 2020 and DL Hard. On the DL 2019 set, while competitive, our method does not show the best performance. However, we note that on DL 2019, there is no single baseline that shows the best performance on all metrics. Specifically, `VAR` shows the best performance on sMARE while `SCS` exhibits a stronger performance on Kendall and Spearman correlations. **(2)** In contrast to the baseline methods, our proposed approach consistently maintains robust performance across all three query sets. Notably, stronger baselines like `SCS` exhibit strong performance in specific query sets, such as TREC DL 2019 and 2020, but fall short in competitiveness on MARCO dev and DL HARD. A similar trend is observed for `IDF`, which excels in TREC DL 2020 but lacks competitiveness in the other three query sets. Despite `SCS` and `IDF` outperforming other baselines, our proposed method surpasses them by a significant margin on MS MARCO Dev set and TREC DL HARD. For instance, on the MS MARCO Dev set, `IDF` and our proposed method show Kendall τ correlations of 0.116 and 0.303, respectively. On TRECDL 2020, `SCS` achieves a Spearman of 0.397, while our method achieves a higher correlation of 0.423. **(3)** On the MS MARCO dev set, all baseline methods report Kendall τ correlations below 0.12, which is negligible compared to our model's correlation of 0.303. Our method consistently demonstrates superior consistency and performance, indicating its robustness across diverse query subsets and evaluation strategies. **(4)** Neural-based baselines exhibit less impressive results across the four query

Table 2: Sample queries color-coded to show term difficulty weights. Darker blue color indicate softer terms, and darker red colors show harder terms. Terms with no background denote terms that are neither hard or soft.

product level activity define	define a multichannel radio
definition of capias issued on a background	how far back do employment background checks
what is the gas called that they give you at the dentist	calculate the mass in grams of 2.74 l of co gas

sets, potentially because they consider the embedding representation of each query term without fine-tuning contextual representations for this specific task. In contrast, our approach predicts a term difficulty weight for each query term, learned through a fine-tuning process over the BERT language model. Therefore, the representations used by our proposed approach may be better tailored for this purpose. **(5)** Considering the impact of $\phi^{+}(q)$ and $\phi^{-}(q)$ on our proposed method's overall performance (last two rows of both tables), we observe significant performance improvement when both sets are taken into account, particularly on DL 2019, 2020, and Hard. On MSMARCO Dev, our method's performance improves on Kendall τ correlation and sMARE metric with both sets, while the performance remains consistent on Spearman correlation.

Finally, to illustrate the learned query term difficulty weights, we color-coded six sample queries in Table 2. Each row includes two queries with at least one overlapping term. In the first row, the common term between the queries is 'define'. In the first row, both queries share the term 'define.' The user seeks definitions for two phrases. Our model recognizes that the phrases enhance retrieval, but the term 'define' diminishes effectiveness. This may be due to BM25, which seeks relevant documents based on query terms, yet documents with phrase definitions may lack the term 'define.' The second example consists of two queries where the common term between them is 'background'. In this case, our model has determined that this term does not have much impact on the first query but improves retrieval effectiveness if included along with 'employment' and 'checks'. In the third row, the shared term is 'gas' and we would like to show that our model considers context to decide on term weights. As seen in this example, 'gas' was considered a hard term on the right and soft on the left.This helps us understand how our model adapts term difficulty to different situations.

4 Concluding Remarks

In this paper, we have proposed to learn contextualized query term difficulty weights that can inform the process of query performance prediction. We have shown that query term weights can be learnt, through finetuning a contextual language model, that estimate how each term can possibly impact the difficulty of a query. Through extensive experiments on five widely used query sets, we have shown that our proposed approach is both effective and consistent for predicting the performance of a range of queries.

References

1. Arabzadeh, N., Bigdeli, A., Hamidi Rad, R., Bagheri, E.: Quantifying ranker coverage of different query subspaces. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 2298–2302 (2023)
2. Arabzadeh, N., Bigdeli, A., Seyedsalehi, S., Zihayat, M., Bagheri, E.: Matches made in heaven: Toolkit and large-scale datasets for supervised query reformulation. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management. pp. 4417–4425 (2021)
3. Arabzadeh, N., Bigdeli, A., Zihayat, M., Bagheri, E.: Query performance prediction through retrieval coherency. In: Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part II 43. pp. 193–200. Springer (2021)
4. Arabzadeh, N., Hamidi Rad, R., Khodabakhsh, M., Bagheri, E.: Noisy perturbations for estimating query difficulty in dense retrievers. In: CIKM (2023)
5. Arabzadeh, N., Khodabakhsh, M., Bagheri, E.: Bert-qpp: Contextualized pre-trained transformers for query performance prediction. In: CIKM (2021)
6. Arabzadeh, N., Mitra, B., Bagheri, E.: Ms marco chameleons: Challenging the ms marco leaderboard with extremely obstinate queries. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management. pp. 4426–4435 (2021)
7. Arabzadeh, N., Yan, X., Clarke, C.L.A.: Predicting efficiency/effectiveness trade-offs for dense vs. sparse retrieval strategy selection. CoRR **abs/2109.10739** (2021), https://arxiv.org/abs/2109.10739
8. Arabzadeh, N., Zarrinkalam, F., Jovanovic, J., Al-Obeidat, F., Bagheri, E.: Neural embedding-based specificity metrics for pre-retrieval query performance prediction. Information Processing & Management **57**(4), 102248 (2020)
9. Arabzadeh, N., Zarrinkalam, F., Jovanovic, J., Bagheri, E.: Neural embedding-based metrics for pre-retrieval query performance prediction. In: Advances in Information Retrieval: 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14–17, 2020, Proceedings, Part II 42. pp. 78–85. Springer (2020)
10. Arabzadeh, N., Zarrinkalam, F., Jovanovic, J., Bagheri, E.: Geometric estimation of specificity within embedding spaces. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management. pp. 2109–2112 (2019)
11. Carmel, D., Yom-Tov, E.: Estimating the query difficulty for information retrieval. Synthesis Lectures on Information Concepts, Retrieval, and Services **2**(1), 1–89 (2010)
12. Carmel, D., Yom-Tov, E., Darlow, A., Pelleg, D.: What makes a query difficult? In: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 390–397 (2006)
13. Craswell, N., Mitra, B., Yilmaz, E., Campos, D.: Overview of the TREC 2020 deep learning track. CoRR **abs/2102.07662** (2021), https://arxiv.org/abs/2102.07662
14. Craswell, N., Mitra, B., Yilmaz, E., Campos, D., Voorhees, E.M.: Overview of the trec 2019 deep learning track. arXiv preprint arXiv:2003.07820 (2020)
15. Dai, Z., Callan, J.: Context-aware sentence/passage term importance estimation for first stage retrieval. arXiv preprint arXiv:1910.10687 (2019)
16. Dai, Z., Callan, J.: Context-aware term weighting for first stage passage retrieval. In: Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval. pp. 1533–1536 (2020)

17. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
18. Faggioli, G., Zendel, O., Culpepper, J.S., Ferro, N., Scholer, F.: smare: a new paradigm to evaluate and understand query performance prediction methods. Information Retrieval Journal **25**(2), 94–122 (2022)
19. Hauff, C.: Predicting the effectiveness of queries and retrieval systems. In: SIGIR Forum. vol. 44, p. 88 (2010)
20. Hauff, C., Hiemstra, D., de Jong, F.: A survey of pre-retrieval query performance predictors. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008, Napa Valley, California, USA, October 26-30, 2008. pp. 1419–1420 (2008). https://doi.org/10.1145/1458082.1458311, https://doi.org/10.1145/1458082.1458311
21. He, B., Ounis, I.: Inferring query performance using pre-retrieval predictors. In: String Processing and Information Retrieval, 11th International Conference, SPIRE 2004, Padova, Italy, October 5-8, 2004, Proceedings. pp. 43–54 (2004). https://doi.org/10.1007/978-3-540-30213-1_5, https://doi.org/10.1007/978-3-540-30213-1_5
22. He, B., Ounis, I.: Query performance prediction. Information Systems **31**(7), 585–594 (2006)
23. Khodabakhsh, M., Bagheri, E.: Semantics-enabled query performance prediction for ad hoc table retrieval. Information Processing & Management **58**(1), 102399 (2021)
24. Khodabakhsh, M., Bagheri, E.: Learning to rank and predict: Multi-task learning for ad hoc retrieval and query performance prediction. Information Sciences **639**, 119015 (2023)
25. Kwok, K.L.: A new method of weighting query terms for ad-hoc retrieval. In: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'96, August 18-22, 1996, Zurich, Switzerland (Special Issue of the SIGIR Forum). pp. 187–195 (1996). https://doi.org/10.1145/243199.243266, https://doi.org/10.1145/243199.243266
26. Mackie, I., Dalton, J., Yates, A.: How deep is your learning: the dl-hard annotated deep learning dataset. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (2021)
27. Meng, C., Arabzadeh, N., Aliannejadi, M., de Rijke, M.: Query performance prediction: From ad-hoc to conversational search. arXiv preprint arXiv:2305.10923 (2023)
28. Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., Deng, L.: Ms marco: A human generated machine reading comprehension dataset. In: CoCo@ NIPS (2016)
29. Nogueira, R., Cho, K.: Passage re-ranking with bert. arXiv preprint arXiv:1901.04085 (2019)
30. Nogueira, R., Lin, J., Epistemic, A.: From doc2query to docTTTTTquery. Online preprint **6**, 2 (2019)
31. Nogueira, R., Yang, W., Lin, J., Cho, K.: Document expansion by query prediction. arXiv preprint arXiv:1904.08375 (2019)
32. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research **21**(140), 1–67 (2020), http://jmlr.org/papers/v21/20-074.html

33. Raiber, F., Kurland, O.: Query-performance prediction: setting the expectations straight. In: Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval. pp. 13–22 (2014)
34. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084 (2019)
35. Roy, D., Ganguly, D., Mitra, M., Jones, G.J.: Estimating gaussian mixture models in the local neighbourhood of embedded word vectors for query performance prediction. Information processing & management **56**(3), 1026–1045 (2019)
36. Salamat, S., Arabzadeh, N., Seyedsalehi, S., Bigdeli, A., Zihayat, M., Bagheri, E.: Neural disentanglement of query difficulty and semantics. In: CIKM. pp. 4264–4268 (2023)
37. Tamannaee, M., Fani, H., Zarrinkalam, F., Samouh, J., Paydar, S., Bagheri, E.: Reque: a configurable workflow and dataset collection for query refinement. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management. pp. 3165–3172 (2020)
38. Yang, P., Fang, H., Lin, J.: Anserini: Enabling the use of lucene for information retrieval research. In: Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval. pp. 1253–1256 (2017)
39. Yom-Tov, E., Fine, S., Carmel, D., Darlow, A.: Learning to estimate query difficulty: including applications to missing content detection and distributed information retrieval. In: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 512–519 (2005)
40. Zhao, Y., Scholer, F., Tsegay, Y.: Effective pre-retrieval query performance prediction using similarity and variability evidence. In: Advances in Information Retrieval , 30th European Conference on IR Research, ECIR 2008, Glasgow, UK, March 30-April 3, 2008. Proceedings. pp. 52–64 (2008). https://doi.org/10.1007/978-3-540-78646-7_8, https://doi.org/10.1007/978-3-540-78646-7_8
41. Zhou, Y., Croft, W.B.: Query performance prediction in web search environments. In: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 543–550 (2007)