Reasoning About Query Difficulty to Predict Retrieval Effectiveness

Anonymous Author(s)

Abstract

Pre-retrieval query performance prediction methods estimate the effectiveness of a query prior to retrieval and enable control of the search pipeline through tasks such as query routing, selective expansion, and retriever selection. However, because they operate solely on query features without access to retrieved documents, pre-retrieval predictors often struggle to provide accurate predictions. They typically treat query difficulty as a latent statistical property rather than one that can be reasoned about. We propose a new pre-retrieval framework that leverages the reasoning capabilities of large language models (LLMs) to uncover and encode interpretable reasons of query difficulty. By integrating these reasons with transformer-based neural regressors, our approach allows the predictor to be both aware of and conditioned on reasons that make a query challenging. Our experiments on MS MARCO Dev, TREC DL 2019, TREC DL 2020, and DL-Hard demonstrate that our method consistently achieves reliable and stable correlations across various TREC DL datasets.

1 Introduction

Query Performance Prediction (QPP) estimates how effectively a query will be served by a given collection and retrieval configuration, either *ex ante* or using lightweight run-time signals [6, 13, 17]. Reliable performance estimates enable principled control of the search pipeline, informing decisions such as query reformulation, ranker or index selection, and query routing under latency and budget constraints [2, 10, 11, 24]. In retrieval-augmented generation settings, QPP further informs whether retrieval is beneficial at all [18, 25]. QPP methods are commonly divided into pre-retrieval and post-retrieval approaches [6, 15, 17]. Post-retrieval predictors exploit evidence obtained from an issued run, such as score distributions, agreement among rankers, or statistics over the top-ranked documents, and often achieve higher correlations with ground-truth effectiveness [1, 19, 28]. However, their reliance on retrieved documents limits their applicability in settings with strict latency or computational constraints [12, 26]. Pre-retrieval methods, in contrast, operate solely on the query and corpus-level statistics, producing predictions from signals such as term specificity, query clarity, or learned query representations [3, 16]. This makes pre-retrieval QPP particularly suitable for early-stage retrieval control, including query routing, selective query expansion, and adaptive ranker or index selection [27], provided that the prediction stage is substantially less expensive than the retrieval operation it conditions. While analogous control decisions may also be made using post-retrieval evidence, the pre-retrieval formulation is especially appealing when this computational asymmetry enables low-cost intervention before candidate generation. We therefore focus our study to the pre-retrieval QPP setting in this paper.

Recent pre-retrieval QPP research has increasingly moved away from hand-crafted features toward learned representations that attempt to infer query *difficulty* directly from the query text [4, 9, 11]. For example, QSD-QPP embeds queries into a learned query space and predicts effectiveness based on proximity to neighborhoods associated with historically easy or hard queries [5]. While such approaches substantially improve over classical corpus-statistic-based predictors, they treat difficulty as a latent quantity. As a result, they can indicate that a query is likely to perform poorly without providing insight into *why* it is difficult, limiting its interpretability.

In this paper, we argue that effective pre-retrieval QPP should go beyond producing a single difficulty score and instead make the sources of difficulty explicit. We view large language models (LLMs) as most useful in this setting not as direct difficulty scorers [22], but as mechanisms for eliciting and organizing interpretable reasons that explain why a query is likely to be challenging. Concretely, we hypothesize that an LLM can (i) induce a compact, stable inventory of difficulty reasons and (ii) attribute these reasons consistently to individual queries. A lightweight predictor can then condition on both the query text and its reason attributions, yielding improved predictive accuracy and better generalization across collections while preserving the efficiency required in pre-retrieval scenarios.

Based on this hypothesis, we propose Difficulty Reason Aware Query Performance (DRAQ), pronounced /dreɪk/, a two-stage framework that cleanly separates expensive reasoning from lightweight inference. In the *offline stage*, we systematically prompt an LLM to identify and categorize the underlying reasons why queries may be difficult to retrieve. Candidate reasons are aggregated across the training set, filtered to retain the most frequent explanations, and further consolidated by a more capable LLM into a compact inventory of 10 high-coverage difficulty reasons. These reason annotations, together with the query text, are then used to train a regression model for performance prediction. In the *online stage*, an incoming query is mapped to the same fixed set of difficulty reasons using the learned prompting strategy, and the trained regressor consumes both the query and its assigned reasons to produce a pre-retrieval effectiveness estimate.

We evaluate DRAQ on MS MARCO Dev [23], TREC DL 2019, 2020 [7, 8], and DL-Hard [21]. Our results show that DRAQ consistently achieves strong and stable correlations with ground-truth effectiveness. Beyond predictive accuracy, DRAQ enables dataset-level characterization of query difficulty: by examining the distribution of difficulty reasons across query sets, we reveal how different collections emphasize distinct types of challenging queries. Notably, the TREC DL datasets exhibit highly similar difficulty profiles, suggesting shared underlying retrieval challenges, while MS MARCO Dev displays a markedly different distribution. This provides a principled, interpretable lens for comparing query-set difficulty across datasets and for understanding why certain queries are systematically harder to satisfy in specific evaluation settings.

Our code, models, and difficulty reason annotations are made publicly available at https://anonymous.4open.science/r/draq-77E8.

Table 1: Query Difficulty Reasons Examples

Difficulty Reason	Short Name	Example Query (ID)
Too narrow or specific	Too Narrow	*what is orthorexia*
Too broad or vague	Too Broad	*what is bro?*
Lacks necessary context	No Context	*what are the three percenters?*
Ambiguous	Ambiguous	*define visceral?*
Requires specialized knowledge	Specialized	*what type of tissue are bronchioles*
Not a general knowledge query	Not Common	*who formed the commonwealth*
No clear answer available	Unclear Answer	*how much money do speakers make*
Not a standard question	Non-Standard	*ia suffix meaning*
Unconventional structure	Unconventional	*what is mca and get weekly paychecks fake?*

2 Proposed Approach

We make difficulty explicit by modeling *why* a query is likely to perform poorly and using these reasons as predictive signals. We define *difficulty reasons* as short, interpretable explanations of common failure modes, such as missing context, semantic ambiguity, or overly narrow scope. We use an LLM not as a direct difficulty scorer, but to (i) induce a compact, stable inventory of such reasons and (ii) assign them consistently to queries. A lightweight regressor then conditions on the query and its reason attributions to produce an effectiveness estimate.

Overview. DRAQ operates in two clearly separated stages: an *offline stage*, in which we construct and learn from an interpretable representation of query difficulty, and an *online stage*, in which this representation is used to produce pre-retrieval performance estimates for incoming queries. This separation allows all expensive reasoning and model training to be performed offline, while preserving the efficiency required in pre-retrieval scenarios.

Offline Stage: Difficulty Reason Discovery & Model Training.
Identifying Difficulty Reasons. The first objective of the offline stage is to uncover a small, interpretable set of reasons that explain *why* a query may be difficult for a retrieval system, without issuing any retrieval. Rather than treating difficulty as an opaque signal, we aim to represent it in terms of human-interpretable failure modes that recur across queries and collections.

Formally, given a set of training queries $Q = \{q_1, \ldots, q_n\}$, we seek to construct a compact inventory $\mathcal{R} = \{r_1, \ldots, r_k\}$ of difficulty reasons, where each r_j corresponds to a distinct and semantically coherent explanation of potential query failure. To balance expressiveness with interpretability, we explicitly constrain the size of this inventory to a small range ($10 \le k \le 20$), ensuring that the resulting reasons remain high-level and non-redundant.

To populate this inventory, we prompt a language model $\mathcal{L}$ to generate candidate difficulty reasons independently for each query q_i, yielding a set $\mathcal{R}^{(i)} = \mathcal{L}(q_i)$. Aggregating these candidates across all training queries produces a large pool of explanations, many of which are lexically different but semantically overlapping. To consolidate this space, we first apply lightweight surface normalization (e.g., punctuation removal) to collapse trivial string variants, count the frequency of the resulting candidate reasons, and retain the top 100 most frequently occurring ones, which capture the dominant sources of difficulty in the training data. At this stage frequency reflects only surface-level matching; semantically equivalent but lexically distinct reasons are merged in the next step. We then prompt a more capable language model (GPT-4o) to cluster, merge, and canonicalize these frequent reasons into a smaller set of non-redundant, high-coverage categories. This aggregation model (GPT-4o) is used only in the offline discovery step and is distinct from the candidate-generation and assignment model reported in Table 2, which is either `Qwen3:8b` or `GPT-4o-mini`. This process results in a final inventory of 10 difficulty reasons that jointly explain the majority of observed query failures while remaining concise and interpretable.

Difficulty Reason Assignment. Once the inventory of difficulty reasons has been fixed, the next offline objective is to determine how these reasons apply to individual training queries. Since query difficulty often arises from multiple interacting factors, we model this step as a multi-label attribution problem.

Given a query q_i and the finalized inventory $\mathcal{R} = \{r_1, \ldots, r_k\}$, we assess which reasons plausibly contribute to the difficulty of q_i. This assessment is performed with a single language-model call per query that jointly returns a multi-label decision over all k reasons (rather than one call per reason). The result is a binary attribution vector $\mathbf{a}_i = [a_{i1}, \ldots, a_{ik}] \in \{0,1\}^k$, where $a_{ij} = 1$ indicates that reason r_j is assigned to query q_i. Allowing multiple active reasons reflects the empirical observation that difficult queries frequently exhibit overlapping failure modes, such as ambiguity combined with missing context.

Model Input Construction and Training. The final component of the offline stage is to train a pre-retrieval performance predictor that directly incorporates difficulty-reason information. Rather than using difficulty reasons solely for interpretation, we integrate them into the model input so that they can actively guide prediction.

For each training query q_i, we construct an augmented textual representation by concatenating the query text with its binary reason attribution vector $\mathbf{a}_i$:

$$\tilde{q}_i = q_i \; a_{i1} \; a_{i2} \; \ldots \; a_{ik}.$$

This augmented sequence allows the model to jointly reason over the semantic content of the query and explicit indicators of potential difficulty. The sequence $\tilde{q}_i$ is processed by an encoder $\mathcal{E}$, `bert-base-uncased` , which takes the input $[CLS]$ $\tilde{q}_i$ $[SEP]$ and produces contextualized representations. We use the final-layer hidden state of the `[CLS]` token, denoted $h^L_{[CLS]} \in \mathbb{R}^d$, as a fixed-dimensional representation of the query and its associated difficulty reasons. After dropout, a single linear regression head followed by a sigmoid activation maps this representation to a predicted effectiveness score in $[0,1]$:

$$\widehat{M}_{q_i} = \sigma\big(W\, h^L_{[CLS]} + b\big),$$

where $W \in \mathbb{R}^{1\times d}$, $b \in \mathbb{R}$, and σ denotes the logistic sigmoid. The model is trained end-to-end by minimizing the mean squared error between the predicted score $\widehat{M}_{q_i}$ and the ground-truth normalized effectiveness score M_{q_i}, jointly fine-tuning the encoder $\mathcal{E}$ and the regression head.

Online Stage: Pre-Retrieval Performance Prediction. At inference time, the objective is to produce a pre-retrieval estimate of query effectiveness using only the query text and the artifacts

Algorithm 1 Difficulty Reason Discovery

Require: Training queries $Q = \{q_1, \ldots, q_n\}$
Require: Large Language Models $\mathcal{L}_1$ (candidate generation), $\mathcal{L}_2$ (aggregation)
Ensure: Compact difficulty reason set $R = \{r_1, \ldots, r_k\}$
for each query $q_i \in Q$ **do**
  $R_i \leftarrow \mathcal{L}_1(q_i)$
end for
$R_{\text{all}} \leftarrow \bigcup_i R_i$
Select top-N frequent reasons from R_{all}
$R \leftarrow \mathcal{L}_2(R_{\text{top}})$
Require: Queries $Q = \{q_1, \ldots, q_n\}$, difficulty reasons $R = \{r_1, \ldots, r_k\}$
Require: Language Model $\mathcal{L}_1$
Ensure: Reason attribution vectors $\mathbf{a}_i \in \{0, 1\}^k$
for each query $q_i \in Q$ **do**
  $\mathbf{a}_i \leftarrow \mathcal{L}_1(q_i, R)$ {single multi-label call returning all k labels}
end for

learned during the offline stage. For an incoming query, we first assign difficulty reasons using the fixed inventory $\mathcal{R}$ and the same prompting strategy used during training. We then form the augmented query representation, encode it with the trained transformer encoder, and apply the regression head to obtain the final effectiveness prediction.

3 Experiments and Results

3.1 Experimental Setup

Datasets. We conduct experiments on the MS MARCO Passage Ranking dataset (V1) [23], with over 500,000 training queries accompanied by sparse relevance annotations. For evaluation, we adopt four widely used benchmark query sets: (1) the MS MARCO Dev set containing 6,980 queries, (2) TREC DL 2019 [8] with 43 queries, (3) TREC DL 2020 [7] with 54 queries, and (4) DL-Hard [21] comprising 50 particularly challenging queries. All three query collections of DL 2019, DL 2020, and DL-Hard are comprehensively labeled with graded relevance judgments on a 0–3 scale, whereas the MS MARCO Dev set is sparsely labeled.

Evaluation Metrics. We assess the model's ability to predict the performance of the BM25 retriever on the MS MARCO collection. For each dataset, we predict the official evaluation metrics: MRR@10 for MS MARCO Dev and nDCG@10 for TREC DL 2019, TREC DL 2020, and DL-Hard. We then evaluate the quality of the predicted scores against the ground truth using both linear and rank-based correlation measures, specifically, Pearson's correlation coefficient for linear relationships and Kendall's τ and Spearman's ρ for rank-based associations.

Baselines. We compare our proposed framework against a comprehensive set of existing QPP methods spanning both traditional statistical approaches and recent neural models. The statistical-based baselines include widely adopted predictors such as `SCQ` [30], `VAR` [30], `PMI` [14], `IDF` and `ICTF` [20], which rely on corpus-level term statistics and lexical matching heuristics. In addition, we incorporate neural embedding-based models such as `CC`, `DC` and `IEF` [3], and contextualized transformer-based predictors including `MRL` [29] and `QSD-QPP` [5], representing state-of-the-art neural and pre-trained transformer QPP frameworks. To further examine whether LLMs alone can perform pre-retrieval performance prediction, we also implement a few-shot QPP baseline using in-context learning. Specifically, we embed all training queries using `all-MiniLM-L6-v2` and construct an index using `FAISS`. For each test query, we retrieve the five most similar training queries and use them as in-context examples in the prompt, asking the LLM to directly predict the query's effectiveness score. Due to space constraints, we refer readers to the respective original publications for full implementation details of the baselines. The prompts and implementation details for our few-shot approach are provided in our public repository. We note that our reported `QSD-QPP` correlations differ from those in the original paper because we retrain it under our passage-level MS MARCO setup and predict BM25's MRR@10/nDCG@10 rather than the original document-retrieval target. We state this explicitly to ensure that the comparison is understood as being made under correct conditions.

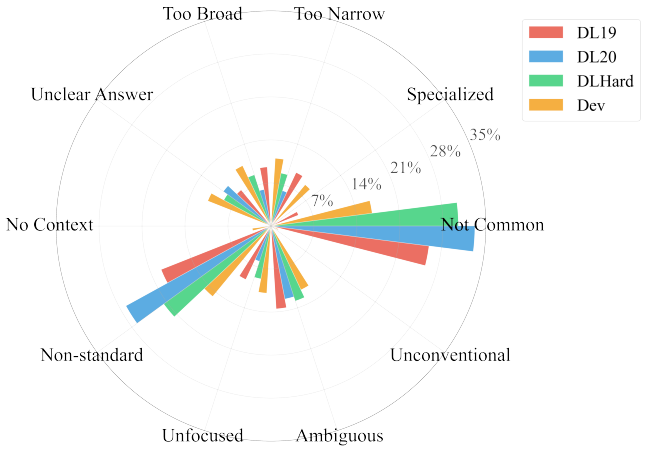

Figure 1: Distribution of difficulty reasons.

3.2 Findings

Table 2 presents the performance of DRAQ alongside all baselines across the four benchmark query sets.

Across nearly all settings, DRAQ outperforms both statistical and neural transformer-based baselines, achieving the highest correlations on every dataset except Kendall's τ on TREC DL 2020 and Pearson on DL-Hard, where it remains competitive.

Among our two instantiations, DRAQ with the open-source Qwen3:8b produces stronger and more consistent attributions, suggesting that closed-source frontier models are not necessary for effective difficulty-reason extraction.

We also observe that the FewShot baseline performs inconsistently and often poorly across datasets. This indicates that *standalone prompting without task-specific training is insufficient for reliable pre-retrieval performance prediction.* In contrast, DRAQ benefits from structured reason induction combined with supervised regression training, which leads to significantly improved predictive alignment. Against neural baselines (MRL [29], QSD-QPP [5], and CC/DC/IEF [3]), DRAQ shows substantial improvements across all benchmarks. QSD-QPP is competitive only on DL-Hard and, unlike

Table 2: Correlation between predicted and actual performance. All reported correlations of our approach are statistically significant at $p < 0.01$.

Method	MS MARCO Dev			TREC DL 2019			TREC DL 2020			TREC DL Hard		
	P-ρ	*K-τ*	*S-ρ*	*P-ρ*	*K-τ*	*S-ρ*	*P-ρ*	*K-τ*	*S-ρ*	*P-ρ*	*K-τ*	*S-ρ*
SCQ	0.012	0.011	0.014	0.298	0.116	0.162	0.226	0.076	0.132	0.264	0.127	0.179
VAR	0.074	0.062	0.083	0.197	0.107	0.152	0.107	0.059	0.077	0.053	0.016	0.035
PMI	0.019	0.017	0.023	0.090	0.009	0.017	0.017	0.040	0.056	0.014	0.022	0.031
IDF	0.125	0.116	0.154	0.311	0.158	0.245	0.395	0.245	0.353	0.075	0.111	0.125
ICTF	0.112	0.114	0.152	0.309	0.153	0.240	0.287	**0.345**	0.330	0.073	0.107	0.115
CC	0.080	0.065	0.085	0.103	0.099	0.055	0.034	0.106	0.026	0.098	0.103	0.141
DC	0.102	0.107	0.144	0.065	0.095	0.053	0.050	0.091	0.035	0.112	0.123	0.165
IEF	0.085	0.094	0.104	0.156	0.177	0.166	0.056	0.064	0.081	0.189	0.140	0.191
MRL	0.018	0.024	0.032	0.149	0.079	0.14	0.093	0.078	0.117	-0.013	0.049	0.057
QSD-QPP	0.018	0.172	0.227	0.096	0.100	0.144	0.323	0.203	0.298	**0.401**	0.241	0.341
FewShot	0.0419	0.022	0.025	0.039	-0.035	-0.038	0.21	0.100	0.125	0.171	0.087	0.104
DRAQ (Qwen3:8b)	**0.401**	**0.328**	**0.434**	**0.317**	**0.181**	**0.270**	**0.466**	0.324	**0.450**	0.377	**0.291**	**0.402**
DRAQ (GPT-4o-mini)	**0.400**	**0.328**	**0.433**	0.273	**0.19**	0.223	0.343	0.262	**0.39**	0.391	**0.271**	**0.389**

DRAQ, requires constructing a learned query space before inference. Among traditional statistical predictors, ICTF [20] emerges as the strongest competitor, particularly on TREC DL 2020. Nevertheless, DRAQ consistently surpasses ICTF across datasets, demonstrating that explicit difficulty reasoning provides stronger generalization than corpus-level term statistics alone.

Overall, DRAQ maintains stable and high correlations across all datasets and metrics, with no Pearson correlation falling below 0.3. This stability underscores the benefit of incorporating interpretable difficulty-reason information, enabling improved generalization to unseen query sets while preserving strong predictive alignment with actual retrieval effectiveness.

Inference Latency. Despite invoking an LLM, DRAQ is as cheap as the simplest LLM predictor, i.e., per-query latency is 0.57 ± 0.24s versus 0.56 ± 0.21s for FewShot, an overhead of only ∼0.01s (one `bert-base-uncased` pass), while DRAQ is far more accurate (Table 2). Cost is fixed at a single LLM call per query, independent of inventory size k and corpus size.

3.3 Validation of Difficulty Reasons

A central concern is whether the LLM-assigned difficulty reasons are faithful to query difficulty or merely arbitrary model artifacts. To assess this directly, we conduct a human annotation study. Three researchers (with expertise in IR/NLP) independently reviewed the assignments for 50 queries, judging each of the 10 reasons per query, which yields $3 \times 50 \times 10 = 1{,}500$ (query, reason) judgments. We take the majority vote across annotators as the human reference and compare it against the binary reason vectors produced by DRAQ. Per-decision agreement is high (label accuracy 0.891), and on the positive (reason-present) class the assignment reaches a micro-F1 of 0.644 (precision 0.667, recall 0.622). This shows a strong relation between human and LLM judgment, which supports our hypothesis about using LLM as a judge for this purpose.

3.4 Distribution of Query Difficulty Reasons

Figure 1 illustrates the distribution of the identified difficulty reasons across the four evaluation datasets. The goal of this analysis is to examine whether certain difficulty reasons occur more frequently than others and to identify difficulty reasons that characterize different query collections. We observe that some difficulty reasons are noticeably more prevalent than others. In particular, the categories *"non-standard question"*, which often corresponds to long-tail or idiosyncratic information needs, and *"not general knowledge"* appear most frequently across all datasets. By contrast, reasons such as *"requiring specialized knowledge"* or *"unconventional structure"* are relatively rare. This is expected, as the four datasets are composed of real user queries, which tend to reflect everyday information needs rather than highly technical or domain-specific questions [23]. Interestingly, we find that DL19, DL20, and DL-Hard exhibit similar overall patterns in their distribution of reasons, yet the exact proportions differ substantially. This suggests that, although these datasets share a common source (the TREC Deep Learning track) and learning-based curation process, their selection criteria introduce subtle biases that influence which types of difficult queries are represented in each of them. For instance, certain reasons, such as *"query too broad"* and *"query too narrow"*, display comparable frequencies, implying that overly specific and overly general queries have comparable predicted frequencies. We find that while the spectrum of difficulty reasons is consistent across datasets, their relative frequencies vary depending on how the queries were collected and selected, possibly pointing to dataset-specific biases in what constitutes a 'hard' query; however, the strong and consistent predictive performance of DRAQ indicates that difficulty reasons are both transferable and generalizable across datasets. Moreover, beyond performance prediction, DRAQ offers a new lens for characterizing the overall difficulty profile of a query set, enabling a systematic analysis of dataset-level challenges and possible biases.

4 Concluding Remarks

This paper presented a new approach for pre-retrieval performance estimation by explicitly capturing and integrating reasons of query difficulty. Instead of treating difficulty as a latent property, our work leverages LLMs to elicit interpretable potential difficulty reasons of queries and integrates them into a transformer-based regressor. Experiments on MS MARCO Dev, TREC DL 2019, TREC DL 2020, and DL-Hard show that our approach consistently outperforms strong statistical and neural baselines, achieving stable correlations across datasets, while a human study confirms that the induced reasons align with expert judgments rather than being model artifacts. Future work will extend reason-aware modeling to post-retrieval and RAG settings for adaptive retrieval.

References

[1] Arabzadeh, N., Khodabakhsh, M., Bagheri, E.: Bert-qpp: Contextualized pre-trained transformers for query performance prediction. In: CIKM (2021)

[2] Arabzadeh, N., Meng, C., Aliannejadi, M., Bagheri, E.: Query performance prediction: From fundamentals to advanced techniques. In: European Conference on Information Retrieval. pp. 381–388. Springer (2024)

[3] 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). https://doi.org/10.1016/j.ipm.2020.102248

[4] 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)

[5] Bigdeli, A., Ebrahimi, S., Arabzadeh, N., Salamat, S., SeyedSalehi, S., Khodabakhsh, M., Zarrinkalam, F., Bagheri, E.: Query performance prediction using neural query space proximity. ACM Trans. Intell. Syst. Technol. (Sep 2025). https://doi.org/10.1145/3762197, https://doi.org/10.1145/3762197, just Accepted

[6] Carmel, D., Yom-Tov, E.: Estimating the Query Difficulty for Information Retrieval, Synthesis Lectures on Information Concepts, Retrieval, and Services, vol. 2. Morgan & Claypool Publishers (2010). https://doi.org/10.2200/S00237ED1V01Y201006ICR016

[7] 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

[8] 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)

[9] Cronen-Townsend, S., Zhou, Y., Croft, W.B.: Predicting query performance. In: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 299–306 (2002)

[10] Faggioli, G., Ferro, N., Mothe, J., Raiber, F., Fröbe, M.: Report on the 1st workshop on query performance prediction and its evaluation in new tasks (qpp++ 2023) at ecir 2023. SIGIR Forum **57**(1) (Dec 2023). https://doi.org/10.1145/3636341.3636356, https://doi.org/10.1145/3636341.3636356

[11] Faggioli, G., Ferro, N., Muntean, C.I., Perego, R., Tonellotto, N.: A geometric framework for query performance prediction in conversational search (2023)

[12] Faggioli, G., Formal, T., Lupart, S., Marchesin, S., Clinchant, S., Ferro, N., Piwowarski, B.: Towards query performance prediction for neural information retrieval: Challenges and opportunities. In: Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval. p. 51–63. ICTIR '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3578337.3605142, https://doi.org/10.1145/3578337.3605142

[13] Faggioli, G., Formal, T., Marchesin, S., Clinchant, S., Ferro, N., Piwowarski, B.: Query performance prediction for neural ir: Are we there yet? In: European Conference on Information Retrieval. pp. 232–248. Springer (2023)

[14] Hauff, C.: Predicting the effectiveness of queries and retrieval systems. In: SIGIR Forum. vol. 44, p. 88 (2010)

[15] Hauff, C., Azzopardi, L., Hiemstra, D., de Jong, F.: Query performance prediction: Evaluation contrasted with effectiveness. In: European Conference on Information Retrieval. pp. 204–216. Springer (2010)

[16] 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. p. 1419–1420. CIKM '08, Association for Computing Machinery, New York, NY, USA (2008). https://doi.org/10.1145/1458082.1458311

[17] He, B., Ounis, I.: Query performance prediction. Information Systems **31**(7), 585–594 (2006)

[18] Huly, O., Carmel, D., Kurland, O.: Predicting rag performance for text completion. In: Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 1283–1293 (2025)

[19] Kurland, O., Shtok, A., Carmel, D., Hummel, S.: A unified framework for post-retrieval query-performance prediction. In: Conference on the Theory of Information Retrieval. pp. 15–26. Springer (2011)

[20] 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

[21] 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)

[22] Meng, C., Arabzadeh, N., Askari, A., Aliannejadi, M., Rijke, M.d.: Query performance prediction using relevance judgments generated by large language models. ACM Transactions on Information Systems **43**(4), 1–35 (2025)

[23] 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)

[24] Poesina, E., Ionescu, R.T., Mothe, J.: iqpp: A benchmark for image query performance prediction. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 2953–2963. SIGIR '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3539618.3591901, https://doi.org/10.1145/3539618.3591901

[25] Pogrebinsky, I., Carmel, D., Kurland, O.: Enhancing retrieval-augmented generation for text completion through query selection. In: Proceedings of the 2025 International ACM SIGIR Conference on Innovative Concepts and Theories in Information Retrieval (ICTIR). pp. 410–415 (2025)

[26] 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)

[27] Roitman, H., Erera, S., Feigenblat, G.: A study of query performance prediction for answer quality determination. In: Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval. pp. 43–46 (2019)

[28] Shtok, A., Kurland, O., Carmel, D.: Predicting query performance by query-drift estimation. In: Conference on the theory of information retrieval. pp. 305–312. Springer (2009)

[29] Zamani, H., Bendersky, M.: Multivariate representation learning for information retrieval. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 163–173. SIGIR '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3539618.3591740, https://doi.org/10.1145/3539618.3591740

[30] 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