Query Decomposition for RAG: Balancing Exploration-Exploitation

Roxana Petcu[1], Kenton Murray[2], Daniel Khashabi[2], Evangelos Kanoulas[1], Maarten de Rijke[1], Dawn Lawrie[2], Kevin Duh[2]

[1]University of Amsterdam, [2]Johns Hopkins University
r.m.petcu@uva.nl, kenton@jhu.edu, danielk@cs.jhu.edu, e.kanoulas@uva.nl, m.derijke@uva.nl, lawrie@jhu.edu, kevinduh@cs.jhu.edu

Abstract

Retrieval-augmented generation (RAG) systems address complex user requests by decomposing them into subqueries, retrieving potentially relevant documents for each, and then aggregating them to generate an answer. Efficiently selecting informative documents requires balancing a key trade-off: (i) retrieving broadly enough to capture all the relevant material, and (ii) limiting retrieval to avoid excessive noise and computational cost. We formulate query decomposition and document retrieval in an exploitation-exploration setting, where retrieving one document at a time builds a belief about the utility of a given sub-query and informs the decision to continue exploiting or exploring an alternative. We experiment with a variety of bandit learning methods and demonstrate their effectiveness in dynamically selecting the most informative sub-queries. Our main finding is that estimating document relevance using rank information and human judgments yields a 35% gain in document-level precision, 15% increase in α-nDCG, and better performance on the downstream task of long-form generation. Code is available on GitHub.[1]

1 Introduction

Complex user queries usually involve discourse operators such as the exclusion of information (Zhang et al., 2025), negation (Petcu et al., 2025; Weller et al., 2024; van den Elsen et al., 2025), or missing entities (Qi et al., 2019; Bhargav et al., 2022), and often require retrieving evidence found in multiple documents. One way to handle them is to decompose the request into atomic sub-queries (Khot et al., 2023), as shown in Figure 1. Retrieving documents independently for each sub-query leads to coverage of complex information needs by maximizing recall. However, it may result in a large number of documents, of which many are irrelevant (Kim et al., 2025). This is problematic in two ways. First, too many documents cannot fit in the context window of an LLM, without being arbitrarily truncated, potentially removing vital information. Second, including a large proportion of irrelevant documents introduces noise in the generation (Jin et al., 2024). Mitigating these problems involves filtering by either human annotators or LLM agents. This motivates our core question: *how to efficiently identify which sub-queries are likely to retrieve relevant information and what constitutes an appropriate retrieval depth for each sub-query*?

Existing approaches to sub-query decomposition and document retrieval lack a principled mechanism for allocating documents, with most methods retrieving a fixed number of documents regardless of sub-query utility. Meanwhile, we want to study a method that adaptively decides, under a fixed budget, whether to continue retrieving documents from an observed, promising sub-query or to explore alternatives that may yield more relevant evidence, while avoiding irrelevant overlap. A multi-armed bandit framework naturally captures this process: each sub-query is treated as an arm, for which each observed document provides evidence of its utility. Framing query decomposition and document selection as a bandit problem addresses two core challenges of complex information needs. First, retrieval is inherently sequential and budget-constrained, as it is impossible to verify all documents. Second, the relevance of a sub-query is initially uncertain, while our belief of its relevance builds with each retrieved document.

Figure 1 illustrates our setting. We estimate the utility of each sub-query, modeled as an arm, by observing the relevance of one document at a time. The choice of assessment directly influences the cost. By estimating utility under a fixed budget, i.e., at each step the system makes a choice between going down the ranked list of a certain sub-query or retrieving from a new one, the process becomes

[1]https://anonymous.4open.science/r/query-decomposition-bandits-2A0D

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics
Volume 1: Long Papers, pages 6857–6871

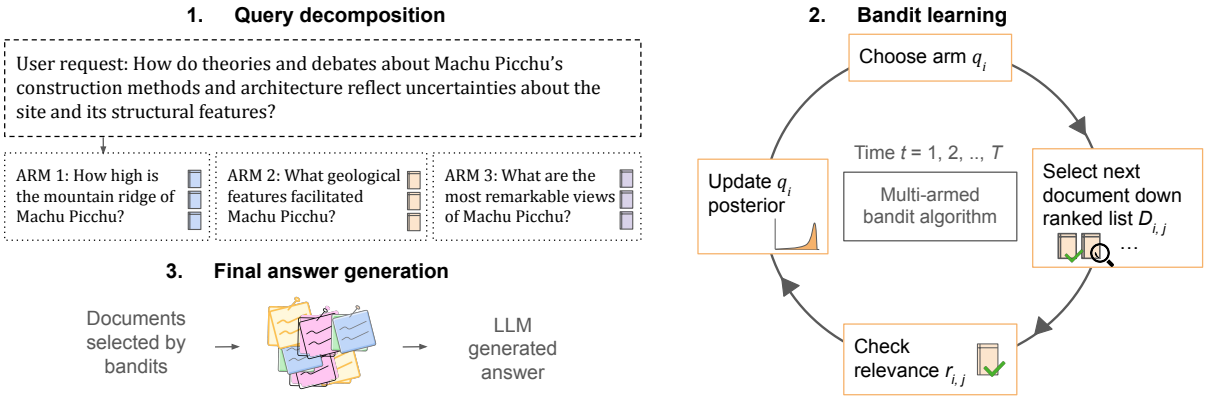

Figure 1: A user request is first decomposed into sub-queries. Bandit learning iteratively selects a sub-query (arm), retrieves a document down its ranked list, observes its relevance, and updates the sub-query posterior belief over time. The selected documents across iterations are then used as evidence in a RAG setting to generate a grounded answer, balancing exploration (more sub-queries) and exploitation (more documents per sub-query).

considerably more efficient. This perspective is complementary to recent approaches on efficient selection under computational constraints, such as compute-efficient re-ranking (Podolak et al., 2025) and training data selection (Petcu and Maji, 2024). In the example shown in Figure 1 we assume access to a user request, its decomposition, document relevance (assessed by either LLM judges or human annotators), and a search engine We aim to answer the following research questions:

(RQ1) How can query selection be framed as a multi-armed bandit problem? Does bandit learning outperform full exploitation and full exploration strategies? What strategies best balance exploration with exploitation?

(RQ2) Does document selection using multi-armed bandits improve evaluation metrics on the downstream task of report generation?

(RQ3) Can bandit learning be used to guide hierarchical sub-query decomposition?

For answering RQ1, we model query decomposition and document retrieval in an efficient way using reinforcement learning (RL) policies in a multi-armed bandit setting, demonstrating that estimating Bernoulli distributions boosts relevance estimates by 17% compared to simply going down the ranked list. With RQ2, we look into the performance of using an optimal subset of documents for report generation, in which we improve on evaluation metrics such as nugget coverage and sentence support. With RQ3, we examine the use of hierarchical, multi-level sub-query decomposition, which yields a 30% precision gain over selecting all documents from a single-level decomposition.

2 Related Work

RAG systems. RAG systems extend language models with access to external knowledge from retrieved evidence (Lewis et al., 2021; Asai et al., 2020; Soudani et al., 2024). While this approach brings advantages in grounding the generation into real information (Askari et al., 2025), it treats the query as a unitary piece of information. There are variants of RAG systems that handle complex queries, such as multi-step RAG systems which perform iterative retrieval (Gu et al., 2025), or feedback-based retrieval implemented with RLHF or model-estimated policies for re-generating the query until a satisfying answer can be composed by the system (Deng et al., 2022; Rafailov et al., 2024; Jin et al., 2025). This line of work follows earlier efforts in RL-based query reformulation (Buck et al., 2018), which have been extended to multi-modal and retrieval tasks (Odijk et al., 2015).

Query decomposition. Research in discourse phenomena such as exclusion (Zhang et al., 2025), negation (Weller et al., 2024; Petcu et al., 2025), and compositions of logic operators (Zhang et al., 2024) shows that such formulations are difficult to encode by retrieval models (Krasakis et al., 2025). These cases often benefit from decomposing the original request into simpler, unitary sub-queries (Yao et al., 2023). Query decomposition can be performed semantically using external models (Khot et al., 2023). However, naively splitting a complex query and retrieving documents for each sub-query does not necessarily improve results: longer contexts can degrade downstream performance (Shao et al., 2025). To address this, we propose an

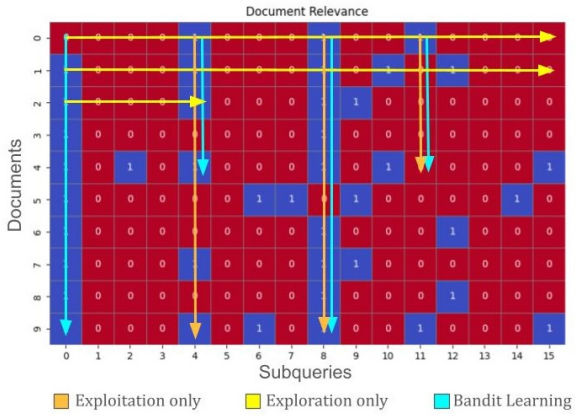

Figure 2: Exploration–exploitation across n subqueries and m documents in an offline setting; colors indicate document relevance, and arrows show how exploitation, exploration, and bandit-based policies allocate effort across sub-queries.

adaptive allocation of retrieval budgets across sub-queries.

Research gap. We position our work at the intersection of query decomposition, using LLMs for unitary information needs, and efficient retrieval of documents by observing and modeling relevance-aware distributions for each decomposed sub-query. The closest existing work applies multi-armed bandits and active learning techniques for document selection for large-scale evaluation, and for building fair IR test collections (Li and Kanoulas, 2017; Rahman et al., 2020; Voorhees, 2018).

3 Methodology

3.1 The exploitation-exploration problem

The exploration-exploitation dilemma is a fundamental decision-making concept that balances the act of exploiting known low-risk options with exploring unknown high-risk alternatives. A popular paradigm for this setting is the multi-armed bandit problem, which assumes access to multiple fixed choices, called arms, observed iteratively by a decision maker. The properties of each arm are initially uncertain, and the belief about their relevance is refined as more evidence is observed. A fundamental aspect of the bandit problem is that sampling from an arm does not affect the underlying distribution of that arm or any other. It can be seen as a set of real distributions $B = \{R_1, R_2, \ldots, R_K\}$, where each observed value is associated with a reward.

Algorithm 1: Thompson Sampling in Discrete Space

Input : Sub-queries $\mathcal{S} = \{s_1, \ldots, s_K\}$; Budget b

for $i = 1, \ldots, K$ **do**
  Define Beta priors $\alpha_i \leftarrow 1,\ \beta_i \leftarrow 1$
Initialize observation set $\mathcal{O} \leftarrow \emptyset$;
for $t = 1, \ldots, b$ **do**
  for $i = 1, \ldots, K$ **do**
    Sample $\theta_i \sim \text{Beta}(\alpha_i, \beta_i)$
  Select arm $a_t \leftarrow \arg\max_i \theta_i$;
  Pick next document $d_{a_t,n} \in \mathcal{D}_{a_t}$ where
    $n \leftarrow \min\{\, j \in \{1, \ldots, N\} \mid (a_t, j) \notin \mathcal{O} \,\}$;
  Observe reward $r(a_t, n)$;
  Update posterior:
    $\alpha_{a_t} \leftarrow \alpha_{a_t} + r(a_t, n)$;
    $\beta_{a_t} \leftarrow \beta_{a_t} + (1 - r(a_t, n))$;
  Save observation $\mathcal{O} \leftarrow \mathcal{O} \cup \{(a_t, n)\}$;

3.2 Bandit learning for query decomposition

Figure 2 illustrates the setting of our problem: given a user request $\mathcal{Q}$ and its decomposition $\{q_1, q_2, \ldots, q_K\}$, we retrieve a ranked list of N documents for each sub-query q_i as $\{d_{i,1}, d_{i,2}, \ldots, d_{i,N}\}$. We observe one document at a time -we retrieve a document, assess its relevance with respect to $\mathcal{Q}$, and store it as an evaluated document- while trying to maximize observed relevance while not going over a budget of b documents, where $b < N \cdot K$. We model each sub-query q_i as an arm, where its associated set $\mathcal{D}_i$ has an unknown distribution over relevance labels w.r.t. $\mathcal{Q}$. We treat each arm as an unknown distribution over $\mathcal{D}_i$ for which we maintain a posterior over its expected utility. We initialize the distribution with an uninformative (flat) prior and update the posterior after each observation over $\mathcal{D}_i$. We employ Thompson sampling, a standard multi-armed bandits (MAB) algorithm, both in discrete (Algorithm 1) and continuous space (Algorithm 2).

In discrete space, we model a Bernoulli distribution with a Beta conjugate prior, while in continuous space, we model Gaussians with Gaussian priors. Importantly, (i) we model the multi-armed bandit problem on query decomposition, where the arm chosen at time t is denoted as a_t, (ii) each observation is represented by the next document $d_{:,j}$ down a ranked list, and (iii) the reward is calculated at the document level, i.e., $d_{a_t,j}$ for the chosen arm a_t and document rank j.

3.3 Assumptions

We make several assumptions throughout this work. First, we formulate the problem in an offline supervised setting, where relevance labels for re-

trieved documents are available and can be directly used to compute rewards. Alternatively, in an online setting, the reward can be derived from other document-level signals such as retrieval scores (see our Gaussian reward in Table 1), LLM judges, or user clicks. Secondly, we assume that the ranked documents represent a good estimate of absolute document relevance, implying that the underlying ranking system operates at a sufficiently high level of performance. When this assumption does not hold, noise introduced by imperfect rankings propagate into the reward signal and perturb the estimated query representations.

3.4 Methods and rewards

We study different properties that we hypothesize to play an important role in the task of sub-query-dependent document retrieval for RAG:

Rank information: Each sub-query q_i is associated with a ranked list of retrieved documents, whose order encodes an implicit estimate of relevance. To incorporate this rank-based signal, we calculate the reward for the sub-query a_t chosen at time t as the average relevance over a local window of k documents down the ranked list: $\frac{1}{k}\sum_{i=n}^{n+k-1}\text{Relevance}(d_{a_t,i})$, where $d_{a_t,i}$ represents the i-th document retrieved for subquery chosen at time t, n is the current position in the ranked list, and $\text{Relevance}(d_{a_t,i})$ represents the relevance of one document, either as a binary label (from human or LLM judges), or a continuous relevance signal (ranking score).

Diversity of documents: We want the retrieved documents to be diverse; retrieving a relevant document that has a high content overlap with a previously observed one does not bring new information. To encourage this, we estimate the novelty of the currently observed document $d_{a_t,i}$ with respect to the set of previously observed documents $\mathcal{O}$; we model this by estimating the novelty of each sub-query using cosine similarity: $1 - \frac{\max_{(i,j)\in\mathcal{O}}\cos(d_{a_t,n},d_{i,j})+1}{2}$.

Exploration: We want to explore each sub-query at least once; we enforce this using an upper confidence bound (UCB) term $c\cdot\sqrt{\frac{1}{n}\log_2(n+1)}$, with $c\to 0$. This term diverges to infinity for subqueries with no observations, i.e., when $n\to 0$.

These factors compose our final reward policy, which we propose as a Bernoulli top-k UCB diversity-aware estimate in Eq. 1, used on line 9 in Algorithm 1:

$$
\begin{aligned}
r(a_t, n) = \\
&\frac{1}{k}\sum_{i=n}^{n+k-1}\text{Relevance}(d_{a_t,i})\cdot \\
&\left(1-\frac{\max\limits_{(i,j)\in\mathcal{O}}\cos(d_{a_t,n},d_{i,j})+1}{2}\right) \\
&+c\cdot\sqrt{\frac{\log_2(n+1)}{n}},\ \text{with } c\to 0.
\end{aligned}
\tag{1}
$$

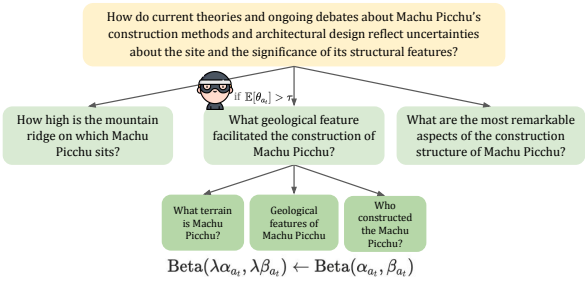

Figure 3: Hierarchical query decomposition with correlated bandits, where informative sub-queries are decomposed into arms with inherited posterior beliefs.

3.5 Hierarchical query decomposition with Correlated MAB

In hierarchical decompositions, a complex user request is iteratively split into smaller subqueries, where each can be further decomposed into more fine-grained information needs (see Figure 3). As a result, the retrieval space forms a hierarchy of subqueries and their associated document distributions, where each child sub-query inherits properties from its parent. As some sub-queries may be more informative than others, instead of uniform expansion, we selectively expand those that demonstrate high utility according to their posterior estimates. A sub-query q_i is highly informative when its estimated value is above a set informativeness threshold $\mathbb{E}_{q_i} = \frac{\alpha_i}{\alpha_i+\beta_i} > \tau$ and when it has been observed at least n times. If an informative sub-query q_i is expanded, its children $\{q_{i,1}, q_{i,2}, ..., q_{i,m}\}$ become new arms in the multi-armed bandit formalization, whose distributions are correlated with their parent sub-query q_i. This correlation is captured by an inheritance factor $\lambda \in (0, 1]$, i.e. a parent with posterior $\text{Beta}(\alpha_i, \beta_i)$ will yield a child sub-query initialized as $\text{Beta}(\lambda\alpha_i, \lambda\beta_i)$. By modeling hierarchical decomposition with correlated bandits, we do not assume independence between them.

4 Experimental Setup

We run our experiments on two datasets, which come with different properties and challenges.

4.1 NeuCLIR

Dataset. The **NeuCLIR** dataset (Lawrie et al., 2025) features decomposable user requests; the sub-queries are generated in a serialized manner, i.e., a one-level decomposition. NeuCLIR is part of TREC (Text Retrieval Conference) and is designed to evaluate information retrieval models on multilingual data. More precisely, the dataset contains complex user requests and a corpus of documents in Chinese (∼3M documents), Russian (∼5M documents), and Persian (2M documents). [2] For the purpose of this study, all documents have been translated into English. We run our experiments on the entirety of the human-annotated nugget partition of the dataset. The average length of each user request is 51.95 ± 19.46 words.

Retrieval. We decompose NeuCLIR user requests into $k = 16$ sub-queries using LLM calls as specified in prompts 6 and 7. For each generated sub-query, we run a search engine to retrieve $n = 10$ documents. The retriever used in this study is a combination of PLAID-X, a dual encoder with late interaction (Nair et al., 2022; Yang et al., 2024), learned sparse retrieval (LSR) (Nguyen et al., 2023), which combines sparse retrieval with contextualized dense embeddings, and a Qwen retriever (Bai et al., 2023; Yang et al., 2025).

4.2 ResearchyQuestions

Dataset. ResearchyQuestions (Rosset et al., 2024) is composed of $100k$ complex Bing questions that are non-factoid, multi-perspective, and require decomposition. Each instance in the dataset is decomposed into sub-queries hierarchically over two levels: *headers* (first-level sub-queries), and *sub-queries* (second-level). We build the corpus by aggregating all documents marked as relevant across the entire ResearchyQuestions dataset.

Retrieval. We apply BM25 to retrieve top 10 documents for each query. Since some instances have limited document coverage, we filter the data and keep only those where at least 10% of the retrieved documents are labeled as relevant. The final dataset used for evaluation consists of 140 instances.

[2] https://huggingface.co/neuclir

4.3 Implementation Details

Rewards. We experiment with the reward introduced in Equation 1 and with several alternatives and baselines described in Table 1. We sample documents up to a budget $b \in \{10\%, 20\%, \ldots, 100\%\}$ out of the total document set of size $N \times K$. For $b = 100\%$ we expect all policies and baselines to converge to the same performance. To eliminate noise from random policy starts, we run each experiment 1000 times. Moreover, as we decompose the user request using LLM calls, we run the decomposition 10 times for each user request, leading to a total of $19 \times 10 \times 1000$ experiments over which we average for each budget. For the Gaussian reward, we experiment with both the RFF and PLAID-X ranking scores, depending on the search engine used. For the diversity concave reward, we set hyperparameters $a = 5$ and $b = 15$.

Metrics. We evaluate performance using precision and α-nDCG (Clarke et al., 2008) for document selection. For long-form report generation, we use the Auto-ARGUE framework (Walden et al., 2025).

5 Empirical Results

Our experiments are designed to answer the research questions presented in Section 1. We answer RQ1 (can we frame document selection as a multi-armed bandit problem?) through experiments on NeuCLIR and ResearchyQuestions in Section 5.1. We answer RQ2 (do policy-observed documents lead to better performance on downstream tasks?) by long-form generation of reports on NeuCLIR in Section 5.2. We answer RQ3 (can we apply bandit-learning on hierarchical sub-queries?) by modeling correlated bandits on existing sub-query splits from ResearchyQuestions in Section 5.3.

5.1 Document selection as a multi-armed bandit problem

Figure 4 illustrates macro-precision over document budget. We highlight the main findings in Figure 4, while Figures 8 and 9 in the Appendix present all results. Exploitation and exploration-only policies perform similarly to our baselines: the exploitation-only policy returns a precision of 0.57 and exploration-only a precision of 0.55 on NeuCLIR, while both exploitation and exploitation-only achieve 0.14 precision on ResearchyQuestions. Based on results on our proposed rewards, we observe that: (i) reward policies effectively model sub-query relevance; (ii) ϵ-greedy performs

Reward type	Formula
Baseline random	$r(a_t) = \text{Relevance}(d_{a_t,n})$ where $a_t \sim \text{unif}(a_1, ..., a_K), n \sim \text{unif}(rank_1, rank_2, ..rank_m)$
Baseline random rank-aware	$r(a_t, n) = \text{Relevance}(d_{a_t}, n)$ where $a_t \sim \text{unif}(a_1, ..., a_K)$
ϵ-greedy	$r(a_t, n) = \text{Relevance}(d_{a_t,n})$ if $\text{Relevance}(d_{a_t,n-1})$ is 1 else $a_t \sim \text{unif}(a_1, ..., a_K)$
Bernoulli	$r(a_t, n) = \text{Relevance}(d_{a_t,n})$
Bernoulli UCB	$r(a_t, n) = \text{Relevance}(d_{a_t,n}) + c \cdot \sqrt{\frac{\log_2(n+1)}{n}}$ with $c \to 0$
Bernoulli top-k	$r(a_t, n) = \frac{1}{k} \sum_{i=n}^{n+k-1} \text{Relevance}(d_{a_t,i})$
Bernoulli rank-aware	$r(a_t, n) = \frac{\text{Relevance}(d_{a_t,n})}{\log_2(n+2)}$
Gaussian	$r(a_t, n) = \text{RFF}(d_{a_t,n})$ or $\text{ColBERT}(d_{a_t,n})$
Diversity	$r(a_t, n) = \text{Relevance}(d_{a_t,n}) \cdot \left(1 - \frac{\max_{(i,j)\in\mathcal{O}} \cos(d_{a_t,n}, d_{i,j})+1}{2}\right)$
Diversity concave	$r(a_t, n) = \text{Relevance}(d_{a_t,n}) \cdot \begin{cases} 1, & \text{if } \max_{j<n} \cos(d_{a_t,n}, d_{a_t,j}) < 0 \\ \exp\left(-a \cdot \left(\max_{(i,j)\in\mathcal{O}} \cos(d_{a_t,n}, d_{i,j})\right)^b\right), & \text{otherwise} \end{cases}$
Bernoulli top-k UCB diversity	$r(a_t, n) = \frac{1}{k} \sum_{i=n}^{n+k-1} \text{Relevance}(d_{a_t,i}) \cdot \left(1 - \frac{\max_{(i,j)\in\mathcal{O}} \cos(d_{a_t,n}, d_{i,j}))+1}{2}\right) + c \cdot \sqrt{\frac{\log_2(n+1)}{n}}$, with $c \to 0$

Table 1: Reward policies used for evaluating sub-query selection strategies. For all but the baseline random, we assume $n = \min\{j \in \{1, \ldots, N\} \mid (a_t, j) \notin \mathcal{O}\}$. For the diversity concave policy, we assume $a = 5$ and $b = 15$.

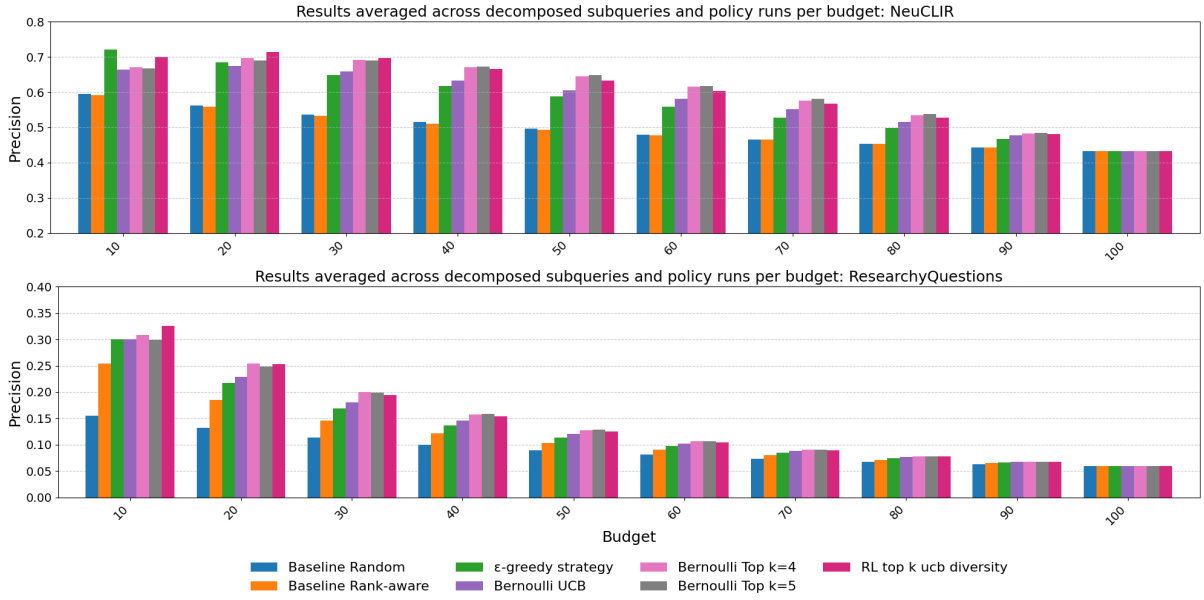

Figure 4: Precision scores on NeuCLIR24 and ResearchyQuestions for baselines and reward policies, evaluated across varying budgets of observed documents. The budgets are expressed as percentages of the total available corpus. Modelling the queries using Bernoulli distribution considering the top ranked k documents performs the best across budgets, with all policies reaching the same performance as the budget covers the whole space of actions.

best when using 10–20% of the budget, likely because early exploitation of high-relevance arms is optimal; and (iii) using rank-information gives a noticeable advantage. All rewards achieve the same macro-precision as the budget reaches maximum value, due to coverage of the same set of documents by all policies and baselines. Figure 10 (in the Appendix) shows similar insights on recall w.r.t. the total amount of relevant documents retrieved for all sub-queries.

Table 3 shows results for multiple values of α-nDCG. We observe that α-nDCG achieves higher values for baselines and all policies for budget $K \in [5, 10, 20]$. As the budget grows, all baselines and policies converge towards the same pool of documents, reducing marginal novelty, as α-nDCG punishes redundant content. Out of all policies, notably the ϵ-greedy, UCB, and ranked

	Budget $b = 20\%$			Budget $b = 30\%$		
	Citation Support	Nugget Coverage	Sentence Support	Citation Support	Nugget Coverage	Sentence Support
Full	0.788	0.461	0.780	0.788	0.461	0.780
Random	0.826 ± 0.043	0.477 ± 0.055	0.822 ± 0.045	0.827 ± 0.054	0.446 ± 0.052	$\mathbf{0.837 \pm 0.055}$
Rank	0.834 ± 0.038	0.462 ± 0.045	0.819 ± 0.045	0.818 ± 0.053	0.462 ± 0.060	0.791 ± 0.067
ϵ-greedy	0.823 ± 0.042	0.490 ± 0.053	0.809 ± 0.049	0.799 ± 0.071	0.478 ± 0.080	0.775 ± 0.077
Bernoulli UCB	0.834 ± 0.031	0.477 ± 0.042	0.823 ± 0.037	0.830 ± 0.051	0.497 ± 0.071	0.798 ± 0.080
Bernoulli $k = 4$	$\mathbf{0.866 \pm 0.037}$	0.463 ± 0.044	$\mathbf{0.855 \pm 0.039}$	0.834 ± 0.060	0.480 ± 0.075	0.817 ± 0.075
Bernoulli $k = 5$	0.849 ± 0.035	$\mathbf{0.492 \pm 0.045}$	0.839 ± 0.038	0.831 ± 0.056	$\mathbf{0.500 \pm 0.064}$	0.819 ± 0.072
Top-K UCB Div.	0.837 ± 0.031	0.463 ± 0.040	0.836 ± 0.030	$\mathbf{0.835 \pm 0.046}$	0.475 ± 0.050	0.826 ± 0.061

Table 2: Report Generation results for budgets $b = 20\%$ and $b = 30\%$ with their confidence intervals. Selecting a relevant subset of documents boosts downstream report generation performance with 6.0–9.9% increase in citation support, 6.7–8.5% in nugget coverage, and 7.3–9.6% in sentence support.

	α-nDCG ↑ NeuCLIR24					α-nDCG ↑ Researchy	
	K=5	K=10	K=20	K=40	K=50	K=5	K=10
Random	0.411	0.454	0.479	0.263	0.311	0.134	0.127
Rank	0.426	0.462	0.476	0.265	0.313	0.172	0.137
ϵ-greedy	**0.500**	**0.539**	0.546	**0.286**	**0.314**	0.207	0.145
Bernoulli	0.445	0.508	0.543	0.246	0.282	0.210	0.157
Bernoulli UCB	0.460	0.517	0.545	0.248	0.279	0.210	**0.164**
Bernoulli $k = 4$	0.461	0.536	0.551	0.275	0.283	**0.240**	0.153
Bernoulli $k = 5$	0.448	0.523	**0.555**	0.267	0.291	0.232	0.156
Bernoulli Rank	0.437	0.481	0.523	0.264	0.289	0.208	0.156
Diversity	0.444	0.505	0.541	0.260	0.281	0.210	0.154
Diversity Conc.	0.442	0.512	0.540	0.260	0.283	0.211	0.149
Top-k UCB Div.	0.469	0.537	0.552	0.253	0.285	0.222	0.154

Table 3: Comparison of α-nDCG over different budgets k for NeuCLIR24 and ResearchyQuestions. Bernoulli posteriors are the best to estimate document relevance regardless the value of k for α-nDCG, with an additional performance boost when considering the top k ranked documents. UCB also brings an advantage. Div. in the last metrics refers to Diversity.

Bernoulli rewards perform the best.

We frame query decomposition and document selection as a stochastic multi-armed bandit where each subquery is an independent arm, and pulling an arm corresponds to observing a document down the ranked list. Each observation receives a Bernoulli reward based on relevance, updating the subquery's relevance distribution. Across experiments, we show that bandit learning outperforms full exploitation and exploration, and that a policy combining rank-information, diversity, and an upper confidence bound (Equation 1) outperforms other variants in Table 1.

5.2 Impact on report generation

We generate reports on NeuCLIR24 using document subsets retrieved by our policies for budgets $b = [20\%; 30\%]$, which provided the best trade-off between evidence coverage and precision in preliminary experiments (Figure 4). Evaluations are conducted with the Auto-ARGUE framework, which measures four aspects of grounded generation: citation relevance (quality of cited sources), citation support (alignment between claims and citations), nugget coverage (proportion of unique information units), and sentence support (share of sentences with appropriate evidence). An example report is provided in Appendix A.3. As shown in Table 2, top-k Bernoulli policies achieve the strongest performance in both nugget coverage and factual support. Citation relevance is omitted, as it remains constant across policies. Results show that selecting documents using multi-armed bandits improves downstream report generation. The strongest performance gains follow the rank-aware top-k Bernoulli policies, including top-k UCB with diversity *citation support* at b=30% (0.835).

5.3 Bandit learning for hierarchical sub-queries

We use ResearchyQuestions to evaluate our correlated bandits formulation. Results in Table 4 show clear improvements over policies for a small budget $b = 10\%$. We pick the three hyperparameters described in Section 3.5 through Bayesian hyperparameter search. The results are presented in Figure 12 in the Appendix, given which we chose a minimal number of observations $n = 4$, informa-

tiveness threshold $\tau = 0.77$, and inheritance factor $\lambda = 0.91$.

Table 4 shows that we can use bandit-learning to model hierarchical query decomposition. For budget $b = 10\%$, there is a clear performance boost when using the hierarchical structure compared to treating all subqueries at the same serial level. Performance using Bernoulli increases by 21.6%, while top-$k = 5$ has a performance boost of 32.3%, and RL top-k UCB diversity a gain of 20.4%. As the budget increases, the gains converge. These results indicate that hierarchy-aware (correlated) bandits best balance exploration/exploitation early by focusing on promising branches of the sub-query tree, yielding higher precision under small budgets.

	$b = 10\%$		$b = 20\%$	
	Ser.	Hier.	Ser.	Hier.
Baseline Random	0.157	0.159	0.137	0.136
Baseline Rank-aware	0.263	0.264	0.191	0.193
ϵ-greedy strategy	0.306	0.306	0.223	0.223
Bernoulli	0.305	**0.371**	0.237	0.246
Bernoulli UCB	0.306	**0.368**	0.234	0.245
Bernoulli Top $k = 3$	0.325	**0.391**	0.260	0.250
Bernoulli Top $k = 4$	0.317	**0.398**	0.261	0.2512
Bernoulli Top $k = 5$	0.303	**0.401**	0.257	0.250
Bernoulli Rank-Aware	0.304	**0.375**	0.232	0.242
Gaussian	0.260	0.264	0.200	0.197
Gaussian ING	0.264	0.263	0.201	0.189
Bernoulli Diversity	0.305	**0.371**	0.237	0.247
Bernoulli Diversity Concave	0.306	**0.371**	0.236	0.247
RL Top-k UCB diversity	0.324	**0.390**	0.256	0.249

Table 4: Comparison of precision under budgets $b = 10\%$ and $b = 20\%$ for both serialized (Ser.) and hierarchical (Hier.) document selection strategies with correlated bandits. For all discrete estimate policies, selecting a subset of documents by taking into account sub-query hierarchies yields better performance, with Bernoulli top $k = 5$ having a 24% performance boost over the serialized representation, where we use the number of observations $n = 4$, informativeness threshold $\tau = 0.77$ and inheritance factor $\lambda = 0.91$

5.4 NeuCLIR analysis

Rank information. We run our experiments in a setting where each sub-query corresponds to a ranked list of documents. Therefore, we make the assumption that the ranked list is a good reflection of document relevance. To verify this hypothesis, we calculate the mean reciprocal rank and fit a linear regression to estimate whether the rank is negatively correlated with the relevance of the documents. Figure 5 shows that very few instances have a negative slope, indicating that the ranked lists are not always consistent with relevance.

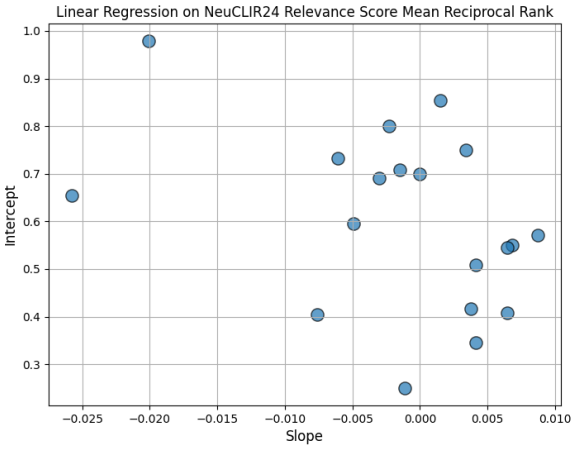

Figure 5: Fitting linear regression functions $y = \alpha + \beta x + \epsilon$ between the mean reciprocal ranks $\bar{r}_i = \frac{1}{K}\sum_{k=1}^{K} r_{i,k}$ for each user request in NeuCLIR24 (represented by a point).

	α-nDCG ↑ NeuCLIR24					α-nDCG ↑ Researchy	
	K=5	K=10	K=20	K=40	K=50	K=5	K=10
Bernoulli	0.445	0.508	0.543	0.246	0.282	0.210	**0.157**
Gaussian	0.413	0.442	0.468	0.252	0.281	0.172	0.140
Gaussian ING	0.409	0.445	0.463	0.259	0.284	0.170	0.135
Bernoulli $k = 3$	**0.462**	0.531	**0.556**	0.264	0.285	**0.230**	0.155
Bernoulli $k = 4$	0.461	**0.536**	0.551	**0.275**	0.283	**0.240**	0.153
Bernoulli $k = 5$	0.448	0.523	**0.555**	**0.267**	**0.291**	**0.232**	**0.156**

Table 5: Comparison of α-nDCG for NeuCLIR24 and ResearchyQuestions. Gaussian vs. Bernoulli modelling.

Top-k. We explore different values of the top k documents in a ranked list. More precisely, we investigate performance for values of $k \in [3, 4, 5]$. We observed that any Bernoulli taking into account top-k documents outperforms all the other rewards across budget, and therefore we report only top-k for $k = 3$ in the main tables. In Table 5 we report the values for all experiments as reference. For analysis on Gaussian distributions, check A.2.

6 Conclusions

This study proposes reframing query decomposition and document retrieval for complex query answering as an exploitation–exploration problem under a multi-armed bandit setting. We conduct extensive empirical studies across a range of policies and discover that, when having access to binary

relevance labels, Bernoulli distributions with rank information perform best for selecting relevant content under a constrained budget. These results are demonstrated on two datasets, NeuCLIR24 and ResearchyQuestions. We further show that modeling correlations between sub-queries in a hierarchical setting consistently boosts performance across all rewards. We also stress that such an approach performs robustly, while the choice of budget is data-dependent w.r.t. the real distribution of relevance across retrieved documents.

We propose further analyses into correlated and contextualized bandits where correlations need not be explicit in the data but may instead emerge naturally in the representational space of sub-queries. Another direction would be to train a retrieval model optimized for relevance, document diversity, exploration, or long-form generation within a policy-learning framework.

Future Work. We believe that our approach can be used in an online setting by applying other signals such as clicks or LLM judgments calculated on the fly. Furthermore, it can also be applied to more cost-efficient online evaluation pipelines such as (Ju et al., 2026).

Limitations

Our work proposes a reframing of an existing paradigm in a popular experimental setup. The study comes with a couple of challenges and limitations: when a retrieval module is involved, the rewards directly depend on the performance of the retriever; moreover, the performance over budgets directly depends on the real distribution of observations. In a dynamic setting, often a set of subqueries is not pre-defined, such as in the NeuCLIR dataset, which introduces noise by regenerating sub-queries every time. In addition, often the subqueries are not only not generated, but they are samples from an unbounded search space for a given user request.

Acknowledgments

This research was (partially) supported by the Dutch Research Council (NWO), under project numbers 024.004.022, NWA.1389.20.183, and KICH3.LTP.20.006, and the European Union under grant agreements No. 101070212 (FINDHR) and No. 101201510 (UNITE).

Views and opinions expressed are those of the authors only and do not necessarily reflect those of their respective employers, funders and/or granting authorities.

References

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Arian Askari, Roxana Petcu, Chuan Meng, Mohammad Aliannejadi, Amin Abolghasemi, Evangelos Kanoulas, and Suzan Verberne. 2025. Self-seeding and multi-intent self-instructing llms for generating intent-aware information-seeking dialogs. *Preprint*, arXiv:2402.11633.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, and 29 others. 2023. Qwen technical report. *Preprint*, arXiv:2309.16609.

Samarth Bhargav, Georgios Sidiropoulos, and Evangelos Kanoulas. 2022. 'It's on the tip of my tongue': A new dataset for known-item retrieval. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21 - 25, 2022*, pages 48–56. ACM.

Christian Buck, Jannis Bulian, Massimiliano Ciaramita, Wojciech Gajewski, Andrea Gesmundo, Neil Houlsby, and Wei Wang. 2018. Ask the right questions: Active question reformulation with reinforcement learning. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Charles L.A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, page 659–666, New York, NY, USA. Association for Computing Machinery.

Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric P. Xing, and Zhiting Hu. 2022. Rlprompt: Optimizing discrete text prompts with reinforcement learning. *Preprint*, arXiv:2205.12548.

Zhanzhong Gu, Wenjing Jia, Massimo Piccardi, and Ping Yu. 2025. Empowering large language models for automated clinical assessment with generation-augmented retrieval and hierarchical chain-of-thought. *Artif. Intell. Medicine*, 162:103078.

Bowen Jin, Jinsung Yoon, Jiawei Han, and Sercan O. Arik. 2024. Long-context llms meet rag: Overcoming challenges for long inputs in rag. *Preprint*, arXiv:2410.05983.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *Preprint*, arXiv:2503.09516.

Jia-Huei Ju, Suzan Verberne, Maarten de Rijke, and Andrew Yates. 2026. Controlled retrieval-augmented context evaluation for long-form rag. *Preprint*, arXiv:2506.20051.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2023. Decomposed prompting: A modular approach for solving complex tasks. *Preprint*, arXiv:2210.02406.

Takyoung Kim, Kyungjae Lee, Young Rok Jang, Ji Yong Cho, Gangwoo Kim, Minseok Cho, and Moontae Lee. 2025. Learning to explore and select for coverage-conditioned retrieval-augmented generation. *Preprint*, arXiv:2407.01158.

Antonios Minas Krasakis, Andrew Yates, and Evangelos Kanoulas. 2025. Constructing set-compositional and negated representations for first-stage ranking. *Preprint*, arXiv:2501.07679.

Dawn Lawrie, Sean MacAvaney, James Mayfield, Paul McNamee, Douglas W. Oard, Luca Soldaini, and Eugene Yang. 2025. Overview of the trec 2024 neuclir track. *Preprint*, arXiv:2509.14355.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Preprint*, arXiv:2005.11401.

Dan Li and Evangelos Kanoulas. 2017. Active sampling for large-scale information retrieval evaluation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 49–58. ACM.

Suraj Nair, Eugene Yang, Dawn Lawrie, Kevin Duh, Paul McNamee, Kenton Murray, James Mayfield, and Douglas W. Oard. 2022. Transfer learning approaches for building cross-language dense retrieval models. In *Proceedings of the 44th European Conference on Information Retrieval (ECIR)*.

Thong Nguyen, Sean MacAvaney, and Andrew Yates. 2023. A unified framework for learned sparse retrieval. *Preprint*, arXiv:2303.13416.

Daan Odijk, Edgar Meij, Isaac Sijaranamual, and Maarten de Rijke. 2015. Dynamic query modeling for related content finding. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, August 9-13, 2015*, pages 33–42. ACM.

Roxana Petcu, Samarth Bhargav, Maarten de Rijke, and Evangelos Kanoulas. 2025. A comprehensive taxonomy of negation for nlp and neural retrievers. *Preprint*, arXiv:2507.22337.

Roxana Petcu and Subhadeep Maji. 2024. Efficient data selection employing semantic similarity-based graph structures for model training. *Preprint*, arXiv:2402.14888.

Jakub Podolak, Leon Perić, Mina Janićijević, and Roxana Petcu. 2025. Beyond reproducibility: Advancing zero-shot llm reranking efficiency with setwise insertion. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '25, page 3205–3213, New York, NY, USA. Association for Computing Machinery.

Peng Qi, Xiaowen Lin, Leo Mehr, Zijian Wang, and Christopher D. Manning. 2019. Answering complex open-domain questions through iterative query generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2590–2602. Association for Computational Linguistics.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Preprint*, arXiv:2305.18290.

Md Mustafizur Rahman, Mucahid Kutlu, Tamer Elsayed, and Matthew Lease. 2020. Efficient test collection construction via active learning. In *Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval*, ICTIR '20, page 177–184. ACM.

Corby Rosset, Ho-Lam Chung, Guanghui Qin, Ethan C. Chau, Zhuo Feng, Ahmed Awadallah, Jennifer Neville, and Nikhil Rao. 2024. Researchy questions: A dataset of multi-perspective, decompositional questions for llm web agents. *Preprint*, arXiv:2402.17896.

Rulin Shao, Rui Qiao, Varsha Kishore, Niklas Muennighoff, Xi Victoria Lin, Daniela Rus, Bryan Kian Hsiang Low, Sewon Min, Wen tau Yih, Pang Wei Koh, and Luke Zettlemoyer. 2025. Reasonir: Training retrievers for reasoning tasks. *Preprint*, arXiv:2504.20595.

Heydar Soudani, Roxana Petcu, Evangelos Kanoulas, and Faegheh Hasibi. 2024. A survey on recent advances in conversational data generation. *Preprint*, arXiv:2405.13003.

Coen van den Elsen, Francien Barkhof, Thijmen Nijdam, Simon Lupart, and Mohammad Aliannejadi. 2025. Reproducing nevir: Negation in neural information retrieval. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2025, Padua, Italy, July 13-18, 2025*, pages 3346–3356. ACM.

Ellen M. Voorhees. 2018. On building fair and reusable test collections using bandit techniques. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 407–416, New York, NY, USA. Association for Computing Machinery.

William Walden, Marc Mason, Orion Weller, Laura Dietz, Hannah Recknor, Bryan Li, Gabrielle Kaili-May Liu, Yu Hou, James Mayfield, and Eugene Yang. 2025. Auto-argue: Llm-based report generation evaluation. *Preprint*, arXiv:2509.26184.

Orion Weller, Dawn J. Lawrie, and Benjamin Van Durme. 2024. Nevir: Negation in neural information retrieval. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2024 - Volume 1: Long Papers, St. Julian's, Malta, March 17-22, 2024*, pages 2274–2287. Association for Computational Linguistics.

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, and 23 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Eugene Yang, Dawn Lawrie, James Mayfield, Douglas W. Oard, and Scott Miller. 2024. Translate-distill: Learning cross-language dense retrieval by translation and distillation. In *Proceedings of the 46th European Conference on Information Retrieval (ECIR)*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Wenhao Zhang, Mengqi Zhang, Shiguang Wu, Jiahuan Pei, Zhaochun Ren, Maarten de Rijke, Zhumin Chen, and Pengjie Ren. 2025. Excluir: Exclusionary neural information retrieval. In *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 13295–13303. AAAI Press.

Zongmeng Zhang, Jinhua Zhu, Wen gang Zhou, Xiang Qi, Peng Zhang, and Houqiang Li. 2024. BoolQuestions: Does dense retrieval understand boolean logic in language? In *Conference on Empirical Methods in Natural Language Processing*.

A Appendix

A.1 Prompts for Query Decomposition

The NeuCLIR dataset contains complex user requests and associated nuggets with annotated documents. We thus generate our own query decomposition using LLM calls. We decompose each NeuCLIR user request into $k = 16$ sub-queries using prompts 6 and 7.

A.2 Gaussian Distributions

Gaussian estimates. The finding in Figure 5 serves as an explanation to the Gaussian policies performing considerably poorly compared to the Bernoulli ones, as seen Table 5. This suggests that Gaussian distributions may not be suited to capturing relevance when the ranking signal is noisy.

A.3 Report Generation Example

In this section, we show an example of a report generated and evaluated using Auto-ARGUE.

Query. What characterizes Ernest Hemingway's writing style, and what is the "iceberg theory"?

Generated report. Ernest Hemingway was a renowned American writer known for a distinctive style built on simplicity, clarity, and concrete detail [5]. He is often associated with the "iceberg theory," the idea that most meaning lies beneath the surface while only a small portion is stated explicitly [3], [5]. Hemingway's years as a journalist shaped this approach: he learned to write concise, unadorned prose that focuses on essentials [2]. He also insisted on authenticity—"never write about what you do not know"—and favored describing observable actions while letting readers infer the emotions behind them [1]. This restrained, lucid style has been widely emulated and is a major reason he is regarded as one of the most influential writers of the twentieth century [4].

Algorithm 2: Thompson Sampling in Continuous Space

Input : Sub-queries $\mathcal{S} = \{s_1, \ldots, s_K\}$
Input : Budget b

for $i = 1, \ldots, K$ **do**
  Define Gaussian priors $\mathcal{N}(\mu_0, \sigma_0^2)$ where
    $\mu_0 \leftarrow 0$ and $\sigma_0^2 \leftarrow 1$
for $t = 1, \ldots, b$ **do**
  for $i = 1, \ldots, K$ **do**
    Sample from posterior $\theta_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$
  Select arm (sub-query): $a_t \leftarrow \arg\max_i \theta_i$
  Pick next document $d_{a_t,n} \in \mathcal{D}_{a_t}$ where
    $n = \min\{j \in \{1, \ldots, N\} \mid (a_t, j) \notin \mathcal{O}\}$
  Observe the likelihood: $r(a_t, n)$
  Update posterior:
    $\sigma_{a_t}^2 = \sqrt{(\frac{1}{\sigma_0^2} + n)^{-1}}$
    $\mu_{a_t} = \sigma_{a_t}^2 \cdot \sum_{r \in S_{a_t}} r$

A.4 Complete results on NeuCLIR and ResearchyQuestions

Figures 8–11 report precision and recall across budgets for all baselines and bandit policies we introduce in Table 1. Figure 13 compared serial and hierarchy-aware document selection.

Prompt for generating Problem Definition Search Queries
You are an assistant that receives a complex user request and must generate search queries whose answers: (1) define the user's request, and (2) define all key entities mentioned in the request.

1. Consider any current events, recent developments, or specific details mentioned in the context to enhance the queries (if context is provided).
2. Write **up to «no_queries»** objective Google search queries related to the request.
   The queries should be general and high-level (e.g., encyclopedic or definitional in nature).
   Avoid narrow or overly specific phrasings. Queries should be phrased as something someone would realistically search for.
3. Assume the current date is «current_date» if required by the request.
4. Return the queries as a JSON list of strings in the format:
   `["query 1", "query 2", "query 3"]`.
   The response must contain **only** this list.

Figure 6: Prompt for Generating Problem Definition Queries

Prompt for Generating Specialized Information Search Queries
You are a seasoned research assistant tasked with generating search queries that ask highly specialized and detailed information about a user request. You have access to a context formed of multiple supporting documents.

1. Use the context (if provided) to identify aspects of the request that would benefit from more specific, in-depth, or expert-level information.
   Consider any current events, real-time developments, or precise facts that can be used to craft better questions.
2. Write **up to «no_queries»** objective Google search queries related to the request.
   These queries should be highly specific and targeted—something a person wouldn't know from general knowledge.
   Each query should reflect a concrete sub-aspect or follow-up inquiry about the original request.
3. Assume the current date is «current_date» if required by the request.
4. Return the queries as a JSON list of strings in the format:
   `["query 1", "query 2", "query 3"]`.
   The response must contain **only** this list.

Figure 7: Prompt for Generating Specialized Information Queries

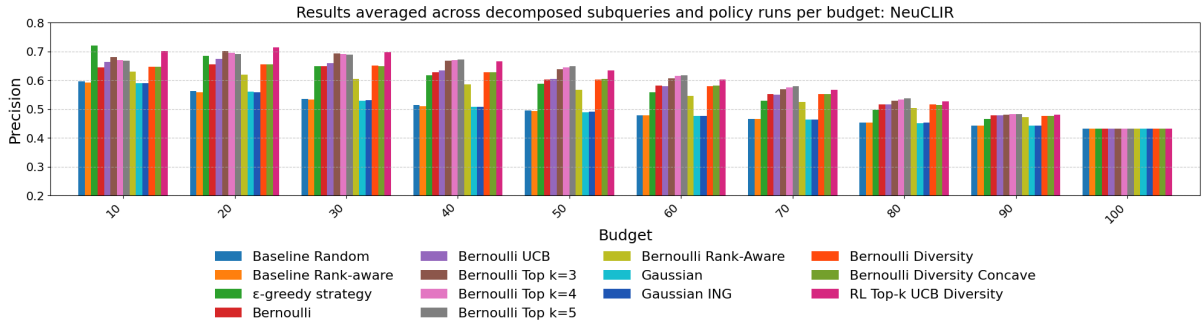

Figure 8: Precision applying baselines and reward policies for query selection for different budgets over observed documents. The budget is calculated as a percentage over the total amount of available documents. Modelling the queries using Bernoulli distribution considering the top ranked k documents performs the best across budgets, with all policies reaching the same performance as the budget covers the whole space of actions.

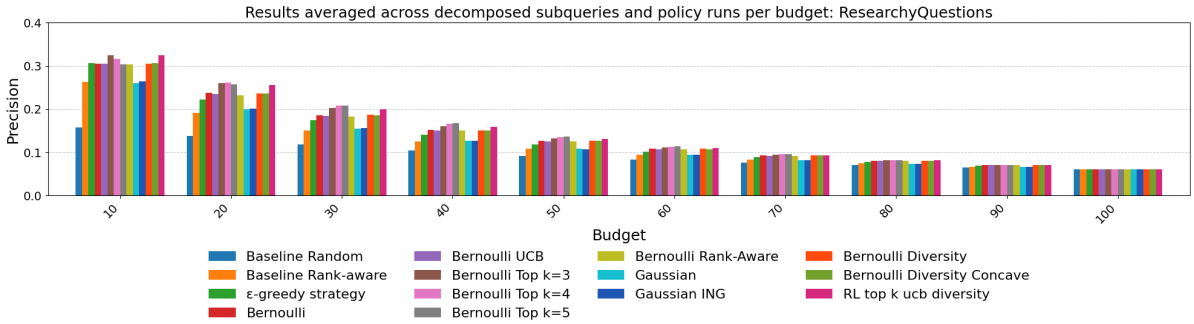

Figure 9: Precision applying baselines and reward policies for query selection for different budgets over observed documents in ResearchyQuestions. The budget is calculated as a percentage over the total amount of available documents. Modeling the queries using Bernoulli distribution considering the top ranked k documents performs the best across budgets, with all policies reaching the same performance as the budget covers the whole space of actions.

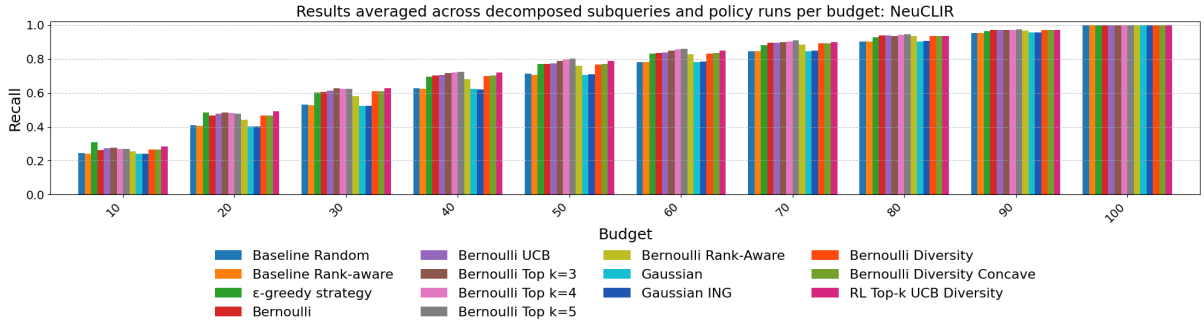

Figure 10: Recall applying baselines and reward policies for query selection for different budgets over observed documents. The budget is calculated as a percentage over the total amount of available documents. Modelling the queries using Bernoulli distribution considering the top ranked k documents performs the best across budgets, with all policies reaching the same performance as the budget covers the whole space of actions.

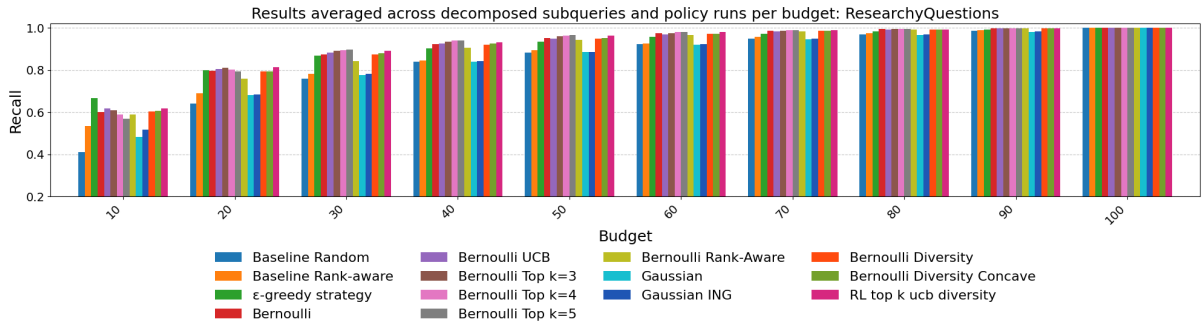

Figure 11: Recall applying baselines and reward policies for query selection for different budgets over observed documents in ResearchyQuestions. The budget is calculated as a percentage over the total amount of available documents. Modeling the queries using Bernoulli distribution considering the top ranked k documents performs the best across budgets, with all policies reaching the same performance as the budget covers the whole space of actions.

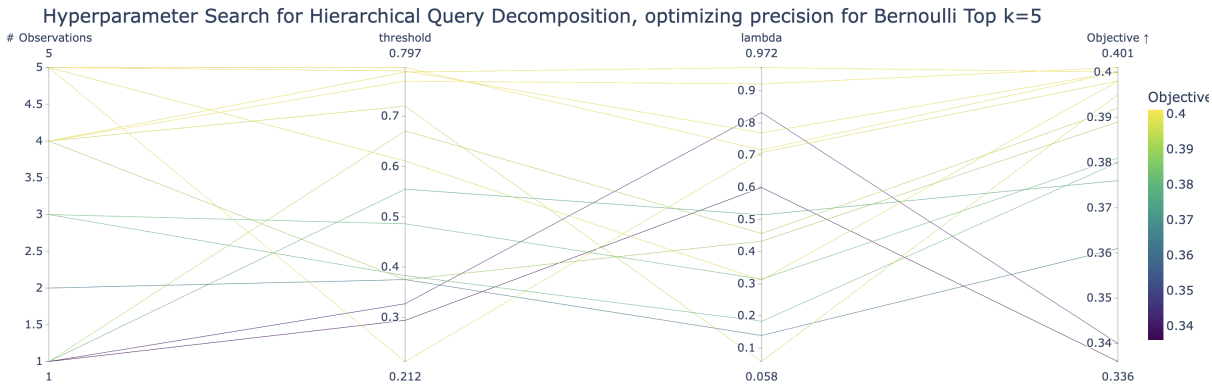

Figure 12: Visualization of hyperparameter search space with Bayesian optimization on the precision objective of the RL top k = 5 approach on the Researchy dataset. Each line corresponds to a trial with specific values of the three hyperparameters, colored according to the achieved objective.

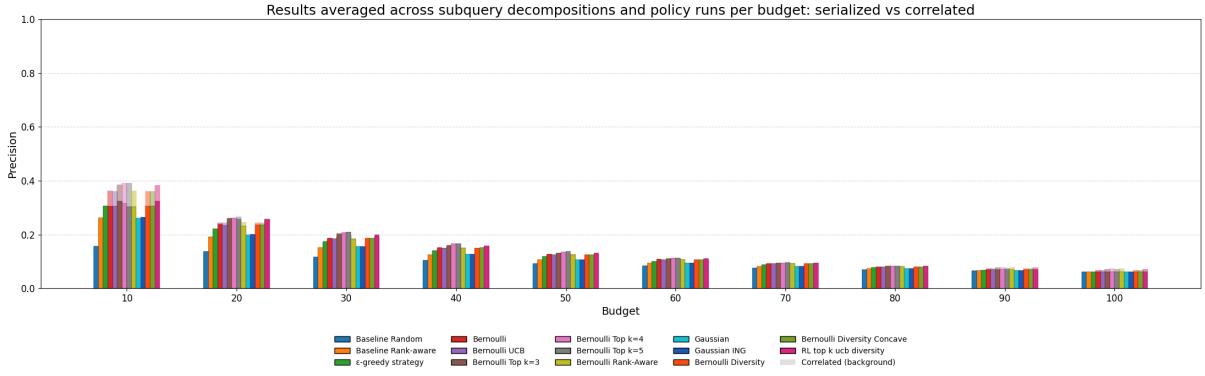

Figure 13: Precision applying baselines and reward policies for query selection for different budgets over observed documents on the ResearchyQuestions dataset: serial vs correlated.