

White Rose Research Online

eprints@whiterose.ac.uk
https://eprints.whiterose.ac.uk
Universities of Leeds, Sheffield and York

Deposited via The University of Sheffield.

White Rose Research Online URL for this paper:
https://eprints.whiterose.ac.uk/id/eprint/243699/

Version: Published Version

Proceedings Paper:
Gabín, J., Parapar, J. and Wang, X. (2026) Beyond Top-e: simulation-based interactive evaluation for query suggestions. In: SIGIR '26: Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval. 49th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2026), 20-24 Jul 2026, Melbourne, Australia. ACM, pp. 3167-3174.

https://doi.org/10.1145/3805712.3808618

Reuse
This article is distributed under the terms of the Creative Commons Attribution (CC BY) licence. This licence allows you to distribute, remix, tweak, and build upon the work, even commercially, as long as you credit the authors for the original work. More information and the full terms of the licence here:
https://creativecommons.org/licenses/

Takedown
If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.





Beyond Top-e: Simulation-Based Interactive Evaluation for Query Suggestions

Jorge Gabín
jorge.gabin@udc.es
IRLab, CITIC, Universidade da Coruña
A Coruña, Spain
Linknovate Science
Santiago de Compostela, Spain

Javier Parapar
javier.parapar@udc.es
IRLab, CITIC, Universidade da Coruña
A Coruña, Spain

Xi Wang
xi.wang@sheffield.ac.uk
University of Sheffield
Sheffield, United Kingdom

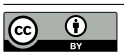
Abstract

Evaluating query suggestion systems in a manner that reflects real-world query formulation remains a persistent challenge. Most offline methodologies adopt static assumptions, such as users accepting all or the top-e suggestions, ignoring the inherently selective and intent-driven nature of interactive search. While online experiments provide realistic behavioural signals, they are costly, difficult to scale, and often irreproducible. To bridge this gap, we introduce SIQSE (Simulation-based Interactive Query Suggestion Evaluation), a framework that models query reformulation as an interactive selection task performed by a simulated user. In SIQSE, a Large Language Model (LLM) acts as a surrogate user that progressively selects suggestions according to contextual relevance and explicit search intent. Unlike static offline protocols, this simulation captures the iterative and selective dynamics of real query formulation. Our contributions are twofold. First, we develop and validate an LLM-based selection model, systematically analysing how varying levels of intent information and selection strategies affect its ability to approximate human selection behaviour. Second, we employ this selector to benchmark multiple query suggestion systems across diverse datasets under interactive conditions. Importantly, while the selector is LLM-based, the final evaluation is computed exclusively through ranking-based effectiveness metrics over the rankings produced by selected expansions, ensuring that system performance reflects retrieval quality rather than alignment with the surrogate user model. By modelling round-based interaction while maintaining metric independence, SIQSE offers a scalable, reproducible evaluation paradigm that brings offline assessment closer to the complexity of real-world search behaviour. To facilitate adoption and reproducibility, we release SIQSE as an open-source Python library[1].

CCS Concepts

• **Information systems** → **Users and interactive retrieval**; **Information retrieval query processing**; • **Computing methodologies** → *Natural language processing*.

[1] https://github.com/IRLab-UDC/SIQSE



This work is licensed under a Creative Commons Attribution 4.0 International License.
SIGIR '26, Melbourne, VIC, Australia
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2599-9/2026/07
https://doi.org/10.1145/3805712.3808618

Keywords

Interactive Information Retrieval, Interactive Query Suggestion Evaluation, Simulation-based Evaluation, Offline Evaluation

ACM Reference Format:
Jorge Gabín, Javier Parapar, and Xi Wang. 2026. Beyond Top-e: Simulation-Based Interactive Evaluation for Query Suggestions. 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, 8 pages. https://doi.org/10.1145/3805712.3808618

1 Introduction

Imagine a user sitting before a search bar, grappling with a complex information need. They do not possess a perfectly articulated query; instead, they carry, what Belkin famously termed, an *Anomalous State of Knowledge* [2], a gap in understanding that they cannot yet fully describe. In this light, Information Retrieval (IR) is not a simple retrieval of documents, but a recursive, cognitive process. Since the pioneering *TREC Interactive tracks* [16, 25, 26], our field has recognised that a search session is an unfolding narrative where the user and system participate in a continuous feedback loop.

Query and keyphrase suggestions have emerged as the primary vehicle for this dialogue in search engines [10]. They are not merely "auto-complete" shortcuts; they are cognitive anchors that estimate users' search goals, help users navigate the search space and refine their latent intent. However, a significant rift has formed between the interactive reality of these tools and the methods we use to evaluate them [18]. While the IIR (Interactive IR) community has long championed user-centric studies [5], the vast majority of query suggestion research remains tethered to the static, offline "Cranfield-style" paradigm [8, 29]. This traditional framework evaluates system performance in a highly controlled, laboratory-like setting by relying on a fixed document corpus, standardized query sets, and static relevance judgments. Consequently, while it offers excellent algorithmic reproducibility and benchmarking, it fundamentally abstracts away the dynamic, user- and context-dependent reality of actual human search behaviour.

The most pervasive symptom of this rift in query suggestion evaluation is the *top-e* assumption. This assessment model assumes a user behaves as a passive recipient, either accepting a fixed number of suggestions or none at all. This "convenient fiction" ignores the quintessence of human search: *selectivity*. However, a real user does not just see a list of search results; they scan, reject, pivot, and iterate. While online experiments (e.g. A/B testing [19]) can capture this nuance, their prohibitive cost and lack of reproducibility

make them an impractical "gold standard" for the rapid, iterative development cycles required in modern IR.

In this paper, we propose to bridge this gap by moving *beyond top-e*. We introduce *SIQSE* (**S**imulation-based **I**nteractive **Q**uery **S**uggestion **E**valuation), a framework designed to bring the fidelity of the interactive search process into the scalable, reproducible world of offline benchmarking. SIQSE operates on the premise that if we cannot always put a human in the loop, high-fidelity user simulations can effectively bridge the gap. By employing Large Language Models (LLMs) as backbone to develop surrogate users, SIQSE transforms evaluation from a static calculation into a round-based simulation over search turns. The LLM acts as a selective agent, evaluating system-generated suggestions and deciding which terms to incorporate to satisfy the underlying search intent.

As a resource paper, our contributions are grounded in a comprehensive validation of this simulation-driven approach:

- *Engineering the surrogate user:* We conduct a rigorous validation of various LLM architectures to determine their capacity for human-aligned selectivity. As detailed in Table 1, we measure how these "selectors" perform across several benchmarks, including CODEC [22], TREC Fair 2021 [12], TREC 7 [31] and TREC 8 [30]. We specifically analyse the impact of intent information by comparing model behaviour across three settings: brief topic descriptions, detailed narratives, and no intent access. We find that the inclusion of detailed narratives allows models to better discriminate between the proposed suggestions. Separately, we observe that an *iterative* interaction mode—as opposed to single-batch selection—further improves performance by allowing models to align more closely with human-like relevance patterns.
- *Re-evaluating the suggestion landscape:* We use SIQSE to benchmark a broad spectrum of keyphrase suggestion methods, ranging from classical embedding-based models to state-of-the-art RAG-enhanced LLMs. The results, presented in Table 2, provide a comparison between traditional static top-e selection and our interactive simulation. We observe that certain systems that appear mediocre under a fixed $e = 5$ ranking actually shine when a selective user is allowed to pick the most useful terms, leading to higher downstream retrieval performance in terms of MAP and Precision.
- *A tangible resource for the community:* To ensure our findings are actionable and to support the shift toward more authentic performance metrics, we have made our evaluation suite publicly available through SIQSE. As a modular Python library, SIQSE provides the necessary infrastructure for the community to implement and iterate upon realistic evaluation protocols in future studies.

By bridging the gap between fast offline testing and complex user interaction, SIQSE provides a more accurate way to evaluate suggestion models. Our findings show that traditional static top-e metrics are often misleading because they force a fixed selection that ignores how users actually filter or reject terms. We encourage the community to move away from these rigid baselines and toward interactive simulations. This shift is essential to ensure that evaluation reflects real-world usage, where a system's value is determined by its ability to provide useful choices in a dynamic dialogue, rather than its performance on a fixed, pre-determined list of candidates.

2 Related Work

The development of SIQSE sits at the intersection of three major research areas: the theory of Interactive Information Retrieval, the evolution of query suggestion evaluation, and the emerging use of Large Language Models as user simulators.

2.1 Interactive Information Retrieval

Traditional IR evaluation has long been dominated by the *Cranfield paradigm* [7], which treats retrieval as a static matching problem between a fixed query and a document corpus. However, the IIR community has consistently argued that this overlooks the cognitive and behavioural dimensions of search. Central to this is the concept of *dynamic information needs*, where users refine their understanding as they interact with system feedback [5].

Early attempts to bridge the gap between static and interactive evaluation led to the *simulated work task* framework [4], which encouraged evaluating systems within a specific user context. Despite these theoretical advances, the complexity of conducting large-scale user studies has meant that most IR benchmarks (e.g., TREC Deep Learning Tracks [9] and MS MARCO [24]) still rely on static snapshots. Our work aims to operationalise these IIR principles by providing a scalable, automated way to simulate the very interaction loops that Borlund and others identified as critical [5]. This theoretical need for interactivity is most evident in how modern systems attempt to guide users through suggestions.

2.2 Query Suggestion

Query suggestion systems have evolved from simple dictionary-based completions [11] to sophisticated models leveraging search logs, click-through data, and, more recently, large-scale embeddings and LLMs [6, 14, 15]. While the *generation* of suggestions has seen massive leaps in performance, the *evaluation* methods have remained relatively stagnant.

Most studies evaluate suggestions adopting the *top-e* assumption mentioned earlier. This presupposes that the suggestions' value is independent of the user's choice to ignore or selectively pick terms. While some work has looked at "session-based" evaluation [17], these approaches often still treat the user as a predictable actor. SIQSE challenges this by introducing *selectivity*, allowing a simulated user to only accept suggestions that move them closer to their specific narrative goal. However, modelling such selectivity requires a level of semantic reasoning that traditional probabilistic models could not provide.

2.3 LLMs as Simulated Users

The idea of simulating users to evaluate search systems is not new. Early simulators used rule-based models or probabilistic "user models" to mimic click behaviour or query reformulation [23]. However, these models often lacked the semantic depth required to understand *why* a user might prefer one term over another.

The advent of LLMs has revolutionised this space, providing the "reasoning engine" necessary for realistic simulation. Recent research has demonstrated that LLMs can act as "surrogate users"

or "LLM-as-a-judge", showing high correlation with human relevance judgments in various tasks [13, 28]. In the context of search simulation, LLMs have been used to generate synthetic data [3] or simulate multi-turn conversations in RAG settings [27]. SIQSE builds upon this by moving beyond simple judgment; it employs the LLM as an *active agent* in a feedback loop, specifically focusing on the iterative selection of suggestions, a nuance that is often missing from existing LLM-based evaluation frameworks.

Despite the maturity of IIR theory and the rapid advancement of LLM-based agents, there remains a critical gap in the available research infrastructure: the field lacks a unified, open-source framework that models the *iterative selectivity* of users when interacting with query suggestions. SIQSE addresses this by providing a reproducible environment where suggestions are evaluated not as static rankings, but as dynamic instruments of search navigation.

3 Simulation-based Interactive Query Suggestion Evaluation

The SIQSE method moves beyond static, list-based query suggestions by modelling it as a dynamic, interaction-driven process. This section defines the formal logic of the simulation and the mechanism by which the surrogate user selects suggestions.

3.1 Formalizing Interactive Selection

We define a search session as a sequence of discrete time steps $t \in \{1, \ldots, T\}$. In traditional offline evaluation, a system typically assesses the top-e suggestions as a static list. In SIQSE, with the availability of the surrogate user, we instead model the *transition function* of a user navigating through a suggestion space. The state of a search session at step t is defined by the tuple:

$$\mathcal{Z}_t = \langle \mathcal{I}, q_t, \mathcal{R}(q_t), \mathcal{D}_t \rangle \tag{1}$$

where $\mathcal{I}$ represents the *search intent*, encompassing the user's underlying information need; q_t is the *current query* issued at time t; $\mathcal{R}(q_t)$ is the *ranked list* of documents returned by the search engine for q_t; and $\mathcal{D}_t$ is the *feedback context*, which provides the surrogate user with the content (defined here as the document text) and relevance labels of the top-ranked documents in $\mathcal{R}(q_t)$.

When the suggestion engine $\mathcal{S}$ provides a set of candidate queries $C_t = \{c_1, c_2, \ldots, c_k\}$, the surrogate user $\mathcal{U}$ executes a selection function:

$$f_{\mathcal{U}}(\mathcal{Z}_t, C_t) \rightarrow q_{t+1} \tag{2}$$

Here, q_{t+1} represents the query for the subsequent round. This formulation allows for two realistic user behaviours: selection (choosing a candidate c_i), and continuation (retaining q_t if the provided suggestions lack utility).

3.2 The Simulation Workflow

The SIQSE simulation follows a structured three-phase loop designed to mimic real-world search sessions:

Phase 1: User initiation and intent grounding. Every session is initiated by a mandatory initial query q_0, which serves as the concrete starting point for the search. This query may be optionally grounded in a specific, explicit intent $\mathcal{I}$ (representing a detailed internal goal). If provided, this intent acts as a static "plug-in" component that remains constant throughout the session to guide the agent's choices. If the intent is absent, it never appears; in such cases, the agent relies entirely on the evolving query history to drive the simulation.

Phase 2: Informed feedback and selection. After q_t is issued, the agent observes the top-retrieved results to assess the current "information scent". The agent then evaluates the available suggestions in C_t against the session goal. If an intent $\mathcal{I}$ was provided in Phase 1, it serves as the fixed benchmark for this evaluation. By comparing the retrieved documents and the candidate suggestions against this constant intent (or, in the absence of a defined intent, the initial query), the agent identifies which term most effectively fills the remaining information gap.

Phase 3: State update and query expansion. The session state $\mathcal{Z}$ is updated to include the history of selected terms. The subsequent query q_{t+1} is formed by appending the accepted suggestion to the current query q_t. This additive process reflects how users incrementally refine their search by incorporating suggested keyphrases into their existing query string. This loop repeats until the information need is satisfied, the agent determines no further suggestions are useful (in the *e=auto* setting), or the maximum number of turns T is reached.

3.3 Modelling the Surrogate User

The surrogate user in SIQSE is not a simple ranker; it is a decision agent. By utilising a LLM, the framework replicates two critical aspects of human search behaviour:

- **Contextual Relevance:** The capacity to critically evaluate whether a suggested query is semantically aligned with the user's underlying latent intent, effectively bridging the remaining information gap. Unlike traditional keyword-based models, the LLM-driven agent recognizes the utility of a suggestion by identifying conceptual links and logical progressions toward the goal, even in the complete absence of explicit lexical overlap or shared keyphrases with the current query.
- **Information Foraging and Termination:** Guided by the intent $\mathcal{I}$ and the feedback context $\mathcal{D}_t$, the agent dynamically determines when to conclude the search process. It can elect to stop entirely if the latent information need has been satisfied, if the current path is identified as a dead end, or if the marginal utility of additional suggestions no longer justifies continued interaction.

3.4 Practical Integration

The SIQSE framework is implemented as a modular Python package. While it can be used as a standalone tool for simulating user sessions, it is also designed for seamless integration into existing retrieval ecosystems like `PyTerrier` [21]. By decoupling the *selector* from the *retriever*, the framework acts as a "smart transformer" that can be plugged into any standard IR pipeline. This modularity allows researchers to swap out different selection agents or retrieval models without altering the core simulation logic.

PyTerrier Integration. To demonstrate the framework's interoperability, the following snippet illustrates a SIQSE-enhanced retrieval pipeline, where the library functions as a modular component for generating dynamic, intent-aware query suggestions:

```python
import pyterrier as pt
from siqse import LLMSuggestionSelector

index = pt.IndexFactory.of("./index_path")
bm25 = pt.BatchRetrieve(index, wmodel="BM25")
sel = LLMSuggestionSelector(
    model="llama3.1:8b",
    selection_mode="iterative",
    use_feedback_docs=True
)

def siqse_expand(df):
    def get_feedback_docs(query, suggestions):
        [...]

    df["query"] = df.apply(
        lambda r: " ".join([r["query"]] +
        sel.select_suggestions(
            query=r["query"],
            suggestions=r["suggestions"],
            user_intent=r.get("intent"),
            n=None, # Automatic stopping
            get_docs=get_feedback_docs
        ),
        axis=1
    )

    return df

queries_df = pd.DataFrame([{
    "qid": "1", "query": "climate change",
    "suggestions": ["global warming", "emissions", ...],
    "intent": "Climate change on polar regions"
}])

pipeline = pt.apply.generic(siqse_expand) >> bm25
results = pipeline.transform(queries_df)
```

Architectural Versatility. The framework's plug-and-play nature is facilitated by an OpenAI-compatible API, allowing it to interface with local LLM providers (e.g., Ollama, vLLM [20]) or cloud endpoints. This setup ensures that SIQSE is not tied to a specific LLM model. This modularity allows researchers to transition from static evaluation to interactive simulation with minimal code overhead, effectively "protecting" retrieval pipelines from expansion noise through LLM-guided selection criteria.

4 Experiments

To evaluate the validity of the SIQSE framework, we address three core research questions. The analysis proceeds hierarchically: first establishing the surrogate's fidelity to human choices, then examining the role of intent in shaping those choices, and finally evaluating the downstream search utility of the dynamic selections.

RQ1: Surrogate fidelity (Human-likeness): Can LLM-based decision agents accurately replicate human relevance judgments when selecting query suggestions?

RQ2: Role of user intent: How does providing the surrogate with explicit narrative context (user intent) influence its selection accuracy and subsequent retrieval outcomes?

RQ3: Dynamic utility: Does the dynamic, simulation-based selection of suggestions yield different retrieval performance compared to static, list-based ranking?

4.1 Experimental Setup

To evaluate SIQSE, we utilise a diverse set of test collections and simulate users with varying levels of intent specificity.

Datasets. We conduct experiments on four standard benchmarks. TREC Fair 2021, which focuses on matching WikiProjects to article drafts, and CODEC, a document entity collection for domain-specific search, both provide manual annotations that we utilise as ground-truth suggestions. For these datasets, we model the search intent using the *scope* provided in TREC Fair and the *guidelines* provided in CODEC. In contrast, while TREC 7 and TREC 8 (Ad Hoc Track) provide detailed *descriptions* and *narratives* to model the user's intent, they do not include manual query suggestions out-of-the-box. To bridge this gap, we utilize query variations from the UQV100 dataset [1] as the ground-truth manual suggestions. This setup ensures a consistent expert-curated benchmark across all four collections.

Surrogate User Models. We employ a range of Large Language Models to act as the surrogate user $\mathcal{U}$, including Llama 3.1 (8B, 70B), Gemma 3 (4B, 27B), Mistral NeMo 12B, and Qwen 2.5 7B. These models represent a cross-section of architectures to ensure the framework's generalizability across different LLM capabilities.

Suggestion models. We use the SIQSE framework to compare several keyphrase suggestion (a specific instance of query suggestion) models: (1) no expansions (BM25); (2) embedding-based suggestions (E5, FastKeywords, SBERT, PhraseBERT, and KBIR); (3) LLM-based suggestions (zero-shot suggestions from GPT-4o and Mistral); and (4) a RAG-based suggestion approach: RAKS (Retrieval-Augmented Keyprhase Suggestion) [14]).

4.2 Results and Discussion

4.2.1 RQ1: Surrogate fidelity. The first stage of our evaluation establishes the reliability of the LLM-based selector as a proxy for human decision-making. To test this, we constructed a custom dataset that challenges the surrogate to navigate a "noisy" suggestion space. For each query, we introduce 10 "non-manual" (automatically generated) keyphrase suggestions alongside the expert-curated manual suggestions.

We use the manual suggestions as the ground-truth "correct" labels for our evaluation proxy. We acknowledge that this is a non-perfect proxy, as automatically generated "noisy" suggestions may occasionally possess high semantic relevance or even surpass manual ones in specific contexts. However, treating the expert-curated terms as the gold standard allows us to measure the surrogate's alignment with human-intended search goals.

As shown in Table 1, the results, measured by the averaged F1-score, show that how the LLM interacts with the list is more important than how big the model is. A key finding is that iterative selection (picking one by one) works much better than batch selection (picking all at once). For example, the Qwen 2.5 7B model's score jumped from 0.52 to 0.73 on TREC 7 just by switching to iterative mode.

Table 1: Comparative analysis of selector alignment with human-generated suggestions across batch and iterative selection modes using the averaged F1-score. The results highlight the performance impact of incorporating explicit search intent versus relying on latent information needs. Bold values denote the optimal configuration for each respective model and dataset combination.

Model	Dataset	Batch Selection		Iterative Selection	
		Intent	No Intent	Intent	No Intent
Gemma 3 4B	CODEC	0.2897	0.2709	**0.3640**	0.3600
	TREC Fair	0.4412	0.4201	**0.4880**	0.4589
	TREC 7 (desc)	0.4208	0.3620	0.4181	**0.4700**
	TREC 7 (narr)	0.4366	0.3620	0.4052	**0.4700**
	TREC 8 (desc)	**0.4117**	0.3544	0.3893	0.3743
	TREC 8 (narr)	**0.4066**	0.3544	0.3676	0.3743
Gemma 3 27B	CODEC	0.4033	0.3438	**0.5435**	0.4933
	TREC Fair	**0.4705**	0.4255	0.4419	0.4589
	TREC 7 (desc)	0.5071	0.3727	**0.5100**	0.4514
	TREC 7 (narr)	**0.5158**	0.3727	0.5001	0.4514
	TREC 8 (desc)	0.5465	0.3927	0.5366	**0.5477**
	TREC 8 (narr)	0.5047	0.3927	0.4924	**0.5477**
Llama 3.1 8B	CODEC	0.4107	0.3584	**0.4372**	0.4298
	TREC Fair	0.3796	0.3012	**0.4167**	0.4078
	TREC 7 (desc)	0.4221	0.3372	**0.6124**	0.5789
	TREC 7 (narr)	0.4092	0.3372	**0.6151**	0.5789
	TREC 8 (desc)	0.4841	0.3968	**0.5999**	0.5939
	TREC 8 (narr)	0.4524	0.3968	**0.6523**	0.5939
Llama 3.1 70B	CODEC	0.3359	0.2829	**0.5128**	0.4986
	TREC Fair	**0.4344**	0.3976	**0.4334**	0.4023
	TREC 7 (desc)	0.4163	0.3350	**0.5624**	0.5239
	TREC 7 (narr)	0.4545	0.3350	**0.5973**	0.5239
	TREC 8 (desc)	0.4998	0.3115	**0.5966**	0.5393
	TREC 8 (narr)	0.4658	0.3115	**0.5855**	0.5393
Mistral NeMo 12B	CODEC	0.3588	0.2885	**0.4335**	0.3972
	TREC Fair	0.2687	0.2552	0.2763	**0.3621**
	TREC 7 (desc)	0.3307	0.1978	0.4055	**0.4141**
	TREC 7 (narr)	0.3599	0.1978	0.3932	**0.4141**
	TREC 8 (desc)	0.3601	0.2664	**0.4956**	0.4434
	TREC 8 (narr)	0.3756	0.2664	0.4169	**0.4434**
Qwen 2.5 7B	CODEC	0.4058	0.3200	0.4548	**0.4875**
	TREC Fair	0.4178	0.3861	**0.4755**	0.4595
	TREC 7 (desc)	0.5267	0.4878	**0.6813**	0.5995
	TREC 7 (narr)	0.5209	0.4878	**0.7322**	0.5995
	TREC 8 (desc)	0.5060	0.4989	**0.5831**	0.5345
	TREC 8 (narr)	0.4838	0.4989	**0.6101**	0.5345

This occurs because sequential selection enables the LLM to update its internal state while maintaining focus on individual suggestions. By processing terms one by one, the agent mimics the human cognitive limit for batch processing, avoiding the performance degradation often seen with long, noisy inputs. This step-by-step approach ensures each selection adds marginal utility and prevents the redundancy that arises when a model is forced to evaluate a large batch of candidates simultaneously.

4.2.2 RQ2: The role of user intent. Transitioning from the agent's behaviour to its contextual grounding, we examine how explicit intent influences selection precision. In our framework, *intent* is an umbrella construct realised through various dataset-specific metadata: *guidelines* in CODEC, *scope* in TREC Fair, and *narratives* and *descriptions* in TREC 7 and 8. Our results in Table 1 indicate a "selection sharpening" effect when the surrogate is provided with this "why" behind a search.

The inclusion of these intent forms, whether as a task-specific *scope* or a detailed search *narrative*, serves as a critical normative framework for the LLM. While a raw query provides a semantic target, these intent markers provide the *constraints* necessary to distinguish between thematic relevance and task-specific utility. This performance gap underscores a vital conclusion: query expansion in a simulated environment is not just a language exercise; it is a purpose-driven task aimed at finding specific information. Without intent, the surrogate defaults to surface-level semantic similarity, which often leads to query drift. However, when grounded in context, the agent acts as a high-pass filter, not including suggestions that are semantically related but contextually useless. This implies that the effectiveness of the SIQSE framework stems from its ability to ground the simulation in a clear, directed information need, bridging the "semantic gap" between generic keywords and specific user goals.

4.2.3 RQ3: Dynamic utility. Finally, we evaluate the retrieval utility of the SIQSE framework against standard static approaches. The results in Table 2 reveal a critical flaw in traditional evaluation: *static offline methodologies often fail to predict how a system performs in an interactive environment.*

We observe that system rankings frequently shift when moving from a static top-*e* setup to a simulated user loop. Several systems that appear weak under classic evaluation provide significantly better Precision (P@10) and Mean Average Precision (MAP@1000) when an agent selects the terms. This suggests that merely measuring a fixed set of suggestions is not enough to capture a system's true value.

This inconsistency is largely due to the "noise" inherent in fixed-length expansion. In static baselines, the system is forced to include a specific number of suggestions, which often introduces irrelevant terms that dilute the user's original intent. Our framework allows the surrogate to pick only the most useful terms. Furthermore, when using the *e=auto* setting, the agent can act as an "emergency brake", stopping the expansion entirely once it determines that additional terms will no longer improve the query.

Ultimately, these results show that ranking systems based on a fixed top-*e* selection is unreliable. Because static evaluation forces the inclusion of terms that a user would naturally not include in their query, it masks the true potential of models that might provide fewer, but more precise, options. By allowing for a dynamic selection process, we reveal a more accurate hierarchy of system performance, proving that the standard offline approach often identifies the wrong "best" model for interactive search.

Table 2: Retrieval effectiveness of static top-e selection vs. SIQSE (Llama 3.1 70B as the selection agent). Parentheses show baseline ranks; arrows and numbers indicate shifts in ranking relative to the static top-e baseline.

Method	MAP@1000								P@10							
	Top-e		Iterative w/intent			Iterative wo/intent			Top-e		Iterative w/intent			Iterative wo/intent		
	e=5	e=man	e=5	e=man	e=auto	e=5	e=man	e=auto	e=5	e=man	e=5	e=man	e=auto	e=5	e=man	e=auto
None (BM25)				0.0568								0.2188				
Manual	0.2145	0.2546	–	–	–	–	–	–	0.6792	0.7562	–	–	–	–	–	–
Embedding-based methods																
E5	0.0806 (4)	0.0812 (6)	0.0849 = 0	0.0852 ↑ 1	0.0809 ↑ 1	0.0887 ↑ 1	0.0854 ↑ 1	0.0867 ↑ 1	0.3188 (6)	0.3292 (4)	0.3042 = 0	0.3083 ↓ 1	0.3083 ↓ 1	0.3229 ↑ 2	0.3104 ↓ 2	0.3250 ↓ 2
FastKwds	0.0218 (9)	0.0246 (9)	0.0264 = 0	0.0274 = 0	0.0296 = 0	0.0273 = 0	0.0275 = 0	0.0254 = 0	0.0958 (9)	0.0958 (9)	0.1208 = 0	0.1292 = 0	0.1354 = 0	0.1208 = 0	0.1271 = 0	0.1250 = 0
SBERT	0.0750 (7)	0.0730 (8)	0.0909 ↑ 3	0.0877 ↑ 4	0.0887 ↑ 4	0.0899 ↑ 3	0.0844 ↑ 4	0.0851 ↑ 4	0.3167 (7)	0.3021 (7)	0.3437 ↑ 5	0.3396 ↑ 4	0.3292 ↑ 3	0.3104 ↑ 1	0.3125 ↑ 2	0.3312 ↑ 3
PhraseBERT	0.0735 (8)	0.0748 (7)	0.0913 ↑ 5	0.0894 ↑ 4	0.0909 ↑ 4	0.0867 ↑ 4	0.0902 ↑ 4	0.0948 ↑ 5	0.2833 (8)	0.3000 (8)	0.3333 ↑ 4	0.3271 ↑ 4	0.3521 ↑ 6	0.3229 ↑ 5	0.3625 ↑ 6	0.3750 ↑ 6
KBIR	0.0622 (9)	0.0630 (9)	0.0641 = 0	0.0652 = 0	0.0668 = 0	0.0684 = 0	0.0691 = 0	0.0669 = 0	0.2667 (9)	0.2792 (9)	0.2875 = 0	0.2896 ↑ 1	0.2979 ↑ 1	0.2937 = 0	0.2958 ↑ 2	0.2875 ↑ 1
LLM-based methods																
GPT-4o	0.0786 (5)	0.0819 (5)	0.0700 ↓ 2	0.0670 ↓ 3	0.0676 ↓ 3	0.0697 ↓ 2	0.0675 ↓ 3	0.0803 ↓ 1	0.3250 (3)	0.3250 (6)	0.3125 ↓ 2	0.2896 ↓ 3	0.3062 = 0	0.2979 ↓ 5	0.2896 ↓ 2	0.3333 ↑ 3
Mistral	0.0938 (2)	0.0988 (2)	0.0764 ↓ 3	0.0737 ↓ 3	0.0726 ↓ 4	0.0728 ↓ 3	0.0722 ↓ 3	0.0679 ↓ 4	0.3208 (5)	0.3271 (5)	0.3042 ↓ 1	0.3062 ↓ 1	0.3000 ↓ 2	0.3042 ↓ 2	0.2896 ↓ 4	0.2750 ↓ 4
RAG-based methods																
RAKS GPT-4o	0.0888 (3)	0.0894 (4)	0.0753 ↓ 3	0.0735 ↓ 3	0.0750 ↓ 2	0.0791 ↓ 1	0.0751 ↓ 3	0.0769 ↓ 2	0.3229 (4)	0.3333 (3)	0.2958 ↓ 4	0.2979 ↓ 4	0.2938 ↓ 6	0.3187 ↓ 1	0.3146 ↓ 1	0.2854 ↓ 4
RAKS Mistral	0.0997 (1)	0.1090 (1)	0.0983 = 0	0.1012 = 0	0.0970 = 0	0.0963 = 0	0.0902 = 0	0.0870 = 0	0.3771 (1)	0.3646 (1)	0.3354 ↓ 1	0.3542 = 0	0.3438 ↓ 1	0.3333 = 0	0.3146 ↓ 1	0.3292 ↓ 3

(a) TREC Fair

Method	MAP@1000								P@10							
	Top-e		Iterative w/intent			Iterative wo/intent			Top-e		Iterative w/intent			Iterative wo/intent		
	e=5	e=man	e=5	e=man	e=auto	e=5	e=man	e=auto	e=5	e=man	e=5	e=man	e=auto	e=5	e=man	e=auto
None (BM25)				0.1675								0.3820				
Manual	0.2362	0.2708	–	–	–	–	–	–	0.5020	0.5640	–	–	–	–	–	–
Embedding-based methods																
E5	0.1836 (5)	0.1695 (5)	0.2006 = 0	0.1876 = 0	0.1906 = 0	0.1952 = 0	0.1897 = 0	0.1859 = 0	0.3660 (6)	0.3720 (6)	0.4240 = 0	0.4000 ↓ 1	0.4380 ↑ 1	0.4280 = 0	0.4060 ↓ 1	0.4060 ↑ 1
FastKwds	0.1272 (8)	0.1136 (8)	0.1293 ↓ 1	0.1196 ↓ 1	0.1307 ↓ 1	0.1237 ↓ 1	0.1210 ↓ 1	0.1242 ↓ 1	0.2800 (8)	0.2560 (8)	0.2980 ↓ 1	0.2800 ↓ 1	0.2940 ↓ 1	0.2840 ↓ 1	0.2760 ↓ 1	0.2960 ↓ 1
SBERT	0.1749 (6)	0.1570 (6)	0.1940 = 0	0.1859 = 0	0.1818 = 0	0.1882 = 0	0.1718 = 0	0.1661 = 0	0.3740 (5)	0.3200 (7)	0.4380 = 0	0.4120 ↑ 1	0.3940 ↑ 1	0.4220 = 0	0.3920 ↑ 1	0.3740 ↑ 1
PhraseBERT	0.1183 (9)	0.1048 (9)	0.1647 ↑ 1	0.1515 ↑ 2	0.1539 ↑ 2	0.1539 ↑ 1	0.1394 ↑ 1	0.1496 ↑ 1	0.2380 (9)	0.2220 (9)	0.3800 ↑ 1	0.3500 ↑ 1	0.3200 ↑ 1	0.3240 ↑ 1	0.2900 ↑ 1	0.3400 ↑ 1
KBIR	0.1480 (7)	0.1241 (7)	0.1718 = 0	0.1502 ↓ 1	0.1435 ↓ 1	0.1722 = 0	0.1540 = 0	0.1529 = 0	0.3440 (7)	0.3140 (8)	0.4100 = 0	0.3820 = 0	0.3380 = 0	0.3820 = 0	0.3600 = 0	0.3720 = 0
LLM-based methods																
GPT-4o	0.1994 (4)	0.1862 (3)	0.2113 = 0	0.2084 = 0	0.2084 = 0	0.2050 = 0	0.2077 = 0	0.1973 = 0	0.4260 (4)	0.4220 (4)	0.4680 = 0	0.4680 ↑ 1	0.4640 ↑ 1	0.4520 = 0	0.4620 = 0	0.4340 ↑ 1
Mistral	0.2187 (1)	0.2095 (2)	0.2408 = 0	0.2171 = 0	0.2127 = 0	0.2243 ↓ 1	0.2147 = 0	0.2088 = 0	0.4580 (3)	0.4540 (3)	0.4720 ↓ 1	0.4540 ↓ 1	0.4500 ↓ 1	0.4540 ↓ 1	0.4300 ↓ 1	0.4460 ↓ 1
RAG-based methods																
RAKS GPT-4o	0.2179 (3)	0.1853 (4)	0.2225 = 0	0.2033 = 0	0.1996 = 0	0.2105 = 0	0.1994 = 0	0.1884 = 0	0.4460 (4)	0.3880 (5)	0.4740 ↑ 1	0.4260 = 0	0.4300 = 0	0.4700 ↑ 1	0.4220 = 0	0.3980 = 0
RAKS Mistral	0.2186 (2)	0.2243 (1)	0.2370 = 0	0.2290 = 0	0.2195 = 0	0.2339 ↑ 1	0.2201 = 0	0.2130 = 0	0.4680 (1)	0.4780 (1)	0.5000 = 0	0.4920 = 0	0.4740 = 0	0.4940 = 0	0.4640 = 0	0.4660 = 0

(b) TREC 7 (narrative intent)

Method	MAP@1000								P@10							
	Top-e		Iterative w/intent			Iterative wo/intent			Top-e		Iterative w/intent			Iterative wo/intent		
	e=5	e=man	e=5	e=man	e=auto	e=5	e=man	e=auto	e=5	e=man	e=5	e=man	e=auto	e=5	e=man	e=auto
None (BM25)				0.1795								0.3260				
Manual	0.2806	0.3045	–	–	–	–	–	–	0.5240	0.5480	–	–	–	–	–	–
Embedding-based methods																
E5	0.2069 (2)	0.1918 (5)	0.2227 ↓ 2	0.2154 = 0	0.2171 = 0	0.2103 ↓ 2	0.2059 = 0	0.2032 = 0	0.4240 (4)	0.4100 (4)	0.4540 ↓ 1	0.4400 ↓ 1	0.4300 ↓ 2	0.4220 ↓ 2	0.4340 ↑ 1	0.4160 ↓ 1
FastKwds	0.1065 (9)	0.0943 (9)	0.1385 = 0	0.1324 = 0	0.1503 = 0	0.1368 = 0	0.1326 = 0	0.1352 = 0	0.1720 (9)	0.1520 (9)	0.2300 = 0	0.2260 = 0	0.2660 = 0	0.2260 = 0	0.2260 = 0	0.2420 = 0
SBERT	0.1879 (6)	0.1775 (6)	0.2034 = 0	0.1955 ↑ 1	0.2132 ↑ 1	0.2000 = 0	0.1928 ↑ 1	0.1925 ↑ 1	0.3540 (7)	0.3540 (6)	0.4100 ↓ 1	0.3780 ↓ 1	0.4080 ↓ 1	0.3820 ↓ 1	0.3740 ↓ 1	0.3880 ↓ 1
PhraseBERT	0.1764 (7)	0.1579 (7)	0.1941 = 0	0.1920 = 0	0.1898 = 0	0.1899 = 0	0.1764 = 0	0.1910 = 0	0.3720 (6)	0.3340 (7)	0.3800 ↓ 1	0.3860 = 0	0.3860 = 0	0.3980 ↑ 1	0.4040 ↑ 1	0.4080 ↑ 1
KBIR	0.1565 (8)	0.1355 (8)	0.1897 = 0	0.1760 = 0	0.1627 = 0	0.1549 = 0	0.1529 = 0	0.1508 = 0	0.3220 (8)	0.2800 (8)	0.4100 ↑ 1	0.3880 ↑ 1	0.3260 = 0	0.3320 = 0	0.3320 ↑ 1	0.3140 = 0
LLM-based methods																
GPT-4o	0.2030 (4)	0.2073 (2)	0.2682 ↑ 3	0.2580 ↑ 1	0.2367 ↑ 1	0.2528 ↑ 3	0.2276 ↑ 1	0.2123 ↑ 1	0.4340 (3)	0.4580 (2)	0.5080 ↑ 1	0.5160 = 0	0.4740 ↑ 1	0.5100 ↑ 1	0.4660 = 0	0.4700 ↑ 1
Mistral	0.2022 (5)	0.2000 (3)	0.2420 ↑ 1	0.2322 ↑ 1	0.2363 ↑ 1	0.2234 ↑ 1	0.2204 ↑ 1	0.2064 ↑ 1	0.4120 (5)	0.3880 (5)	0.4640 ↑ 1	0.4660 ↑ 1	0.4460 ↑ 1	0.4240 ↑ 1	0.4180 ↑ 1	0.4120 ↑ 1
RAG-based methods																
RAKS GPT-4o	0.2304 (3)	0.1985 (4)	0.2623 ↓ 1	0.2518 ↑ 1	0.2438 ↑ 1	0.2444 ↓ 1	0.2258 ↑ 1	0.1944 = 0	0.4980 (1)	0.4480 (3)	0.5080 = 0	0.4980 ↑ 1	0.4860 = 0	0.4640 ↓ 1	0.4500 ↑ 1	0.4440 ↑ 1
RAKS Mistral	0.2641 (1)	0.2421 (1)	0.2680 ↓ 1	0.2686 ↓ 1	0.2628 ↓ 1	0.2630 ↓ 1	0.2337 ↓ 1	0.2269 ↓ 1	0.4920 (2)	0.4640 (1)	0.4700 ↓ 2	0.5020 ↓ 1	0.4860 ↓ 1	0.4700 ↑ 1	0.4340 ↓ 1	0.4460 ↓ 1

(c) TREC 8 (narrative intent)

5 Conclusion and Future Work

For too long, the evaluation of query suggestions has been treated as a static snapshot: a single moment where a system presents a list and a user supposedly consumes it in its entirety. In this paper, we have argued that this top-*e* approach is a relic of a pre-interactive era. By introducing SIQSE, we have provided a bridge between the efficiency of offline benchmarks and the dynamic reality of human search behaviour. Through multi-dataset validation, we show that LLMs are effective surrogates that expose performance nuances hidden by traditional selection strategies. By simulating user decision-making, SIQSE provides high-fidelity evaluation without the cost of human-in-the-loop testing. This approach moves beyond static scoring to reveal how retrieval systems perform when suggestions are subject to human-like scrutiny.

Our findings demonstrate that the static top-*e* evaluation fundamentally misrepresents system performance. By forcing the inclusion of terms that a user would naturally reject, traditional metrics penalise precision and distort the true ranking of retrieval models. Moving beyond these fixed lists reveals that the "best" model in practice is often different from what static baselines suggest.

Current suggestion systems are largely one-shot generators, they produce a list based on the current query and then reset. However, the simulation framework we have established opens the space to build systems that are truly *state-aware*. Future work will explore models that:

- *Leverage sequential feedback:* Unlike existing systems, these models will treat the "user's" previous selections as high-signal feedback. If a user selects a specific technical term from a list, the next set of suggestions should instantly pivot to reflect that refined intent, rather than repeating generic alternatives.
- *Corrective query reformulation:* Future iterations of the simulator will allow the surrogate to *remove* previously added suggestions, not just append new ones. This capability will enable the agent to correct mistakes or pivot strategies if a previous selection leads to a semantic dead-end.
- *Active document consumption:* We aim to model the cost of examining results by requiring the agent to explicitly decide whether to "read" a document. Only upon this decision will the agent access the document's content or relevance label, mimicking the real-world trade-off users face between exploring results and refining their query.

Acknowledgments

First and second authors acknowledge funding from the Ministry of Science, Innovation and Universities of the Government of Spain (project PID2022-137061OB-C21, MCIN/AEI/10.13039/501100011033), as well as from the Department of Education, Science, Universities, and Vocational Training of the Xunta de Galicia (grant GRC ED431C 2025/49). CITIC, as a center accredited for excellence within the Galician University System and a member of the CIGUS Network, receives subsidies from the Department of Education, Science, Universities, and Vocational Training of the Xunta de Galicia. Additionally, it is co-financed by the EU through the FEDER Galicia 2021-27 operational program (Ref. ED431G 2023/01).

References

[1] Peter Bailey, Alistair Moffat, Falk Scholer, and Paul Thomas. 2016. UQV100: A Test Collection with Query Variability. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Pisa, Italy) *(SIGIR '16)*. Association for Computing Machinery, New York, NY, USA, 725–728. doi:10.1145/2911451.2914671

[2] Nicholas J Belkin. 1980. Anomalous states of knowledge as a basis for information retrieval. *Canadian journal of information science* 5, 1 (1980), 133–143.

[3] Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. InPars: Unsupervised Dataset Generation for Information Retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Madrid, Spain) *(SIGIR '22)*. Association for Computing Machinery, New York, NY, USA, 2387–2392. doi:10.1145/3477495.3531863

[4] Pia Borlund. 2000. Experimental components for the evaluation of interactive information retrieval systems. *Journal of documentation* 56, 1 (2000), 71–90.

[5] Pia Borlund. 2003. The IIR evaluation model: a framework for evaluation of interactive information retrieval systems. *Inf. Res.* 8, 3 (2003). http://www.informationr.net/ir/8-3/paper152.html

[6] Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. 2008. Context-aware query suggestion by mining click-through and session data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Las Vegas, Nevada, USA) *(KDD '08)*. Association for Computing Machinery, New York, NY, USA, 875–883. doi:10.1145/1401890.1401995

[7] Cyril Cleverdon. 1997. *The Cranfield tests on index language devices.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 47–59.

[8] Paul D. Clough and Mark Sanderson. 2013. Evaluating the performance of information retrieval systems using test collections. *Inf. Res.* 18, 2 (2013). http://www.informationr.net/ir/18-2/paper582.html

[9] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Jimmy Lin. 2021. Overview of the TREC 2021 Deep Learning Track. In *Proceedings of the Thirtieth Text REtrieval Conference, TREC 2021, online, November 15-19, 2021.*

[10] Silviu Cucerzan and Ryen W. White. 2007. Query suggestion based on user landing pages. 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, 875–876. doi:10.1145/1277741.1277953

[11] John J. Darragh and Ian H. Witten. 1992. *The Reactive Keyboard.* Cambridge University Press, USA.

[12] Michael D. Ekstrand, Amifa Raj, Graham McDonald, and Isaac Johnson. 2021. Overview of the TREC 2021 Fair Ranking Track. In *Proceedings of the Thirtieth Text REtrieval Conference, TREC 2021, online, November 15-19, 2021 (NIST Special Publication, Vol. 500-335)*, Ian Soboroff and Angela Ellis (Eds.). National Institute of Standards and Technology (NIST). https://trec.nist.gov/pubs/trec30/papers/Overview-F.pdf

[13] Guglielmo Faggioli, Laura Dietz, Charles L. A. Clarke, Gianluca Demartini, Matthias Hagen, Claudia Hauff, Noriko Kando, Evangelos Kanoulas, Martin Potthast, Benno Stein, and Henning Wachsmuth. 2023. Perspectives on Large Language Models for Relevance Judgment. In *Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval* (Taipei, Taiwan) *(ICTIR '23)*. Association for Computing Machinery, New York, NY, USA, 39–50. doi:10.1145/3578337.3605136

[14] Jorge Gabín and Javier Parapar. 2025. Leveraging Retrieval-Augmented Generation for Keyphrase Synonym Suggestion. In *Advances in Information Retrieval*, Claudia Hauff, Craig Macdonald, Dietmar Jannach, Gabriella Kazai, Franco Maria Nardini, Fabio Pinelli, Fabrizio Silvestri, and Nicola Tonellotto (Eds.). Springer Nature Switzerland, Cham, 311–327.

[15] Ahmed Hassan, Xiaolin Shi, Nick Craswell, and Bill Ramsey. 2013. Beyond clicks: query reformulation as a predictor of search satisfaction. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management* (San Francisco, California, USA) *(CIKM '13)*. Association for Computing Machinery, New York, NY, USA, 2019–2028. doi:10.1145/2505515.2505682

[16] William R. Hersh and Paul Over. 2000. The TREC-9 Interactive Track Report. In *Proceedings of The Ninth Text REtrieval Conference, TREC 2000, Gaithersburg, Maryland, USA, November 13-16, 2000 (NIST Special Publication, Vol. 500-249)*, Ellen M. Voorhees and Donna K. Harman (Eds.). National Institute of Standards and Technology (NIST). http://trec.nist.gov/pubs/trec9/papers/t9irep.pdf

[17] Evangelos Kanoulas, Ben Carterette, Paul D. Clough, and Mark Sanderson. 2011. Evaluating multi-query sessions. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Beijing, China) *(SIGIR '11)*. Association for Computing Machinery, New York, NY, USA, 1053–1062. doi:10.1145/2009916.2010056

[18] Diane Kelly and Cassidy R. Sugimoto. 2013. A systematic review of interactive information retrieval evaluation studies, 1967–2006. *Journal of the American Society for Information Science and Technology* 64, 4 (2013), 745–770. doi:10.1002/asi.22799

[19] Ron Kohavi, Roger Longbotham, Dan Sommerfield, and Randal M. Henne. 2009. Controlled experiments on the web: survey and practical guide. *Data Mining and Knowledge Discovery* 18, 1 (2009), 140–181. doi:10.1007/s10618-008-0114-1

[20] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

[21] Craig Macdonald and Nicola Tonellotto. 2020. Declarative Experimentation inInformation Retrieval using PyTerrier. In *Proceedings of ICTIR 2020*.

[22] Iain Mackie, Paul Owoicho, Carlos Gemmell, Sophie Fischer, Sean MacAvaney, and Jeffery Dalton. 2022. CODEC: Complex Document and Entity Collection. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[23] David Maxwell and Leif Azzopardi. 2016. Agents, Simulated Users and Humans: An Analysis of Performance and Behaviour. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management* (Indianapolis, Indiana, USA) *(CIKM '16)*. Association for Computing Machinery, New York, NY, USA, 731–740. doi:10.1145/2983323.2983805

[24] 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*.

[25] Paul Over. 1996. TREC-5 Interactive Track Report. In *Proceedings of The Fifth Text REtrieval Conference, TREC 1996, Gaithersburg, Maryland, USA, November 20-22, 1996 (NIST Special Publication, Vol. 500-238)*, Ellen M. Voorhees and Donna K. Harman (Eds.). National Institute of Standards and Technology (NIST). http://trec.nist.gov/pubs/trec5/papers/trackreport.ps.gz

[26] Paul Over. 1997. TREC-6 Interactive Report. In *Proceedings of The Sixth Text REtrieval Conference, TREC 1997, Gaithersburg, Maryland, USA, November 19-21, 1997 (NIST Special Publication, Vol. 500-240)*, Ellen M. Voorhees and Donna K. Harman (Eds.). National Institute of Standards and Technology (NIST), 73–81. http://trec.nist.gov/pubs/trec6/papers/t6irep.ps

[27] Nirmal Roy, Leonardo F. R. Ribeiro, Rexhina Blloshmi, and Kevin Small. 2024. Learning When to Retrieve, What to Rewrite, and How to Respond in Conversational QA. 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, 10604–10625. doi:10.18653/v1/2024.findings-emnlp.622

[28] Paul Thomas, Seth Spielman, Nick Craswell, and Bhaskar Mitra. 2024. Large Language Models can Accurately Predict Searcher Preferences. 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, 1930–1940. doi:10.1145/3626772.3657707

[29] Ellen M. Voorhees. 2001. The Philosophy of Information Retrieval Evaluation. In *Evaluation of Cross-Language Information Retrieval Systems, Second Workshop of the Cross-Language Evaluation Forum, CLEF 2001, Darmstadt, Germany, September 3-4, 2001, Revised Papers (Lecture Notes in Computer Science, Vol. 2406)*, Carol Peters, Martin Braschler, Julio Gonzalo, and Michael Kluck (Eds.). Springer, 355–370. doi:10.1007/3-540-45691-0_34

[30] Ellen M. Voorhees and Donna Harman. 1999. Overview of the Eighth Text REtrieval Conference (TREC-8). In *Proceedings of The Eighth Text REtrieval Conference, TREC 1999, Gaithersburg, Maryland, USA, November 17-19, 1999 (NIST Special Publication, Vol. 500-246)*, Ellen M. Voorhees and Donna K. Harman (Eds.). National Institute of Standards and Technology (NIST). http://trec.nist.gov/pubs/trec8/papers/overview_8.ps

[31] Ellen M. Voorhees and Donna K. Harman. 1999. Overview of the Seventh Text Retrieval Conference (TREC-7). https://api.semanticscholar.org/CorpusID:13102016