When More Reformulations Hurt: Avoiding Drift using Ranker Feedback

Venktesh V
DSV, Stockholm University
Stockholm, Sweden
venktesh.viswanathan@dsv.su.se

Mandeep Rathee
L3S Research Center
Hannover, Germany
rathee@l3s.de

Avishek Anand
Delft University of Technology
(EEMCS/EWI, TU Delft)
Delft, The Netherlands
avishek.anand@tudelft.nl

Abstract

Modern retrieval pipelines increasingly rely on query reformulation and neural reranking to improve effectiveness, but this comes at a significant computational cost and introduces a fundamental tradeoff between *recall* and *query drift*. Generating many reformulated queries can substantially increase recall, yet naïvely merging or exhaustively reranking their results is prohibitively expensive. In this work, we argue that the core challenge is not reformulation generation itself, but the adaptive selection of reformulations and their retrieved documents under a strict inference budget.

We propose ReformIR, a budget-aware retrieval framework that treats query reformulations as first-class features and performs online relevance estimation using a strong reranker as a teacher. Given multiple reformulated queries, ReformIR constructs a large candidate pool and learns a lightweight surrogate model that estimates document utility from reformulation-specific retrieval signals. Under a fixed reranking budget, the surrogate adaptively prioritizes both reformulations and documents, selectively querying a teacher reranker anchored to the original query. This process increases recall while actively suppressing drift through online feature selection over reformulations. We conduct extensive experiments on the MSMARCO passage corpora and TREC Deep Learning benchmarks (DL19–DL22), evaluating effectiveness under realistic reranking budgets. Our results show that ReformIR consistently outperforms existing reformulation strategies, particularly as the number of reformulations increases, where prior methods suffer from severe quality degradation due to drift. Our findings also suggest a shift in retrieval system design: rather than using large language models as rerankers, their capacity is more effectively leveraged in the reformulation stage with feedback-driven optimization. Our code:

https://github.com/VenkteshV/ReformIR

CCS Concepts

• **Information systems → Retrieval models and ranking**; **Query reformulation**.

Keywords

Query Reformulations, Reranking, Document Selection

SIGIR '26, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-2599-9/2026/07
https://doi.org/10.1145/3805712.3809721

ACM Reference Format:
Venktesh V, Mandeep Rathee, and Avishek Anand. 2026. When More Reformulations Hurt: Avoiding Drift using Ranker Feedback. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '26), July 20–24, 2026, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3805712.3809721

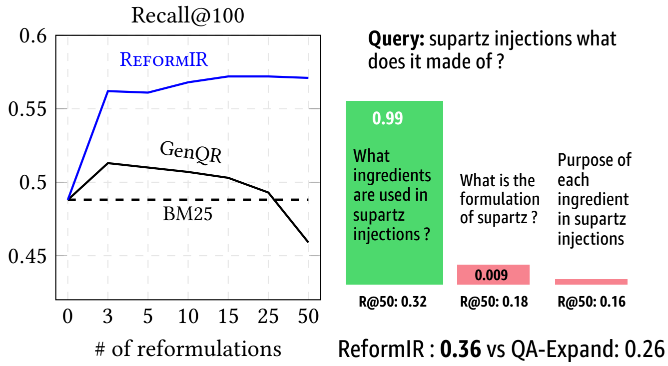
1 Introduction

Query expansion is one of the fundamental tasks in document ranking and has been studied for a long time in the field [3, 35]. Long-standing work has shown that reformulating a user query, by adding, removing, or reweighting terms, is often essential to overcome vocabulary mismatch between queries and documents using user or pseudo-relevance feedback [3, 11, 34]. At the same time, this line of work also identified a fundamental risk: indiscriminate expansion can introduce query drift, where added terms shift the query away from the user's original intent and degrade performance at lower depths [35].

Instead of relying solely on corpus statistics, modern approaches use LLMs to generate reformulated queries, expansion terms, or even pseudo-documents based on their parametric memory and their improved language understanding. Methods such as Query2Doc [42] generate synthetic documents from the query, while GenQR and its variants [7, 43] directly prompt LLMs to rewrite or expand queries. More recent work, such as QA-Expand [36], introduces diversity through question–answer generation, using LLMs to synthesize multiple sub-questions and answers as expansion signals.

Despite their empirical success, existing LLM-based reformulation methods largely share a common limitation. They are potentially effective at the *recall-increasing step* of retrieval, retrieving documents that would be missed by a single query, but remain weak at the *drift-reduction step*. Generated reformulations are typically combined using heuristic rules, fixed interpolation weights, or post-hoc filtering, without being explicitly guided by retrieval feedback. As a result, irrelevant or weak reformulations often contribute noise, forcing systems to conservatively down-weight expansion or accept increased drift. We argue that in this sense, many modern approaches *inherit* the classical query drift problem, but without the explicit feedback mechanisms that earlier relevance models used to control it.

In this work, we introduce ReformIR, a retrieval framework that uses LLMs exclusively as *low-cost reformulation generators* and employs a budgeted reranker as a feedback mechanism to re-weight reformulations. ReformIR generates multiple reformulations, retrieves documents independently for each reformulation to maximize recall, and then learns, under a strict budget, which reformulations are useful. Inspired from existing works on online



(a) (b)

Figure 1: Drift control and learned reformulation weights. (a) Recall@100 on TREC DL19 as the number of reformulations increases. While GenQR suffers from performance degradation as additional reformulations introduce drift, ReformIR remains stable and consistently improves recall by selectively prioritizing effective reformulations using ranker feedback. (b) Reformulation weights learned by ReformIR through online optimization for a representative query (We also show Recall@50 for individual queries, QA-Expand baseline and ReformIR).

feature attributions in explainable IR [18, 38], reformulations are treated as features in a surrogate relevance model. We then employ a bandit-style optimization loop that allocates a limited number of reranker calls to progressively focus on documents retrieved by promising reformulations. Importantly, the reranker always evaluates relevance with respect to the original query, anchoring optimization, and actively suppressing query drift.

This design yields several advantages. First, it decouples query understanding from expensive document-level reasoning. LLMs are used where they are strongest and cheapest: for understanding and rephrasing queries, rather than where they are most costly. Second, because reformulation operates in a short-context regime, ReformIR naturally supports the use of smaller language models, enabling further reductions in inference cost. Third, the framework is generator-agnostic and can be layered like an adapter on top of any reformulation method; our experiments show consistent improvements across a wide range of generators. Finally, by learning explicit weights over reformulations, ReformIR provides interpretability, revealing the aspects of a query driving relevance.

Figure 1a provides a concrete illustration of the query drift problem and how ReformIR addresses it. The figure plots Recall@100 as a function of the number of reformulations for a classical LLM-based reformulation approach (GenQR) and its counterpart augmented with ReformIR. As the number of reformulations increases, the classical approach exhibits a clear degradation in performance: while additional reformulations initially improve recall, further expansions introduce irrelevant reformulations that dominate the retrieval pool, leading to drift and a steady decline in effectiveness. In contrast, ReformIR remains robust as the number of reformulations grows. By using ranker feedback to learn which reformulations are actually useful and suppressing those that induce drift, ReformIR maintains stable and consistently higher recall even under aggressive reformulation scaling. This highlights a key distinction between naive expansion and adaptive optimization: simply adding more reformulations amplifies noise, whereas ReformIR actively controls drift by anchoring relevance feedback to the original query.

Our extensive experiments on TREC Deep Learning benchmarks demonstrate that ReformIR functions as a training-free adapter that consistently reduces query drift across reformulation methods, yielding gains of up to **33.48%**. While classical reformulation approaches degrade as the number of reformulations increases, ReformIR remains stable and continues to improve performance even. These benefits hold across a diverse set of reformulation generators and model scales, while incurring only marginal additional latency compared to classical approaches and remaining **3.3×–4.5×** more efficient than LLM-based reranking.

This paper makes the following key contributions:

- We introduce ReformIR, a novel retrieval framework based on bandit-style optimization that actively helps suppress query drift using reranker feedback.
- We provide extensive empirical evidence showing that ReformIR consistently improves retrieval effectiveness across different reformulation baselines serving as a training-free adapter.
- We demonstrate how LLMs should be pragmatically integrated into retrieval systems, combined with lightweight, feedback-driven optimization, resulting in superior effectiveness, lower latency, and interpretability compared to downstream LLM-based reranking.

2 Related Work

The efficient first stage retrievers, like BM25, only focus on lexical matching and perform poor when there is a vocabulary mismatch, especially for web queries which are usually short and ambiguous. Towards this end, several query expansion and reformulation techniques have been proposed to mitigate this issue.

2.1 Traditional Query Reformulation

Traditional approaches to query reformulation primarily rely on test-time expansion strategies to enhance lexical overlap. Pseudo-Relevance Feedback (PRF) methods, such as RM3 [11], assume that top-ranked documents in an initial retrieval pass contain relevant terms; these terms are then extracted to augment the original query. Similarly, the Rocchio algorithm [34] provides a formal framework for query vector modification by incorporating weighted feedback from both relevant and non-relevant documents.

Further refinements introduced probabilistic and information-theoretic perspectives. KL Expansion [46] utilizes language modeling to optimize interpolation coefficients between the query and expansion terms, while Relevance Modeling (RM) [16] estimates a query-specific language model from top-retrieved passages. Beyond simple term frequencies, Latent Concept Expansion (LCE) [26] uses Markov Random Fields to identify higher-level concepts, and Local Context Analysis (LCA) [45] examines co-occurrence patterns within the local retrieval context. These methods collectively establish the foundation for modern test-time query reformulation.

Though these methods use initially retrieved documents based on lexical overlap, this overlap is still bounded by the original query. Moreover, these methods employ heuristic strategies for combining reformulations and fail to identify and mitigate semantic drift.

2.2 Generative Query Reformulation

More recently, powerful generative language models have been found to generate well-formed variants of the original query [1, 10, 39, 43]. Several methods have been proposed to employ off-the-shelf LLMs, which can be broadly divided into the following classes: rephrasing-based [7, 43], pseudo-relevance feedback based [7], pseudo-document generation [42], and question answer generation [36] based approaches. Since the web style queries are usually shorter and face the vocabulary mismatch problem, LLM reformulated queries can help find further relevant documents.

GenQR [43] employs generative LLMs to rephrase the original query to generate reformulations, but employs heuristic filtering mechanisms to prevent topic drift with respect to the original query. GenQREnsemble [7] argues that reformulations based on a single instruction may lead to worse performance and proposes ensembling of reformulations from multiple instructions to the LLM in a zero-shot setting. Generative Relevance Feedback (GRF) [22–24] generate several reformulations, for example, by generating keywords, entities, documents, an essay, or a news article. Query2Doc [42] employs zero-shot / few-shot by providing query-document pairs and generates a document for a given query. Finally, the generated document is concatenated with the original query. Methods like QueryExpansion [10] expand the query in a few-shot or CoT (Chain-of-Thoughts) setup [44]. HyDE [9], which generates hypothetical relevant documents (can be factually incorrect), using parametric knowledge of LLMs, for a given query and uses their embeddings as enhanced query representations for dense retrieval. However, these hypothetical documents might lead to concept drift in the query, and the retrieval system ends up with noisy documents.

While effective, these "zero-shot" generative methods can introduce concept drift if the LLM generates factually incorrect text. Recent hybrid approaches like Mill [12], BlendFilter [41], MuGI [47], and ReAL [4], and pseudo-relevance feedback-based methods like GenQREnsemblePRF [7] integrate initial retrieved documents into the generation process to try and mitigate this issue. However, [42] observes that when the quality of the original query and its retrieved results are poor, it causes further drift compared to zero-shot approaches. Furthermore, models like GAR [25] and LameR [37] iteratively refine the query based on the LLM's own generated answers, ensuring better alignment with the final task.

For purposes of retrieval, the existing works combine the initial query with reformulated queries in several ways. Primarily, most works adapt a heuristic approach by concatenating the initial query or repetitions of the initial query with the reformulated queries to perform retrieval [7, 36, 42, 43] . However, this approach also incorporates reformulations with drift, which causes the retrieval of irrelevant documents, impacting performance. Alternatively, individual reformulated queries are used to retrieve documents, which are then combined using approaches like RRF [5] in approaches like [8, 17, 36]. However, RRF considers all queries to be equal. Although it is intended to capture multiple facets of queries, it often combines redundant or low-quality candidates as observed in [36], causing deterioration in performance. Our work focuses on a principled approach to adaptively prioritize high-quality queries and documents using a feedback mechanism from the reranker.

While prior works like ODIS [20], explore distilling re-ranker to a smaller lexical scoring function, they require scoring of all documents from first-stage retrieval for a query to enable the distillation. This is followed by the selection of documents from the whole corpus using the distilled model with another round of re-ranking by the cross-encoder model. However, our approach focuses on adaptively prioritizing promising reformulations and also prioritizes corresponding documents. Our approach also relies only on linear combination of simple features for the linear surrogate and obviates need for re-ranking all retrieved documents corresponding to reformulations.

While other line of works, such as DeepRetrieval [13], where LLM is fine-tuned using RL for the query reformulation task exists, we focus only on **training-free** query reformulation methods.

3 Our Approach ReformIR

In this section, we present our approach in detail, addressing the challenges of using multiple query reformulations for retrieval. Our aim is to achieve the high recall afforded by multiple reformulations with a low budget on reranker evaluations by avoiding query drift. An overview of the pipeline is shown in Figure 2.

3.1 Preliminaries and Problem Setup

Let Q be the original user query expressing an information need, and let $\mathcal{D}$ be a large document collection. Modern retrieval systems often generate multiple reformulations of Q (alternative phrasings, expansions, etc.) to improve recall, especially for complex or ambiguous queries. The benefit is that different reformulations can retrieve different relevant documents. The drawback is a well-known issue of *query drift*: some reformulations might stray from the user's intent, retrieving off-target documents [40].

Hence, apart from generating a wide range of reformulated queries; the key challenge is deciding which reformulations (and their results) to prioritize and which to down-weight or ignore. We tackle this by treating reformulations as latent features of the retrieval process and using a strong reranker, anchored to the original query, as a guide (or "teacher") to dynamically select and weight reformulations under a strict budget of reranker evaluations.

3.2 Overview of the ReformIR Pipeline

Our proposed pipeline, called ReformIR, is designed to get the best of both worlds: it leverages multiple query reformulations and employs adaptive selection of reformulation and documents from candidate pool utilizing reranker feedback carefully to minimize drift and achieve higher recall. The pipeline consists of four main stages:

3.2.1 Reformulations Generation. Produce a set of m reformulated queries from the original query Q. These reformulations $Q_1, Q_2, \ldots, Q_m$ aim to cover various interpretations or aspects of the query, increasing the chance of retrieving all relevant information.

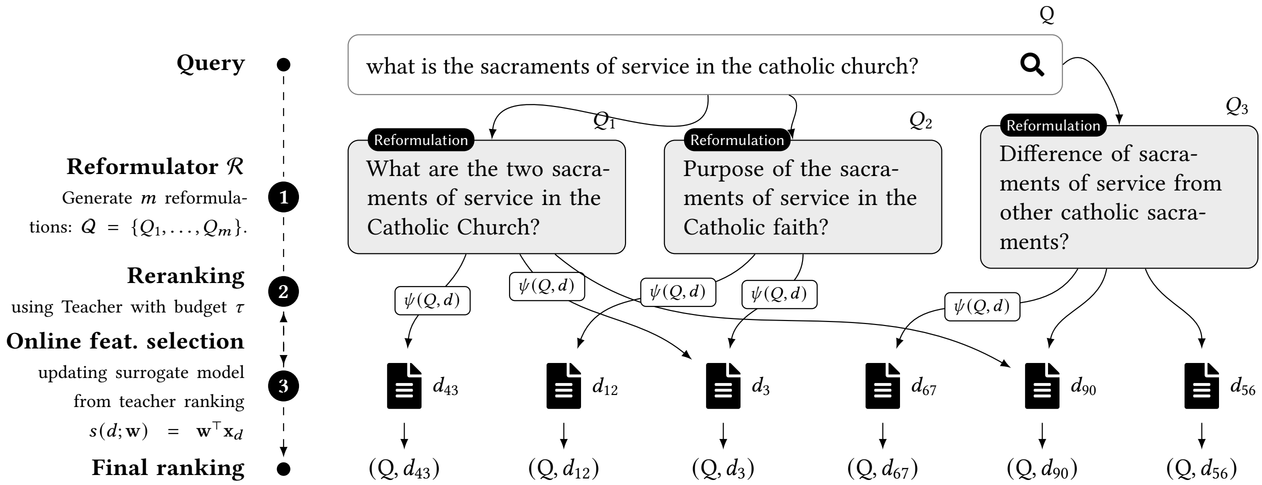


Figure 2: A broad overview of proposed approach ReformIR. The original query is reformulated to several queries as shown. This is followed by BM25 retrieval to form candidate pool as detailed in Section 3. Then Steps 2-3 adaptively prioritize documents using relevance estimated by surrogate for sampling ranker feedback and performs updates to the surrogate are iterative as detailed in Algorithm 1. Final ranking shows that reformulated queries Q_1 and Q_2 are more closer semantically to the original query and hence they are upweighted and their documents are retrieved and ranked higher. Q_3 focuses on "difference of sacraments" and drifts from original meaning of the query which results in downweighting it and corresponding documents.

3.2.2 Candidate Retrieval. Use a first-stage retriever (we use BM25) to execute each reformulated query Q_i and obtain a list of top-k documents $D_i = [d_{i1}, d_{i2}, \ldots, d_{ik}]$. The initial results for the original query Q (denoted R_0) are also obtained. All retrieved documents are combined into a candidate pool:

$$C = R_0 \cup \bigcup_{i=1}^{m} D_i \,,$$

where C may contain duplicate documents if the same document appears in multiple D_i (in practice, such duplicates are de-duplicated in the set union). This union operation ensures maximal recall, any document that is highly ranked for any reformulation or for the original query will be considered. However, C can be quite large and potentially noisy, since some reformulations might pull in irrelevant content. While we typically cap the depth k of each list (typically 100) to keep $|C|$ manageable, while still allowing C to be large enough to include most relevant documents it could still comprise of irrelevant documents from low quality reformulations. Figure 2 illustrates this multi-reformulation retrieval process.

3.2.3 Surrogate Model (Reformulations as Features). We embed reformulations into a feature-based surrogate model that can predict document relevance. Instead of deciding upfront which reformulations are useful, we assign each reformulation a weight that will be learned from data. Concretely, for each document $d \in C$, we construct a feature vector $\vec{x}_d \in \mathcal{R}^m$ with one dimension for each reformulated query's contribution to d.

$$\vec{x}_d = \left[\text{BM25}(Q_1, d), .., \text{BM25}(Q_m, d), \text{BM25}(Q, d), \text{RM3}(Q', d)\right] \tag{1}$$

This way, each reformulation Q_i is represented as a feature indicating how well document d matches that particular query. The original query's BM25 score is also included as a feature, and we add an RM3-based feature to capture pseudo-relevance feedback signals. In our implementation, Q' for RM3 is an expanded version of the original query Q computed using a small set of documents (S) already identified as relevant by the reranker (initially, this could be the top results from R_0 [31], and it can be updated as more feedback is sampled). The RM3 feature for document d is essentially the query-likelihood or relevance model score of d under this expanded query Q', which measures the vocabulary overlap between d and the known relevant set of documents (already ranked). Intuitively, while the BM25 features capture direct term overlap with each query formulation, the RM3 feature captures indirect relevance: it indicates whether d shares terms with other documents that were deemed relevant, thereby serving as a drift-resistant signal, which results in $\vec{x}_d \in \mathcal{R}^{m+2}$.

3.3 Surrogate Relevance Model

We denote our surrogate relevance model as a linear function $s(d; \vec{w})$ that estimates the relevance of document d given the feature vector: $s(d; \vec{w}) = \vec{w}^\top \vec{x}_d$, where $\vec{w}$ is a weight vector that the model will learn using reranker feedback. This estimated utility using an inexpensive surrogate is utilized to prioritize documents for sampling reranker feedback as explained in the following section. Note that the re-ranker feedback is sampled with respect to the original query which serves as a reference point. This also aligns with the notion in query reformulation, to generate query variations that preserve the semantics of the original query [7, 14, 43]. Each component of $\vec{w}$ corresponds to the importance of the associated feature (and thereby the associated reformulation or original query). We first initialize $\vec{w}$ set to some default (e.g., small random) values. Learning $\vec{w}$ is effectively learning which reformulations are useful:

if reformulation Q_i tends to retrieve relevant documents, the weight w_i will increase; if Q_i retrieves mostly irrelevant documents, w_i will decrease, potentially becoming very small if that reformulation is consistently misleading. We frame this *surrogate learning problem* in terms of an *online learning* or bandit problem.

3.4 Modeling documents as multi-arm bandits

In a multi-armed bandit scenario, each arm is an option that can be sampled for obtaining feedback in the form of rewards from an oracle. In our setting, each document can be cast as an "arm" with reformulations implicitly encoded as part of the utility estimation of the arm using a linear surrogate as described above, and the goal is to maximize reward by sequentially selecting arms and learning from feedback. In our setting, each candidate document $d \in C$ (an arm) has an unknown true relevance reward with respect to the original query, which we can only observe by sampling from the expensive reranker (teacher) ψ. The feature vector $\vec{x}_d$ is known for each arm d, as it is computed from simple lexical query-document similarities as shown in Eq. 1. We assume there is an unknown weight vector $\vec{w}$ such that the expected reward of document d (i.e., relevance score from the reranker) is approximately $\vec{w}^\top \vec{x}_d$. This is the standard linear bandit assumption, and it implies that our linear surrogate model is capable of predicting relevance if we find the right weights. The challenge is to learn $\vec{w}$ by sampling a limited number of arms (documents to send to the reranker) and observing their rewards (reranker scores). This is akin to a stochastic linear bandit problem [33], where each teacher evaluation provides a noisy signal of the true relevance.

To learn $\vec{w}$, we define a loss function that quantifies the error in our surrogate's predictions. Whenever the teacher ψ evaluates a document d for query Q, we get a relevance score or label $y = \psi(Q, d)$ (e.g., a relevance score from MonoT5). We want our surrogate to accurately estimate this value. For a given weight vector $\vec{w}$, the estimation error for document d can be written as:

$$\mathcal{E}(\vec{w}; Q, d, \vec{x}_d) = \frac{1}{2}\Big(\psi(Q, d) - s(d; \vec{w})\Big)^2 \qquad (2)$$

The surrogate model's goal is to find weights $\vec{w}$ that minimize this error, using linear least squares, for the documents that have been evaluated by the teacher so far. This setup draws inspiration from the linear bandit [29] and online regression that has also been used in document ranking recently [32, 33]. In particular, since each document (arm) can only be evaluated once by the teacher, our setting can be seen as an instance of **disposable bandits** [15], where each arm is sampled only once. We also impose a strict overall budget c on the total number of teacher calls, to make the learning process sample-efficient and hence our approach can be seen as **budgeted disposable bandits**.

3.5 Budget-aware Online Optimization

Once we have the candidate pool C and the surrogate model structure in place, the next step is to select which documents to send to the teacher for evaluation, under the constraint that we can only afford B teacher evaluations in each round, with c being the total budget. Our strategy is an iterative, feedback-driven selection that uses the estimated relevance from the surrogate model to decide which documents (and implicitly reformulations) to focus on.

Algorithm 1 The ReformIR algorithm

Input: Original query Q, initial retrieved pool R_0, batch size b, reranking budget c, feature vector $\vec{x}_d$ for each document d
Output: Scored result set R_1

1: $R_1 \leftarrow \emptyset$ ▷ Documents scored by the teacher
2: $\mathcal{Q} \leftarrow$ Reformulate(Q) ▷ Generate reformulated queries
3: **for all** $Q_i \in \mathcal{Q}$ **do**
4: $D_i \leftarrow \text{BM25}(Q_i)$
5: **end for**
6: $C \leftarrow R_0 \cup \bigcup_i D_i$ ▷ Candidate pool (documents as arms)
7: Initialize $\vec{w}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{1})$, $t \leftarrow 1$ ▷ Surrogate weights and iteration counter
8: **do**
9: **for all** $d \in C$:
10: $s(d; \vec{w}_t) \leftarrow \vec{w}_t^\top \vec{x}_d$ ▷ Estimated utility
11: $B \leftarrow$ top-b documents from C by $s(d; \vec{w}_t)$
12: $B \leftarrow B$ truncated to respect remaining budget c
13: **if** $|B| > 0$ **then**
14: **for all** $d \in B$:
15: $y_d \leftarrow \psi(Q, d)$ ▷ Teacher reranking (MonoT5)
16: Update $\vec{w}_{t+1}$ by minimizing
$$\sum_{d \in B} \tfrac{1}{2}\big(\psi(Q, d) - \vec{w}_t{}^\top \vec{x}_d\big)^2$$
▷ Using linear least-squares optimization
17: **end if**
18: $R_1 \leftarrow R_1 \cup \{(d, y_d) : \forall d \in B\}$
19: $C \leftarrow C \setminus B$
20: $t \leftarrow t + 1$
21: **while** $|R_1| < c$ and $C \neq \emptyset$

Algorithm 1 implements the core online relevance estimation loop of ReformIR, which adaptively selects documents for expensive reranking under a strict budget. Starting from the original query Q, the algorithm first generates a set of reformulated queries $\mathcal{Q}$ (**Step 2**) and retrieves a candidate pool C (**Steps 3,4**) by issuing each reformulation independently to a first-stage retriever and combining the results with the initial retrieval R_0 pool from the original query (**Step 6**). Each document $d \in C$ is represented by a feature vector $\vec{x}_d$ encoding its relevance signals with respect to the reformulations and the original query. A linear surrogate model with weights $\mathbf{w}_t$ estimates the utility of each document via $s(d; \mathbf{w}_t) = \mathbf{w}_t^\top \vec{x}_d$ (**Step 10**). At each iteration, the algorithm selects a small batch of documents with the highest estimated utility, subject to the remaining reranking budget c, and queries a strong teacher reranker ψ (MonoT5) to obtain relevance scores anchored to the original query (**Steps 13-15**). These teacher scores are then used to update the surrogate weights by minimizing a squared prediction error, effectively performing online feature selection over reformulations (**Step 16**). Scored documents are removed from the candidate pool and added to the result set (**Steps 18,19**), and the process repeats until the budget is exhausted. Through this iterative loop, ReformIR concentrates the reranking effort on documents retrieved by the most effective reformulations, thereby increasing recall while actively suppressing query drift.

By concentrating the limited teacher evaluations on the most promising documents (as predicted by the surrogate), this approach ensures that we find as many relevant documents as possible (high recall) while spending very few evaluations on irrelevant ones. Importantly, because the teacher always evaluates documents with respect to the original query Q, it provides a *stable reference point* that prevents the system from drifting into evaluating documents that might only be relevant to a misconstrued reformulation.

During this process, features corresponding to reformulations that lead to many low-quality documents will be recognized and down-weighted early. For example, if Q_j is a reformulation that retrieves mostly irrelevant documents, those documents (when eventually selected for reranker feedback) will receive low y_d scores, and the weight w_j corresponding to Q_j's BM25 feature will decrease, because the surrogate will learn that a high BM25 score on Q_j doesn't correlate with true relevance. Conversely, a reformulation that consistently retrieves relevant documents will achieve high teacher scores for those documents, pushing its weight upward. This adaptive reweighting is done implicitly through the updates to surrogate.

3.5.1 Efficiency considerations. Another important aspect is efficiency: by focusing on top-scoring candidates first, we maximize the chance of finding relevant documents early. If the budget c is much smaller than $|C|$, the surrogate's guidance ensures we pick the likely-best candidates rather than randomly sampling the pool.

4 Experimental Setup

We aim to answer the following research questions:
RQ1: How effective is adaptive prioritization and ranking with ReformIR compared to existing reformulation methods?
RQ2: How robust is ReformIR to query drift as the number of reformulations increases?
RQ3: Under a fixed inference budget, is it more effective to allocate LLM capacity to upstream query reformulation with adaptive optimization (ReformIR) or to downstream document reranking?

4.1 Datasets

We evaluate on the MSMARCO passage corpus [27] and validate on the TREC Deep Learning (DL19–DL22) benchmarks [6]. Specifically, MSMARCO passage-v1 is used with DL19 and DL20, while the de-duplicated MSMARCO passage-v2 corpus is used with DL21 and DL22, along with their corresponding de-duplicated qrels. All indices are built using Terrier [28]. We report ranking effectiveness using nDCG@c and retrieval effectiveness using Recall@c under re-ranking budgets $c \in \{50, 100\}$.

4.2 Ranking and Reformulation Models

We use BM25 as the initial retriever via PyTerrier [21] and MonoT5-base as the cross-encoder ranker, fine-tuned on MSMARCO for all baselines and our approach. For query reformulation, we use `gpt-4o-mini` in the main experiments (in Tables 1 and 2) and additionally evaluate open-source LLMs of varying scales, including Qwen2.5 0.5B and 32B [30]. All experiments were carried out on 4 NVIDIA GeForce RTX 2080Ti GPUs.

4.3 Baselines

4.3.1 Query Reformulation baselines. We compare ReformIR against representative training-free generative reformulation methods spanning multiple categories: (i) rephrasing-based methods like GenQR [43], and GenQREnsemble (GEns) [7], (ii) pseudo-document generation – Query2Doc [42], (iii) question–answer-based reformulation – QA-Expand [36], and (iv) pseudo-relevance feedback methods –RM3 and GenQREnsemble+PRF (GEns + PRF).

For each baseline, we follow the combination strategies recommended in the original work. Unless otherwise stated, the number of reformulations is fixed to $n = 5$, and each reformulated query / combined query retrieves up to 100 candidates for fair comparison. All reformulation pipelines are implemented using QueryGym [2].

4.3.2 Exhaustive. We also use MonoT5 as a retriever on the MS-MARCO passage corpus, scoring all documents exhaustively for a query denoted by "Exhaustive Retrieval" in the tables in Section 5.

4.4 Hyperparameters

We use a temperature of 0.5 for reformulator LLMs to ensure diversity in generated reformulations while balancing quality, as lower temperatures lead to redundant queries. We do not use a higher temperature as this leads to more noisy reformulations due to hallucinations. While the default max_tokens for all approaches is set to 256, for query2doc, we increase this to 800 as the approach is expected to generate a longer pseudo-document compared to other approaches. In ReformIR we set $|S| = 15$, which is the number of top documents to consider from the already ranked pool for computing RM3 features. We use DL19 set as a validation set for tuning hyperparameters and DL20, DL21, and DL22 as test sets. For *RM*3, we set *fb_docs* to 5 and *fb_terms* to 10, and the *original_query_weight* to 0.3. The weights of ReformIR are initialized as described in Algorithm 1.

5 Results and Analysis

5.1 Query Reformulation based Ranking Effectiveness of ReformIR

To answer **RQ1**, we compare how query reformulation approaches augmented with ReformIR affect ranking performance. The results are shown on TREC-DL19 and DL20 in Table 1 and on TREC-DL21 and DL22 in Table 2. We observe that all query reformulation approaches augmented with ReformIR outperform the classical retrieve-rerank pipeline BM25»MonoT5. For instance, on DL22 GenQR + ReformIR outperforms BM25»MonoT5 by **43.90%** at budget $c = 100$ in Recall@c. We also observe that query reformulation approaches augmented with ReformIR outperform the pseudo relevance feedback based BM25+RM3 »MonoT5. We also observe from Table 1, that some of the query reformulation approaches, when augmented with ReformIR, outperform the exhaustive retrieval approach, which scores all documents in the corpus using the expensive re-ranker without any first-stage retrieval.

Finally, comparing the effectiveness of ReformIR with respect to each query reformulation approach. We observe that GenQR, which generates multiple variants of the original query by rephrasing, has significant drift with respect to the semantics of the original query, offering only marginal gains over the retrieve-rerank pipeline (with

Table 1: Effectiveness comparison of ReformIR across reformulation methods on TREC DL19 and DL20. We report nDCG@c and Recall@c. We omit nDCG@10 our focus is more on mitigating drift to capture all relevant documents. Additionally, prior works find that nDCG@10 value saturates quickly during re-ranking [19]. Significant improvements using paired t-test, $p < 0.05$, with Bonferroni correction, over respective baselines marked with corresponding letters. Note that subscript: (F) generally denotes significance of ReformIR over the RRF variant. Best scores per block are in bold.

Pipeline	DL19				DL20			
	$c = 50$		$c = 100$		$c = 50$		$c = 100$	
	nDCG@c	Recall@c	nDCG@c	Recall@c	nDCG@c	Recall@c	nDCG@c	Recall@c
Exhaustive Retrieval								
MonoT5	0.625	0.512	0.611	0.599	0.592	0.576	0.593	0.670
BM25»MonoT5 [B]	0.541	0.389	0.563	0.488	0.559	0.478	0.581	0.584
w/ RM3	0.557	0.405	0.572	0.507	0.583	0.510	0.608	0.629
Query Reformulation								
GenQR (G)	0.553	0.401	0.574	0.510	0.583	0.525	0.610	0.638
GenQR + RRF [F]	0.547	0.394	0.569	0.505	0.582	0.522	0.617	0.644
GenQR + ReformIR	$^{G}_{BF}$**0.600**	$^{G}_{BF}$**0.453**	$^{G}_{BF}$**0.617**	$^{G}_{BF}$**0.561**	$^{G}_{BF}$**0.634**	$^{G}_{BF}$**0.594**	$^{G}_{BF}$**0.643**	$^{G}_{B}$**0.675**
GEns (GE)	0.563	0.423	0.587	0.537	0.586	0.538	0.611	0.649
GEns + RRF [F]	0.548	0.421	0.575	0.525	0.568	0.519	0.604	0.645
GEns + ReformIR	$^{GE}_{BF}$**0.622**	$^{GE}_{BF}$**0.487**	$^{GE}_{BF}$**0.634**	$_{BF}$**0.584**	$^{GE}_{BF}$**0.627**	$^{GE}_{BF}$**0.591**	$^{GE}_{BF}$**0.641**	$_{BF}$**0.671**
GEns + PRF [R]	0.547	0.412	0.564	0.508	0.569	0.520	0.601	0.625
GEns + PRF + ReformIR	$^{R}_{B}$**0.592**	$^{R}_{B}$**0.454**	$^{R}_{B}$**0.626**	$^{R}_{B}$**0.575**	$^{R}_{B}$**0.628**	$^{R}_{B}$**0.572**	$^{R}_{B}$**0.637**	$^{R}_{B}$**0.671**
QA-Expand (Q)	0.613	0.468	0.633	0.575	0.606	0.576	0.622	0.669
QA-Expand + RRF [F]	0.616	0.464	0.626	0.582	0.608	0.558	0.634	0.689
QA-Expand + ReformIR	$^{Q}_{BF}$**0.672**	$^{Q}_{BF}$**0.526**	$^{Q}_{BF}$**0.668**	$^{Q}_{BF}$**0.629**	$^{Q}_{BF}$**0.651**	$_{BF}$ **0.615**	$^{Q}_{BF}$**0.661**	$^{Q}_{BF}$**0.714**
Query2Doc (QD)	0.627	0.502	0.630	0.605	0.632	0.596	0.649	0.688
Query2Doc + ReformIR	$^{QD}_{B}$**0.671**	$_{B}$ **0.538**	$^{QD}_{B}$**0.673**	$_{B}$ **0.639**	$_{B}$ **0.662**	$_{B}$**0.616**	$^{QD}_{B}$**0.680**	$_{B}$ **0.730**

Table 2: Effectiveness comparison of ReformIR across reformulation methods on TREC DL21 and DL22. We report nDCG@c and Recall@c only. Significant improvements using paired t-test, $p < 0.05$, with Bonferroni correction, over respective baselines marked with corresponding letters. Best scores per block are in bold.

Pipeline	DL21				DL22			
	$c = 50$		$c = 100$		$c = 50$		$c = 100$	
	nDCG@c	Recall@c	nDCG@c	Recall@c	nDCG@c	Recall@c	nDCG@c	Recall@c
Baseline Retrieval								
BM25»MonoT5 [B]	0.436	0.242	0.433	0.331	0.290	0.115	0.275	0.164
w/ RM3	0.455	0.274	0.457	0.375	0.287	0.115	0.275	0.161
Query Reformulation								
GenQR (G)	0.382	0.224	0.398	0.318	0.292	0.136	0.288	0.193
GenQR + RRF [F]	0.390	0.226	0.407	0.324	0.291	0.137	0.294	0.194
GenQR + ReformIR	$^{G}_{BF}$**0.482**	$^{G}_{BF}$**0.299**	$^{G}_{BF}$**0.488**	$^{G}_{BF}$**0.403**	$^{G}_{BF}$**0.352**	$^{G}_{BF}$**0.170**	$^{G}_{BF}$**0.340**	$^{G}_{BF}$**0.236**
GEns (GE)	0.421	0.266	0.432	0.370	0.289	0.135	0.288	0.189
GEns + RRF [F]	0.390	0.232	0.424	0.350	0.274	0.125	0.272	0.176
GEns + ReformIR	$^{GE}_{BF}$**0.480**	$^{GE}_{BF}$**0.308**	$^{GE}_{BF}$**0.484**	$^{GE}_{BF}$**0.402**	$^{GE}_{BF}$**0.351**	$^{GE}_{BF}$**0.171**	$^{GE}_{BF}$**0.334**	$^{GE}_{BF}$**0.227**
GEns + PRF [R]	0.408	0.251	0.410	0.332	0.251	0.100	0.248	0.145
GEns + PRF + ReformIR	$^{R}_{B}$**0.491**	$^{R}_{B}$**0.315**	$^{R}_{B}$**0.485**	$^{R}_{B}$**0.409**	$^{R}_{B}$**0.317**	$^{R}_{B}$**0.142**	$^{R}_{B}$**0.322**	$^{R}_{B}$**0.213**
QA-Expand (Q)	0.483	0.329	0.482	0.437	0.349	0.186	0.345	0.265
QA-Expand + RRF [F]	0.498	0.328	0.493	0.441	0.361	0.191	0.342	0.254
QA-Expand + ReformIR	$^{Q}_{BF}$**0.538**	$_{BF}$**0.356**	$^{Q}_{BF}$**0.523**	$_{B}$**0.463**	$^{Q}_{BF}$**0.403**	$^{Q}_{BF}$**0.214**	$_{BF}$**0.378**	$_{BF}$**0.288**
Query2Doc (QD)	0.456	0.316	0.466	0.420	0.387	0.229	0.366	0.306
Query2Doc + ReformIR	$^{QD}_{B}$**0.531**	$_{B}$ **0.354**	$^{QD}_{B}$**0.533**	$^{QD}_{B}$ **0.481**	$^{QD}_{B}$**0.425**	$_{B}$ **0.240**	$^{QD}_{B}$**0.410**	$_{B}$ **0.325**

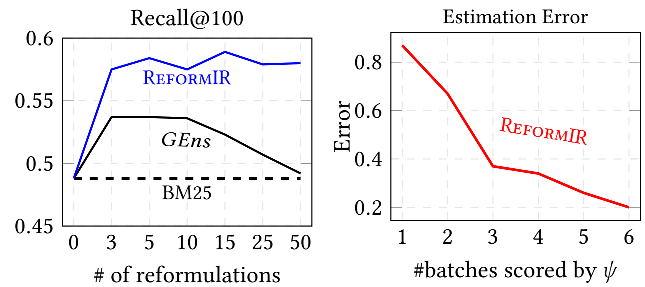


Figure 3: Recall when the number of reformulations increases *GEns* and estimation error (right) when the number of batches of documents scored by cross-encoder (ψ) varies for ReformIR comparison on the TREC DL19 dataset for ranking budget of 100 and batch of size 16.

base query). We observe that in general this primarily occurs because some of the reformulated queries drift from the semantics of the original query, causing deterioration of the candidate document pool retrieved for ranking. This demonstrates the need for prioritizing query reformulations adaptively reduce drift to enhance downstream ranking performance.

Augmenting GenQR with ReformIR, significantly reduces drift, as observed by Recall@50 on DL21, where performance improves by **33.48%** when compared to GenQR and by **32.3%** compared to the RRF variant of GenQR. ReformIR jointly determines the reformulations to prioritize through learned weights and to prioritize the most promising document for ranking through estimated utilities.

In addition to prioritizing queries, we posit that our approach also enables prioritizing key documents from the pool, balancing exploration and exploitation. Our joint optimization setup leverages BM25 scores of reformulated queries and documents, combined with RM3 features obtained using documents already deemed relevant by the reranker. Here, ReformIR could potentially *exploit* documents with high relevance (BM25) scores with respect to reformulated queries or could decide to *explore* documents with high lexical similarity to terms from documents already deemed relevant.

Note that the candidate documents pool in RRF variants is similar to the ReformIR, but RRF considers all reformulations to be equally important when performing rank fusion. Although this is intended to capture multiple facets of the query, it often aggregates across redundant or low-quality candidates as also observed in prior work [36] resulting in degradation of performance.

While QA-Expand generates focused question-answer pairs to mitigate drift, the generated questions could potentially focus on implicit facets of the query that are not critical, resulting in some of the questions and answers being irrelevant to the original query. However, as we observe from results in Tables 1 and 2, ReformIR helps adaptively prioritize the most relevant queries (question-answer pairs) and documents, leading to gains of **15.05%** for Recall@50 on DL22 and upto **12.39%** on DL19.

We also provide a concrete example of drift mitigation in QA-Expand by ReformIR from our manual analysis as shown in **Figure 1b** (Note: here reformulations are QA pairs but we omit answers for readability and also due to space constraints). In this example, we observe that reformulated query 1 has a high individual recall

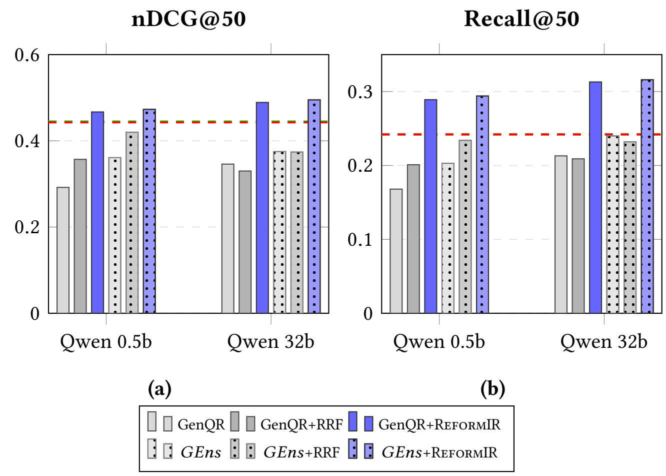


Figure 4: Impact of using LLMs of different sizes as query reformulators. The red dashed line (BM25»RankLlama & BM25»RankZephyr) serves as a baseline and note that nDCG@50 and Recall@50 for both baselines have marginal difference causing overlap in plot. Blue bars represent ReformIR, while dotted patterns indicate ensemble variants.

compared to queries 2 and 3. This is primarily because reformulated query 3, which poses a question about the purpose of the ingredients, drifts from the core semantics of the query, which concerns regarding the list of ingredients of `supartz injections` and not their purpose. We also observe that reformulated query 1 is a better phrasing of the ambiguous query compared to query 2. We observe that the learned weights in ReformIR weigh query 1 are highly relative to the other two reformulations, demonstrating the drift mitigation. This results in a final recall of **0.36** compared to baseline, which has a lower recall of **0.26** for this query.

Query2doc departs from other reformulation approaches by generating pseudo-documents that might potentially contain relevant terms, and the authors claim it does not rely [42] on initial retrieval results, which might be noisy or irrelevant. While the pseudo-documents contain additional context and contribute additional relevant terms, they may also sometimes contribute significantly to drift due to hallucination. We particularly observe this issue in DL21, where Query2Doc + ReformIR offers statistically significant gains of upto **14.52%** in Recall@100. It also achieves **16.45%**, **14.38%** gains in nDCG@50 and nDCG@100 respectively. This demonstrates that ReformIR can help automatically downweigh low-quality queries (pseudo-docs) that retrieve irrelevant documents when drift occurs with respect to the original query, with help of reranker feedback.

Insight 1: *ReformIR serves as a training-free adapter mitigating drift in query reformulations, leading to gains of upto* ***33.48%****. It achieves this through joint optimization over reformulations and documents from the candidate pool using iterative feedback*

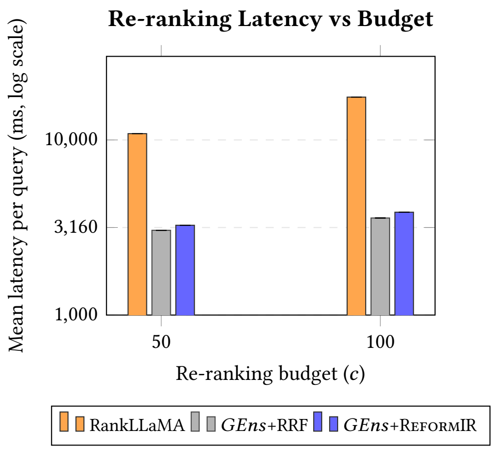


Figure 5: Mean latency per query (log scale) at different re-ranking budgets. Qwen2.5 (0.5B) is used as a reformulator for *GEns*+RRF and *GEns*+ReformIR with MonoT5 as re-ranker. Std-dev is small, hence not visible.

5.2 Effect of ReformIR with scaling the number of query reformulations

To answer **RQ2**, we vary the number of reformulations and observe the effect of drift on retrieval and ranking performance. The results are as shown in Figures 1a, 3 (left). We observe that in Figure 1a, GenQR gradually reduces in ranking performance with an increase in the number of reformulations $n = 3, 5, 10, ..., 25, 50$. We observe that this could primarily result from drift in the reformulated queries, resulting in low-quality results from first-stage retrieval. Further, at a large number of reformulations $n = 50$, the performance drops sharply, rendering the performance worse than the classical retrieve-rerank baseline (BM25»MonoT5) using the base query, indicating that a significant number of reformulated queries cause drift.

However, we observe that GenQR augmented with ReformIR leads to high and stable Recall@100 as observed in Figure 1a. We observe that this is primarily because our approach adaptively prioritizes the documents to consider through our simple surrogate model which is iteratively updated based on ranker feedback. From Figure 3 (right), we observe that the surrogate's estimation error decreases as more feedback is sampled to update the surrogate. When computing the utility of a document, a weighted combination of BM25 scores with respect to each reformulation and related RM3 features are computed. Hence, by consequence, these weights of the surrogate model updated using ranker feedback, also up-weighs or down-weighs the query reformulations based on their contribution to the document's utility and ranker feedback. Hence, query reformulations which drift from semantics of the original query as indicated by ranker feedback, and the corresponding documents which are irrelevant are not prioritized in ReformIR. This results in adaptive filtering of such low-quality queries and corresponding documents from the candidate pool, resulting in high recall and ranking performance. We observe similar results for other reformulation approaches like GEns in the presence of drift as observed in Figure 3 (left). Hence, with an increasing number of reformulations, ReformIR provides an adaptive sample-efficient mechanism drift effectively compared to existing heuristic and rank fusion-based approaches.

Insight 2: *Scaling the number of reformulations significantly degrades performance for classical reformulation approaches. However, when augmented with ReformIR, ranking performance remains stable and better than the classical variants.*

5.3 Efficiency of ReformIR vs LLM based re-ranking and variation in ReformIR with generators of different scales

To answer **RQ3**, we also vary the LLM used for reformulation and observe the impact on ranking performance across several reformulation approaches with and without ReformIR. We also compare ReformIR with MonoT5 re-ranker to LLM based re-rankers like RankLlaMA without reformulation to analyze whether it is efficient and effective to use LLMs for reformulation vs LLMs for re-ranking.

5.3.1 *Effect of ReformIR with generative models of different scales.*

We first analyze the effect of ReformIR across two different reformulation approaches when open-source LLMs of different scales are employed. The results can be observed in Figure 4. We experiment with different open-source LLM - Qwen series with models of different scales - Qwen (0.5B and 32B) and with MonoT5 as the reranker. We observe that GEns suffers from a drop in performance at budgets $c = 50$ and $c = 100$. We observe that this is primarily because some reformulations from Qwen models comprise hallucinations when compared to `gpt-4o-mini`. Additionally, GEns ensembles reformulations from multiple variations of instructions to LLM, which might result in compounding semantic drift. However, we observe that GEns + ReformIR aids in mitigating drift, offering upto **44.83%** gains in Recall@50 and **22.87%** in Recall@100.

We also perform experiments with one of the larger-scale Qwen models (32B) and observe little to no gains in the baseline performance of GenQR. However, we observe that when augmented with ReformIR, it aids in mitigating drift and leads to improvement in Recall@50 **46.95%** and similar gains in nDCG@c ($c = 50, 100$) and Recall@100. However, the gains in Recall@c and nDCG@c of reformulation approaches (GenQR, GEns) augmented with ReformIR with reformulations from Qwen 32B are only marginally better than corresponding performance on reformulations from Qwen 0.5B. This demonstrates that primarily, performance gains of ReformIR stem from mitigating drift in query reformulations. It achieves this by adaptively prioritizing high-quality queries, and the relative gains are transferable to other open-source LLMs used for reformulation. Secondly, our results in Figure 4 demonstrate that augmenting reformulation approaches with ReformIR enables leveraging less expensive LLMs (0.5B scale) for reformulation.

5.3.2 *Efficiency of ReformIR.*

We also analyze the efficiency and overhead incurred by ReformIR when compared to classical reformulation approaches. The results are as shown in Figure 5. We observe that ReformIR only adds marginal overhead over *GEns+RRF*, demonstrating that the adaptive optimization contributes to only minor overheads in terms of latency. Additionally, we also compare ReformIR to employing the RankLlaMA (7B) as a re-ranker without any reformulation to discern if it is more efficient and effective to use expensive LLMs in the ranking stage or employ LLMs (like Qwen2.5-0.5B) for the reformulation stage coupled with ReformIR. From Figure 4, we observe that RankLlaMA has worse ranking performance when compared to different reformulation approaches augmented with ReformIR, which uses only MonoT5 - a relatively inexpensive ranker compared to RankLlaMA. We also observe from Figure 5 that ReformIR is **3.3x-4.5x** more efficient than RankLlaMA based re-ranking as it employs an inexpensive MonoT5 as a ranker and can provide high-quality reformulations even with a LLM of modest scale, as shown in Figure 4.

Insights: *(1) Across diverse reformulation generators of different scales, ReformIR consistently improves performance. 2)Augmenting reformulation approaches with ReformIR only adds marginal latency cost compared to classical approaches and is still **3.3x-4.5x** more efficient than LLM based re-ranking. 3) Hence, it is more efficient to employ LLMs upstream for reformulation coupled with adaptive optimization of ReformIR than for downstream reranking.*

6 Conclusion

In this paper, we introduce ReformIR as a simple adaptive optimization approach which can reduce drift in different query reformulation approaches. We achieve this by using the relevance of documents to original query as reference point and estimate this using an inexpensive linear surrogate which employs query reformulations as features by using lexical similarities between reformulation and documents. The estimated utilities are used to adaptively sample reranker feedback and update the surrogate. This enables joint optimization for prioritization of query reformulations and documents. We demonstrate that ReformIR achieves gains in recall mitigating drift. In future, we plan to explore further on updating the reformulators online using ranker feedback and exploring even smaller LLMs (<0.5B) to further reduce latency.

Acknowledgments

This work was partially funded by the Bundesministerium für Wirtschaft und Energie (BMWE), Germany, in the context of the 8ra Initiative ("Soofi", 13IPC040E), and by the European Union's Horizon Europe Research and Innovation Programme JustREACH under Grant Agreement No 101214666. The experiments were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725 and Zeus cluster provided by Department of Computer and Systems Sciences at Stockholm University.

References

[1] Avishek Anand, Abhijit Anand, Vinay Setty, et al. 2023. Query understanding in the age of large language models. *arXiv preprint arXiv:2306.16004* (2023).

[2] Amin Bigdeli, Radin Hamidi Rad, Mert Incesu, Negar Arabzadeh, Charles L. A. Clarke, and Ebrahim Bagheri. 2025. QueryGym: A Toolkit for Reproducible LLM-Based Query Reformulation. arXiv:2511.15996 [cs.IR] https://arxiv.org/abs/2511.15996

[3] Claudio Carpineto and Giovanni Romano. 2012. A survey of automatic query expansion in information retrieval. *Acm Computing Surveys (CSUR)* 44, 1 (2012), 1–50.

[4] Xinran Chen, Ben He, Xuanang Chen, and Le Sun. 2025. Not All Terms Matter: Recall-Oriented Adaptive Learning for PLM-aided Query Expansion in Open-Domain Question Answering. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (Eds.). Association for Computational Linguistics, Vienna, Austria, 22139–22151. doi:10.18653/v1/2025.acl-long.1076

[5] Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Boston, MA, USA) *(SIGIR '09)*. Association for Computing Machinery, New York, NY, USA, 758–759. doi:10.1145/1571941.1572114

[6] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, Ellen M. Voorhees, and Ian Soboroff. 2021. TREC Deep Learning Track: Reusable Test Collections in the Large Data Regime. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 2369–2375. doi:10.1145/3404835.3463249

[7] Kaustubh D. Dhole and Eugene Agichtein. 2024. GenQREnsemble: Zero-Shot LLM Ensemble Prompting for Generative Query Reformulation. In *Advances in Information Retrieval: 46th European Conference on Information Retrieval, ECIR 2024, Glasgow, UK, March 24–28, 2024, Proceedings, Part III* (Glasgow, United Kingdom). Springer-Verlag, Berlin, Heidelberg, 326–335. doi:10.1007/978-3-031-56063-7_24

[8] Kaustubh D. Dhole, Ramraj Chandradevan, and Eugene Agichtein. 2024. Generative Query Reformulation Using Ensemble Prompting, Document Fusion, and Relevance Feedback. arXiv:2405.17658 [cs.IR] https://arxiv.org/abs/2405.17658

[9] Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. Precise Zero-Shot Dense Retrieval without Relevance Labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, Toronto, Canada, 1762–1777. doi:10.18653/v1/2023.acl-long.99

[10] Rolf Jagerman, Honglei Zhuang, Zhen Qin, Xuanhui Wang, and Michael Bendersky. 2023. Query Expansion by Prompting Large Language Models. *CoRR* abs/2305.03653 (2023). doi:10.48550/ARXIV.2305.03653 arXiv:2305.03653

[11] Nasreen Abdul Jaleel, James Allan, W. Bruce Croft, Fernando Diaz, Leah S. Larkey, Xiaoyan Li, Mark D. Smucker, and Courtney Wade. 2004. UMass at TREC 2004: Novelty and HARD. In *Proceedings of the Thirteenth Text REtrieval Conference, TREC 2004, Gaithersburg, Maryland, USA, November 16-19, 2004 (NIST Special Publication, Vol. 500-261)*, Ellen M. Voorhees and Lori P. Buckland (Eds.). National Institute of Standards and Technology (NIST). http://trec.nist.gov/pubs/trec13/papers/umass.novelty.hard.pdf

[12] Pengyue Jia, Yiding Liu, Xiangyu Zhao, Xiaopeng Li, Changying Hao, Shuaiqiang Wang, and Dawei Yin. 2024. MILL: Mutual Verification with Large Language Models for Zero-Shot Query Expansion. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, Kevin Duh, Helena Gomez, and Steven Bethard (Eds.). Association for Computational Linguistics, Mexico City, Mexico, 2498–2518. doi:10.18653/v1/2024.naacl-long.138

[13] Pengcheng Jiang, Jiacheng Lin, Lang Cao, Runchu Tian, SeongKu Kang, Zifeng Wang, Jimeng Sun, and Jiawei Han. 2025. DeepRetrieval: Hacking Real Search Engines and Retrievers with Large Language Models via Reinforcement Learning. *CoRR* abs/2503.00223 (2025). doi:10.48550/ARXIV.2503.00223 arXiv:2503.00223

[14] Hye-young Kim, Minjin Choi, Sunkyung Lee, Eunseong Choi, Young-In Song, and Jongwuk Lee. 2023. ConQueR: Contextualized Query Reduction using Search Logs. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Taipei, Taiwan) *(SIGIR '23)*. Association for Computing Machinery, New York, NY, USA, 1899–1903. doi:10.1145/3539618.3591966

[15] Melda Korkut and Andrew Li. 2021. Disposable Linear Bandits for Online Recommendations. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 5 (May 2021), 4172–4180. doi:10.1609/aaai.v35i5.16540

[16] Victor Lavrenko and W. Bruce Croft. 2001. Relevance based language models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (New Orleans, Louisiana, USA) *(SIGIR '01)*. Association for Computing Machinery, New York, NY, USA, 120–127. doi:10.1145/383952.383972

[17] Minghan Li, Honglei Zhuang, Kai Hui, Zhen Qin, Jimmy Lin, Rolf Jagerman, Xuanhui Wang, and Michael Bendersky. 2024. Can Query Expansion Improve Generalization of Strong Cross-Encoder Rankers?. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Washington DC, USA) *(SIGIR '24)*. Association for Computing Machinery, New York, NY, USA, 2321–2326. doi:10.1145/3626772.3657979

[18] Lijun Lyu and Avishek Anand. 2023. Listwise Explanations for Ranking Models Using Multiple Explainers. In *Advances in Information Retrieval*, Jaap Kamps, Lorraine Goeuriot, Fabio Crestani, Maria Maistro, Hideo Joho, Brian Davis, Cathal Gurrin, Udo Kruschwitz, and Annalina Caputo (Eds.). Springer Nature Switzerland, Cham, 653–668.

[19] Sean MacAvaney and Nicola Tonellotto. 2024. A Reproducibility Study of PLAID. In *Proceedings of the 47th International ACM SIGIR Conference on Research and*

Development in Information Retrieval (SIGIR 2024). ACM, 1411–1419. doi:10.1145/3626772.3657856

[20] Sean MacAvaney and Xi Wang. 2023. Online Distillation for Pseudo-Relevance Feedback. *CoRR* abs/2306.09657 (2023). doi:10.48550/ARXIV.2306.09657 arXiv:2306.09657

[21] Craig Macdonald, Nicola Tonellotto, Sean MacAvaney, and Iadh Ounis. 2021. PyTerrier: Declarative Experimentation in Python from BM25 to Dense Retrieval. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (Virtual Event, Queensland, Australia) *(CIKM '21)*. Association for Computing Machinery, New York, NY, USA, 4526–4533. doi:10.1145/3459637.3482013

[22] Iain Mackie, Shubham Chatterjee, and Jeffrey Dalton. 2023. Generative and Pseudo-Relevant Feedback for Sparse, Dense and Learned Sparse Retrieval. *CoRR* abs/2305.07477 (2023). doi:10.48550/ARXIV.2305.07477 arXiv:2305.07477

[23] Iain Mackie, Shubham Chatterjee, and Jeffrey Dalton. 2023. Generative relevance feedback with large language models. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2026–2031.

[24] Iain Mackie, Ivan Sekulic, Shubham Chatterjee, Jeffrey Dalton, and Fabio Crestani. 2023. GRM: generative relevance modeling using relevance-aware sample estimation for document retrieval. *arXiv preprint arXiv:2306.09938* (2023).

[25] Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021. Generation-Augmented Retrieval for Open-Domain Question Answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (Eds.). Association for Computational Linguistics, Online, 4089–4100. doi:10.18653/v1/2021.acl-long.316

[26] Donald Metzler and W. Bruce Croft. 2007. Latent concept expansion using markov random fields. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Amsterdam, The Netherlands) *(SIGIR '07)*. Association for Computing Machinery, New York, NY, USA, 311–318. doi:10.1145/1277741.1277796

[27] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016 (CEUR Workshop Proceedings, Vol. 1773)*, Tarek Richard Besold, Antoine Bordes, Artur S. d'Avila Garcez, and Greg Wayne (Eds.). CEUR-WS.org. https://ceur-ws.org/Vol-1773/CoCoNIPS_2016_paper9.pdf

[28] Iadh Ounis, Gianni Amati, Vassilis Plachouras, Ben He, Craig Macdonald, and Douglas Johnson. 2005. Terrier information retrieval platform. In *Advances in Information Retrieval: 27th European Conference on IR Research, ECIR 2005, Santiago de Compostela, Spain, March 21-23, 2005. Proceedings 27*. Springer, 517–519.

[29] Kiran Purohit, V Venktesh, Sourangshu Bhattacharya, and Avishek Anand. 2025. Sample Efficient Demonstration Selection for In-Context Learning. In *International Conference on Machine Learning*. PMLR, 49959–49982.

[30] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 Technical Report. arXiv:2412.15115 [cs.CL] https://arxiv.org/abs/2412.15115

[31] Mandeep Rathee, Sean MacAvaney, and Avishek Anand. 2025. Quam: Adaptive Retrieval through Query Affinity Modelling. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining* (Hannover, Germany) *(WSDM '25)*. Association for Computing Machinery, New York, NY, USA, 954–962. doi:10.1145/3701551.3703584

[32] Mandeep Rathee, Venktesh V, Sean MacAvaney, and Avishek Anand. 2025. Breaking the Lens of the Telescope: Online Relevance Estimation over Large Retrieval Sets. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Padua, Italy) *(SIGIR '25)*. Association for Computing Machinery, New York, NY, USA, 2287–2297. doi:10.1145/3726302.3729910

[33] Clémence Réda, Emilie Kaufmann, and Andrée Delahaye-Duriez. 2021. Top-m identification for linear bandits. In *The 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, April 13-15, 2021, Virtual Event (Proceedings of Machine Learning Research, Vol. 130)*, Arindam Banerjee and Kenji Fukumizu (Eds.). PMLR, 1108–1116. http://proceedings.mlr.press/v130/reda21a.html

[34] JJ Rocchio. 1971. Relevance feedback in information retrieval. *The SMART Retrieval System-Experiments in Automatic Document Processing/Prentice Hall* (1971).

[35] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*. Vol. 39. Cambridge University Press Cambridge.

[36] Wonduk Seo and Seunghyun Lee. 2025. QA-Expand: Multi-Question Answer Generation for Enhanced Query Expansion in Information Retrieval. *CoRR* abs/2502.08557 (2025). doi:10.48550/ARXIV.2502.08557 arXiv:2502.08557

[37] Tao Shen, Guodong Long, Xiubo Geng, Chongyang Tao, Yibin Lei, Tianyi Zhou, Michael Blumenstein, and Daxin Jiang. 2024. Retrieval-Augmented Retrieval: Large Language Models are Strong Zero-Shot Retriever. In *Findings of the Association for Computational Linguistics: ACL 2024*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, Bangkok, Thailand, 15933–15946. doi:10.18653/v1/2024.findings-acl.943

[38] Jaspreet Singh and Avishek Anand. 2020. Model agnostic interpretability of rankers via intent modelling. In *FAT* '20: Conference on Fairness, Accountability, and Transparency, Barcelona, Spain, January 27-30, 2020*, Mireille Hildebrandt, Carlos Castillo, L. Elisa Celis, Salvatore Ruggieri, Linnet Taylor, and Gabriela Zanfir-Fortuna (Eds.). ACM, 618–628. doi:10.1145/3351095.3375234

[39] Mingyang Song and Mao Zheng. 2024. A Survey of Query Optimization in Large Language Models. *CoRR* abs/2412.17558 (2024). doi:10.48550/ARXIV.2412.17558 arXiv:2412.17558

[40] Howard R. Turtle and James Flood. 1995. Query Evaluation: Strategies and Optimizations. *Inf. Process. Manag.* 31, 6 (1995), 831–850. doi:10.1016/0306-4573(95)00020-H

[41] Haoyu Wang, Ruirui Li, Haoming Jiang, Jinjin Tian, Zhengyang Wang, Chen Luo, Xianfeng Tang, Monica Xiao Cheng, Tuo Zhao, and Jing Gao. 2024. BlendFilter: Advancing Retrieval-Augmented Large Language Models via Query Generation Blending and Knowledge Filtering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (Eds.). Association for Computational Linguistics, Miami, Florida, USA, 1009–1025. doi:10.18653/v1/2024.emnlp-main.58

[42] Liang Wang, Nan Yang, and Furu Wei. 2023. Query2doc: Query Expansion with Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 9414–9423. doi:10.18653/v1/2023.emnlp-main.585

[43] Xiao Wang, Sean MacAvaney, Craig Macdonald, and Iadh Ounis. 2023. Generative Query Reformulation for Effective Adhoc Search. *CoRR* abs/2308.00415 (2023). doi:10.48550/ARXIV.2308.00415 arXiv:2308.00415

[44] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 24824–24837. https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf

[45] Jinxi Xu and W. Bruce Croft. 1996. Query expansion using local and global document analysis. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Zurich, Switzerland) *(SIGIR '96)*. Association for Computing Machinery, New York, NY, USA, 4–11. doi:10.1145/243199.243202

[46] Chengxiang Zhai and John Lafferty. 2001. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the Tenth International Conference on Information and Knowledge Management* (Atlanta, Georgia, USA) *(CIKM '01)*. Association for Computing Machinery, New York, NY, USA, 403–410. doi:10.1145/502585.502654

[47] Le Zhang, Yihong Wu, Qian Yang, and Jian-Yun Nie. 2024. Exploring the Best Practices of Query Expansion with Large Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (Eds.). Association for Computational Linguistics, Miami, Florida, USA, 1872–1883. doi:10.18653/v1/2024.findings-emnlp.103