MultiConIR: Towards Multi-Condition Information Retrieval

Anonymous ACL submission

Abstract

Multi-condition information retrieval (IR) presents a significant, yet underexplored challenge for existing systems. This paper introduces **MultiConIR**, the first benchmark specifically designed to evaluate retrieval and reranking models under nuanced multi-condition query scenarios across five diverse domains. We systematically assess model capabilities through three critical tasks: complexity robustness, relevance monotonicity, and query format sensitivity. Our extensive experiments on 15 models reveal a critical vulnerability: most retrievers and rerankers exhibit severe performance degradation as query complexity increases. Key deficiencies include widespread failure to maintain relevance monotonicity, and high sensitivity to query style and condition placement. The superior performance GPT-4o reveals the performance gap between IR systems and advanced LLM for handling sophisticated natural language queries. Furthermore, this work delves into the factors contributing to reranker performance deterioration and examines how condition positioning within queries affects similarity assessment, providing crucial insights for advancing IR systems towards complex search scenarios.

1 Introduction

Information retrieval (IR) is critical for helping users find relevant information across various domains. Traditionally, IR systems retrieve documents by matching queries based on lexical similarity, such as BM25 (Carpineto and Romano, 2012; Ponte and Croft, 2017), or semantic similarity using dense vector representations (Karpukhin et al., 2020; Zhan et al., 2021). Though highly effective for queries with straightforward query-document relationships (Su et al., 2024), they often fail to fully capture nuanced intent as user needs become more complex (Zhu et al., 2023; Su et al., 2024).

A significant challenge arises when users specify multiple requirements simultaneously, as illustrated in Fig. 1. Whether searching for a movie with specific attributes or selecting a product that meets various criteria, multi-condition search has become an integral part of modern information-seeking behavior. Traditional IR systems handle such scenarios using structured filtering, such as SQL-based queries that retrieve information from backend databases based on predefined conditions. However, this approach is inherently rigid and limited, as it relies on explicitly defined attributes and lacks the flexibility to accommodate evolving or diverse user preferences. As a result, it struggles to support nuanced or semantic-level queries that go beyond structured data filtering.

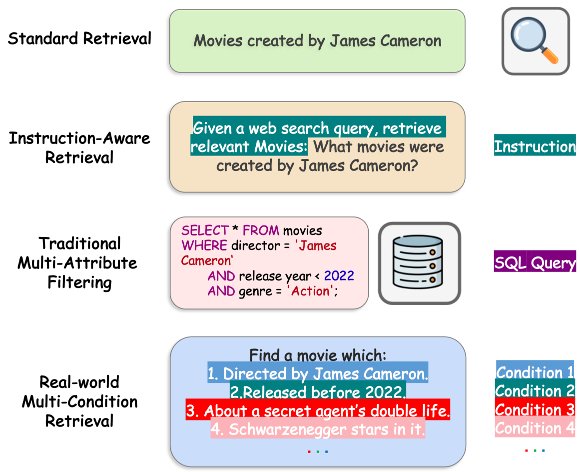


Figure 1: From single-condition to multi-condition retrieval. Standard and instruction-aware retrieval address single-condition queries. SQL-based filtering is restricted to predefined attributes within structured data. Real-world multi-condition retrieval enables the formulation of multiple, often semantic, conditions using natural language query.

The advent of Large Language Models (LLMs) has enhanced IR by introducing instruction-following capabilities (Asai et al., 2023; Weller et al., 2024a; Oh et al., 2024). This approach augments standard queries with explicit instructions, which serve as additional constraints to refine search results, as shown in Fig.1. Despite these advancements, existing evaluation benchmarks remain predominantly focused on single-condition queries and binary relevance assessments—classifying documents as either relevant or irrelevant (Nguyen et al., 2016; Kwiatkowski et al., 2019; Muennighoff et al., 2022)—thus overlooking the nuanced challenges of multi-condition queries, where relevance depends on the degree to which multiple conditions are satisfied.

An ideal multi-condition retrieval system should exhibit the following properties: (1) **Complexity Robustness:** The system should maintain high performance regardless of query complexity (i.e., the number of conditions specified); (2) **Relevance Monotonicity:** The relevance scores should scale monotonically with the number of matched conditions; for example, a document matching all n conditions should be ranked higher than one matching $n-1$; (3) **Format Invariance:** The system should yield consistent results regardless of the query format, whether presented as a structured list or as free-form natural language.

Existing benchmarks do not offer a structured framework for evaluating multi-condition retrieval along these dimensions. To address this gap, we introduce **MULTICONIR**—the first benchmark designed to comprehensively evaluate multi-condition retrieval systems. Through systematic experiments on 15 state-of-the-art models (including dense retrievers, cross-encoders, and LLM-based agents), we uncover several critical insights:

- **Multi-Condition Struggle:** Retrievers and Rerankers struggle with multi-condition retrieval, showing performance decline as query conditions increase, difficulty with relevance monotonicity, and sensitivity to query style variations.

- **Retrievers and Rerankers Differ:** Rerankers excel with single-condition queries but fail under multiple conditions. Retrievers demonstrate greater robustness. GritLM demonstrates the best robustness among retrievers.

- **Position Impacts Model Focus:** Dense retriever pooling strategies emphasize different condition positions, mean pooling focuses on initial positions, while <EOS> pooling emphasizes final positions. Rerankers exhibit non-uniform attention across positions and greater sensitivity to document length variations.

By quantifying these gaps, our work reveals key deficiencies in the ability of current IR systems to understand multi-condition intent, laying the groundwork for advancing IR toward human-like reasoning in complex search scenarios.

2 Related Works

Retriever: From Sparse To Dense Traditional sparse retrieval methods are based on BM25 (Robertson and Zaragoza, 2009), TF-IDF (Ramos et al., 2003), etc., rely on keyword matching and statistical weighting to evaluate relevance, which suffers from the well-known issue of lexical gap (Berger et al., 2000), restricting their ability to effectively capture semantic relationships (Luan et al., 2021; Nian et al., 2024). Dense retrieval addresses this limitation by encoding both queries and documents as embeddings within a joint latent space, where the semantic relationship is captured through the similarity scores between their embeddings (Li et al., 2023a). Pre-trained language models like BERT (Devlin et al., 2019) and T5 (Raffel et al., 2020), are widely used as backbone encoders for dense retrieval (Li et al., 2023b; Sturua et al., 2024; Xiao et al., 2023). Recent advancements have shown that LLMs offer significant potential as backbone encoders for dense retrieval (Wang et al., 2024a; Weller et al., 2024c; BehnamGhader et al., 2024). For instance, Repllama (Ma et al., 2023) fine-tuned Llama-2 to serve as dense retrievers. GritLM (Muennighoff et al., 2024) unified text embedding and generation within a single LLM. LLM2Vec (BehnamGhader et al., 2024) introduced an unsupervised approach for transforming decoder-only LLMs into dense retrievers.

Benchmarks In Complex Retrieval Tasks Existing datasets for information retrieval, such as MS MARCO (Nguyen et al., 2016), Natural Questions (Kwiatkowski et al., 2019), and MTEB (Muennighoff et al., 2022), primarily focus on queries sourced from search engines. The relationships between queries and documents are typically simple and direct (Su et al., 2024). Recent studies have expanded retrieval benchmarks to address more complex scenarios. Instruction-based datasets (Weller et al., 2024a; Qin et al., 2024; Oh et al., 2024), for instance, evaluate the instruction-following capabilities of retrieval models by embedding explicit instructions within queries to better represent users' retrieval intents. Furthermore, some works have assessed retrieval models' abilities to handle logical reasoning tasks, including Boolean logic (Mai et al., 2024; Zhang et al., 2024c), negation (Zhang et al., 2024a; Weller et al., 2024b), and multi-hop reasoning (Su et al., 2024). These efforts mark significant progress in increasing query complexity. However, while research in the generative modeling domain has explored the ability of LLMs to handle multi-constraint instructions (He et al., 2024; Ferraz et al., 2024; Zhang et al., 2024b), studies on retrieval models in multi-condition scenarios remain sparse.

3 MULTICONIR

We introduce MULTICONIR, a benchmark designed to evaluate the capacity of retrieval models to process multi-condition queries. Formally, given a query q_k composed of k conditions $C = \{c_1, c_2, \ldots, c_k\}$ with $k \in \{1, \ldots, 10\}$, we construct a structured retrieval setup consisting of:

(1) **Two query formulations**, denoted as q_k^{inst} and q_k^{desc}, where q_k^{inst} corresponds to a structured instruction-style query, formally expressed as a tuple $q_k^{\text{inst}} = \langle f, C \rangle$ where f is an explicit function describing retrieval constraints, and q_k^{desc} is a natural language descriptive-style query, represented as an unstructured sequence about the same set C,

(2) A **positive document** d^+ that satisfies all k conditions, i.e., $d^+ \models C$,

(3) A sequence of **hard negative (HN) documents** $\{d_0, d_1, \ldots, d_{k-1}\}$, where each d_j satisfies exactly j out of k conditions, formally expressed as $d_j \models \{c_1, \ldots, c_j\}$ and $d_j \not\models \{c_{j+1}, \ldots, c_k\}$.

This controlled design enables a principled evaluation of multi-condition retrieval along three fundamental axes: (1) **Complexity Robustness:** The model's retrieval effectiveness as k increases, measured by its ability to distinguish d^+ from d_{k-1}; (2) **Relevance Monotonicity:** The extent to which the retrieval model enforces a strict ordering such that $S(q_k, d_j) > S(q_k, d_{j+1})$ for all j, ensuring that documents satisfying more conditions are ranked higher; and (3) **Format Invariance:** The stability of retrieval performance under transformations of query representation, quantified by discrepancies in ranking outcomes across query formats.

3.1 Domain Selection

To construct the MULTICONIR dataset, we meticulously selected five domains—Books, Movies, People, Medical Cases, and Legal Documents—each chosen for its practical significance and inherent suitability for evaluating multi-condition retrieval capabilities.

Books & Movies: These domains represent common consumer searches where nuanced preferences are expressed by combining structured attributes (e.g., genre, creator, year, cast) with narrative elements (e.g., *plot details, thematic content like "a story about time travel"*). Effective retrieval demands semantic understanding beyond simple keyword matching to process multifaceted queries, such as *"an action film directed by Christopher Nolan, starring Leonardo DiCaprio, released after 2010, with an intense chase scene."*

People: Queries about individuals frequently rely on partial or vague information, such as notable achievements or specific traits. An example query could be *"a Nobel laureate in Physics who studied black holes."* These searches demand that IR systems effectively handle incomplete data and infer connections between various attributes to identify the correct individual.

Medical Case & Legal Document: The medical case and legal document domains offer more practical and application-driven use cases. In the medical domain, doctors often rely on retrieval systems to reference historical cases to support diagnostic decisions. A typical query might include multiple conditions, such as *"Find a case that meets the following conditions: 1) middle-aged female patient; 2) hospitalized for breathing difficulties; 3) has a history of antibiotic allergies; 4) has a family history of peanut allergies."* Similarly, in the legal domain, retrieval users often seek case law with high similarity to ongoing cases, which requires matching various legal and factual attributes in historical court decisions. These complex queries require IR systems to perform fine-grained condition matching and understand the interdependencies between various factors.

3.2 Dataset Construction Pipeline

To construct MULTICONIR, we design a multi-step data generation framework. As shown in Fig. 2, this pipeline is highly adaptable across multiple domains, enabling the generation of queries and hard negative (HN) documents that progressively satisfy 1 to 10 conditions. To preserve dataset integrity and mitigate the generalization issues associated with fully synthetic datasets (Li et al., 2023c; Wang et al., 2024b), we employ LLM-based generation (GPT-4o) exclusively for modifying sentences within hard negatives, rather than altering entire

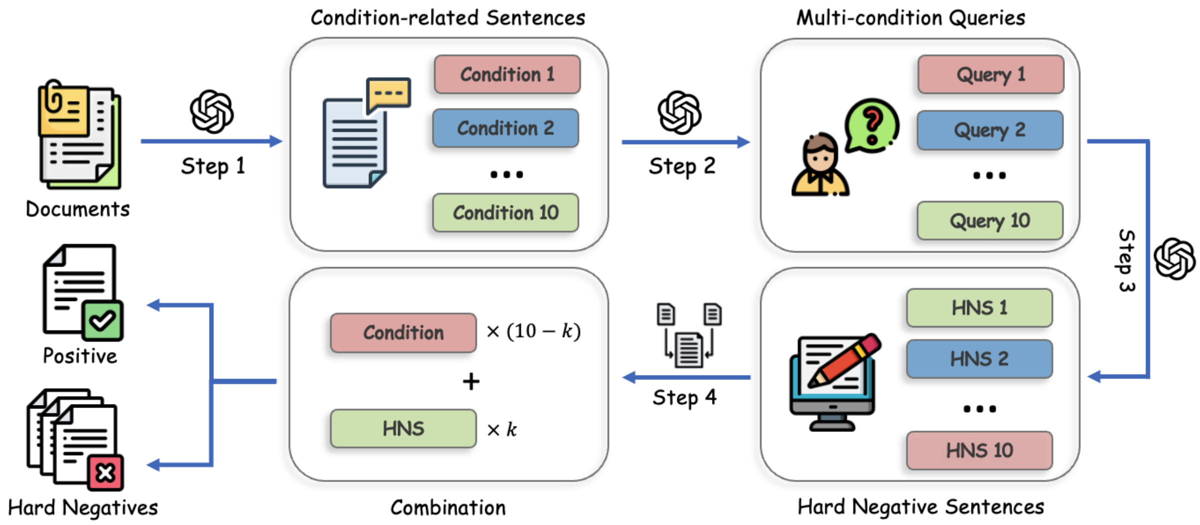


Figure 2: MULTICONIR Dataset Construction Pipeline: (1) relevant condition sentences are extracted from documents; (2) these conditions are then used to generate multi-condition queries; (3) hard negative (HN) versions of the condition sentences are created; and (4) positive documents and progressively challenging HN documents are assembled from these elements.

documents.[1] The data creation process consists of the following steps, with detailed prompt templates provided in Appendix B:

Step 1: Condition Sentences Extraction. For each source document d, we issue a structured prompt to GPT-4o that (1) reads the entire text, (2) identifies ten non-overlapping conditions that describe key features expressed in d, and (3) return ten original sentences, where every sentence is semantically complete, and expresses a distinct condition. Documents yielding fewer than ten qualified sentences are discarded. The final ten sentences constitute the condition set $C(d)$, which we later recombine into queries of varying complexity and use as the ground-truth positive for multi-condition retrieval experiments.

Step 2: Query Generation. Given the ten-sentence condition set $C(d)$, we prompt GPT-4o to synthesise a hierarchy of ten queries $\{q_1, q_2, ..., q_{10}\}$. Each q_k incorporating the first k conditions. To enhance linguistic diversity, For every q_k we request two forms: (1) Instruction-style q_k^{inst}: a bullet-like template (*"Find a document that satisfies: (1)... (2)..."*) for explicit parsing. (2) Descriptive-style q_k^{des}: embedding conditions naturally within a coherent sentence.

Step 3: Hard Negative Sentence Construction. For each condition sentence $c_i \in C(d)$ we instruct **GPT-4o** to produce a semantically divergent yet fluently written variant h_i that no longer satisfies the original constraint. The rewrite must preserve overall length and style while introducing either (i) a subtle alteration of a critical fact (applied to *books*, *movies*, *medical cases*, and *legal documents*) or (ii) an innocuous clause that injects misleading information without changing the existing keywords (used for the *people* corpus). [2] The ten variants form the hard-negative set $\text{HNS}(d) = \{h_1, \ldots, h_{10}\}$.

Step 4: Hard Negative Document Generation. Starting from the ten-sentence condition set $C(d) = \{c_1, \ldots, c_{10}\}$ and its hard-negative counterparts $\text{HNS}(d) = \{h_1, \ldots, h_{10}\}$, we build an ordinal ladder of document variants. The *positive* document d^+ contains the full sequence $[c_1, \ldots, c_{10}]$. For each $k \in \{1, \ldots, 10\}$ we generate a HN document $d_k^- = \left[h_1, \ldots, h_k,\ c_{k+1}, \ldots, c_{10}\right]$, i.e., the first k conditions are replaced by their semantically perturbed versions while the remaining $10 - k$ conditions stay intact. This yields a controlled degradation chain $d^+ \to d_1^- \to \cdots \to d_{10}^-$, ranging from a single violated constraint to a completely adversarial variant. *Coupled with the hierarchical queries* $\{Q_k\}_{k=1}^{10}$, *the corpus enables fine-grained evaluation of retrieval performance under progressively stricter condition sets.*

[1]Fully LLM-generated datasets may introducing inhert linguistic biases of the underlying LLMs, and lacks the contextual richness and complexity in real-world retrieval (Shumailov et al., 2024). To mitigate these issues, we restricted LLM interventions to modifying only condition sentences rather than entire document. We further discuss this problem in Appendix G.1.

[2]Retrievers are more robust when adding misleading information; Modifying critical facts is challenging for both retrievers and rerankers. A detailed comparison of these two strategies is given in Appendix G.2.

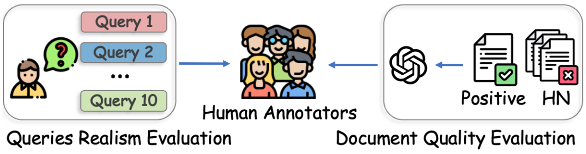


Figure 3: Benchmark quality evaluation framework of MULTICONIR. Query realism was assessed by human annotators. Label accuracy involved initial GPT-4o filtering, followed by a final human double-check.

Benchmark Quality Assurance. The reliability of MULTICONIR was audited on two complementary fronts (Fig. 3). **(1) Query Realism Evaluation.** From each of the five domains we randomly sampled 100 multi-condition queries (500 in total) and asked ten trained annotators to judge whether each query was *natural, precise,* and *contextual plausibility (realistic vs. unrealistic)*. The resulting inter-annotator agreement reached 93.7%, with Fleiss' $\kappa = 0.84$. **(2) Document–label validity.** To detect false positives/negatives, we first applied *GPT-4o* to the *entire* corpus: the model verified for every document variant d_k^- whether exactly k of the ten conditions were satisfied, and we discarded mismatched instances. We then drew another 100 document–query pairs per domain (500 total) for manual spot-checking; two independent annotators reviewed each pair and a third adjudicated disagreements, yielding a residual error rate of 2.4%. After these filtering steps the final benchmark sizes for each domain are summarised in Table 1, and the full evaluation protocol is detailed in Appendix C.

3.3 Evaluation Metrics

Conventional IR metrics—e.g., Precision@1, NDCG@k—only confirm whether a highly relevant document appears early, but they *cannot* distinguish how well a model orders candidates that satisfy different numbers of query conditions. We therefore propose **Win Rate**: the proportion of pairwise comparisons in which a candidate that fulfils more conditions ranks above one that fulfils fewer. We further discuss the difference of Win Rate and traditional IR metrics in the Appendix D

Complexity Robustness Queries range from `query1` to `query10`, each progressively incorporating 1 to 10 conditions. The candidate set comprises a `Positive` document that fully satisfies all conditions and a `HN1` document, which is derived from the positive by modifying a single condition. Complexity robustness is measured using **Win Rate (WR)** [3]under various k, defined as:

$$\mathrm{WR}_k = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}(S(q_k, d^+) > S(q_k, d_{k-1})),$$

where $S(q_i, d^+)$ and $S(q_i, d^-)$ denote similarity scores for the positive document and hard negative.

Relevance Monotonicity The query is fixed as `query10` (containing all 10 conditions), while the candidate set includes one `positive` and ten hard negatives ($d_0 - d_9$), each containing 0–9 conditions.

We evaluate performance using $\mathrm{WR}_{k,k-1}$ between adjacent hard negatives:

$$\mathrm{WR}_{k,k-1} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\big(S(q_{10}, d_k) > S(q_{10}, d_{k-1})\big),$$

Format invariance. We compare two query formats: (1) Instruction-style, which explicitly lists conditions (e.g., *Find a movie that meets the following conditions: 1. Action genre, 2. Directed by James Cameron*). (2) Descriptive-style, which integrates conditions into a natural query (e.g., *Find an action movie directed by James Cameron*).

To quantify ranking variability between query styles, we define the **Flip Rate (FR)**:

$$\mathrm{FR} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\big(\mathbf{1}(S_{\text{inst}}(q_{10}, d_k) > S_{\text{inst}}(q_{10}, d_{k-1})) \neq \mathbf{1}(S_{\text{desc}}(q_{10}, d_k) > S_{\text{desc}}(q_{10}, d_{k-1}))\big),$$

where S_{inst} and S_{desc} denote similarity scores under instruction-style and descriptive-style queries. The indicator function returns 1 if the ranking order of positive and hard negative documents changes between query styles and 0 otherwise. A higher FR indicates greater sensitivity to query formulation.

4 Experiments

We evaluate 15 representative retrieval and reranking models from diverse architectures and varying model sizes, including sparse retrieval model: BM25 (Robertson and Zaragoza, 2009); BERT-based retrieval models: gte-large-en-v1.5 (Li et al., 2023b) and jina-embeddings-v3 (Sturua et al., 2024); LLM-based retrieval models: NV-Embed-v2 (Lee et al., 2024), bge-en-icl (Li et al., 2024), gte-Qwen2-7B-instruct (Li

[3]In Complexity Robustness evaluation, WR_k and Precision@1 are numerically equivalent.

Domain	Number			Avg. Length		Source	Example
	D^+	Q	D^-	Q	D		
People	420	4200	4200	31.2	225.2	People Wikipedia Dat(Mahajan, 2017)	Table 7
Books	482	4820	4820	25.6	235.3	Books Dataset (Rustamov, 2021)	Table 8
Movies	500	5000	5000	24.8	184.6	Wikipedia Movie Plots (Robischon, 2018)	Table 9
Medical Case	479	4790	4790	28.4	212.1	Medical Cases (HPE AI Solutions, 2023)	Table 10
Legal Document	426	4260	4260	34.1	302.2	LexGLUE (Chalkidis et al., 2022)	Table 11

Table 1: Data statistics of MutiConIR. For each dataset, we show the number of positive documents (D^+), queries (Q) and hard negative documents(D^-), and the average length (in words) of queries and documents, and the source dataset of each domain.

et al., 2023b), gte-Qwen2-1.5B-instruct (Li et al., 2023b), e5-mistral-7b-instruct (Wang et al., 2024a), GritLM-7B (Muennighoff et al., 2024), LLM2Vec (BehnamGhader et al., 2024); Pointwise reranking models: bge-reranker-v2-m3 (Chen et al., 2024), bge-reranker-v2-gemma (Chen et al., 2024), FollowIR-7B (Weller et al., 2024a); Fine-tuned list-wise reranker: RankZephyr (Pradeep et al., 2023); Advanced LLM for zero-shot ranking: GPT-4o (OpenAI, 2024). Details of each model are provided in Appendix A.

4.1 Results for Complexity Robustness

Table 2 presents the average Win Rate scores for evaluating complexity robustness across five datasets. The results reveal several notable trends:

Performance decline with more conditions As the number of conditions in the query increases, the performance of both retrieval and reranking models declines. This suggests that with more conditions, models struggle to accurately distinguish between positives and HNs. Among all models, *GPT-4o* maintained the highest win rate from Query1 to Query10, with its performance declining by 9.23%. *GritLM-7B* exhibits the lowest performance degradation of 6.13%.The remaining models all exceeded 10% decline.

Rerankers exhibit steeper performance drop As shown in Table 2, fine-tuned rerankers outperform retrievers with single-condition queries. However, as the number of conditions increases, their performance declines more sharply. *Eventually, rerankers even fell behind some retrievers.* The Win Rates for all rerankers declined by over 25%, with an average decline of 35.76%. For retriever models, the average decline was 14.06%.

4.2 Results for Relevance Monotonicity

Fig. 4 illustrates the trend of average $\mathrm{WR}_{k,k-1}$ in the multi-condition retrieval setting of Task 2, which evaluates the model's ability to distinguish the relevance hierarchy among documents with varying conditions. The complete results are provided in Table 12. Several key observations can be made:

Relevance monotonicity struggle As documents become increasingly hard (i.e., satisfying more conditions in the query), it becomes harder for retrieval and reranking models to accurately distinguish d_k and d_{k-1}, leading to a decline in Win Rate performance. This failure emphasizes the challenge of preserving relevance monotonicity in multi-condition retrieval settings and highlights a gap in current model capabilities when handling complex queries.

Sensitive to exact match and complete mismatches We observe a slight upward trend at the end of Win Rate curves for most dense retrievers, likely due to their contrastive learning-based training. Traditional contrastive learning treats retrieval as a binary task, pulling query-positive pairs closer while pushing negatives further apart, without accounting for partial matches. As a result, dense retrievers perform more reliably in clear-cut "exact match" or "complete mismatch" cases.

4.3 Results for Format Invariance

Table 13 presents the Flip Rate induced by query format variations. *GPT-4o showed the lowest flip rate of 6.98%, showcasing the robustness of advanced LLMs against variations in query style.* Additionally, most models exceed 10%, indicating a substantial impact of query formulation on retrieval performance.

Dense retrieval models show relatively lower sensitivity than rerankers, with Flip Rates between 8% to 16%. GritLM-7B (8.21%), NV-Embed-v2 (9.12%), and LLM2Vec (9.78%) exhibit less variation. In contrast, *reranking models show significantly higher sensitivity to query format changes,*

Model	Query1	Query2	Query3	Query4	Query5	Query6	Query7	Query8	Query9	Query10	**Decline**
Sparse Retriever											
BM25	28.59	34	33.14	36.53	37.38	37.95	38.08	38.41	38.86	39.87	↓-11.28
Dense Retriever											
jina-embeddings-v3	76.09	71.26	71.84	71.04	65.59	65.65	64.75	64.24	64.62	60.71	↓15.38
gte-large-en-v1.5	75.87	77.26	73.79	70.22	70.71	67.40	67.53	64.97	65.36	61.36	↓14.51
NV-Embed-v2	80.53	80.32	78.81	75.70	75.68	72.61	73.28	71.54	70.00	68.02	↓12.51
bge-en-icl	83.42	80.65	78.44	76.77	74.54	73.00	74.23	73.25	69.70	68.00	↓15.42
gte-Qwen2-7B-instruct	70.75	72.22	69.99	68.51	65.20	63.53	62.22	62.20	59.15	56.17	↓14.58
gte-Qwen2-1.5B-instruct	73.64	74.97	72.23	71.37	69.94	67.46	66.92	64.64	63.65	58.68	↓14.96
e5-mistral-7b-instruct	75.05	70.85	68.18	67.45	63.70	61.60	59.70	59.07	57.85	58.12	↓16.93
GritLM-7B	82.08	80.32	78.38	76.40	76.40	73.50	75.69	74.62	74.53	75.95	**↓6.13**
LLM2Vec	83.13	77.42	75.49	75.48	72.49	72.56	70.56	70.73	68.71	67.00	↓16.13
Fine-tuned Reranker											
bge-reranker-v2-m3	87.14	85.56	78.62	76.05	74.29	68.41	67.86	59.48	55.59	44.87	↓42.27
bge-reranker-v2-gemma	91.07	90.02	86.70	84.99	83.17	79.00	75.89	72.29	67.11	56.09	↓34.98
followIR	83.41	79.72	76.25	74.60	70.12	67.94	62.62	55.93	48.59	43.52	↓39.89
RankZephyr	92.72	90.29	88.38	87.69	84.57	80.88	78.93	75.99	72.39	66.84	↓25.88
Zero-shot LLM for Ranking											
GPT-4o	**95.49**	**94.89**	**93.71**	**92.11**	**90.81**	**89.08**	**88.43**	**88.08**	**86.82**	**85.26**	↓9.23

Table 2: Impact of increasing condition quantity in queries on average Win Rate (Task 1). The Decline reflects the degree of Win Rate reduction from query1 to query10.

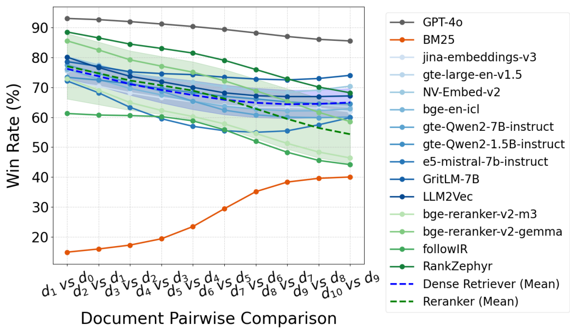


Figure 4: Relevance Monotonicity Distinction. Win Rate reflects the success rate between documents satisfying different numbers of conditions under a multi-condition query.

with Flip Rates exceeding 20%. The highest Flip Rate observed is 33.81% for bge-reranker-v2-m3.

5 Analysis

5.1 Retrievers vs. Rankers

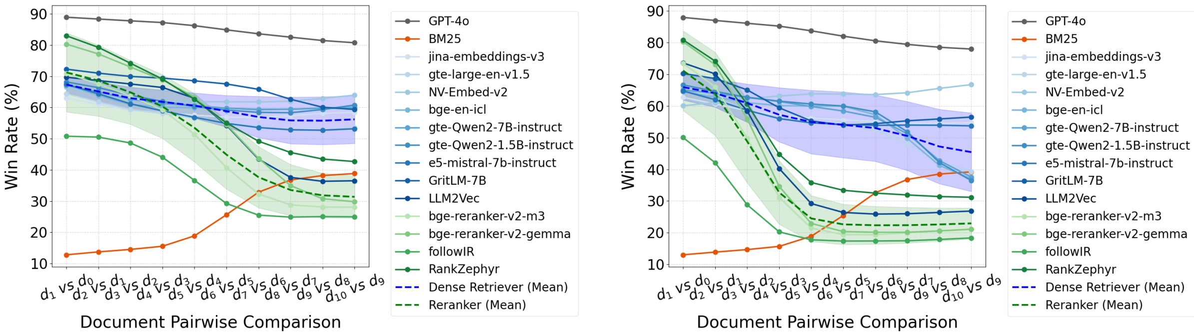
Our experiments reveal notable differences between retrievers and rerankers across the three tasks. Fine-tuned rerankers exhibited excellent ranking performance under single-condition queries, but their efficacy rapidly diminished as the number of conditions increased. Retrievers demonstrate greater robustness under multi-condition queries and query style variations.

We hypothesize that one contributing factor to these performance disparities lies in the training datasets. Many dense retrieval models are trained on a mixture of retrieval-specific and general textual datasets (Lee et al., 2024; BehnamGhader et al., 2024; Wang et al., 2024a). Such diverse training enhances their generalization across various retrieval scenarios and query styles, which, in turn, improves their robustness against query complexity.

Beyond training data, we posit that the distinct input processing mechanisms also contribute to the observed performance differences. Retriever models typically employ a bi-encoder architecture, processing queries and documents independently. Conversely, rerankers, which process a concatenation of the query and document as a single input, appear more susceptible to input complexification—whether arising from an increase in query conditions or changes in query style. To validate this speculation, we re-evaluated the Win Rate for the relevance monotonicity task under multi-condition queries using documents that padded to 512 and 1024 words. Results as shown in Fig.5 , revealed that fine-tuned rerankers are highly sensitive to such increases in document length, which further illustrates the sensitivity of rerankers to complex input. In contrast, retriever models demonstrated greater robustness to length modifications.

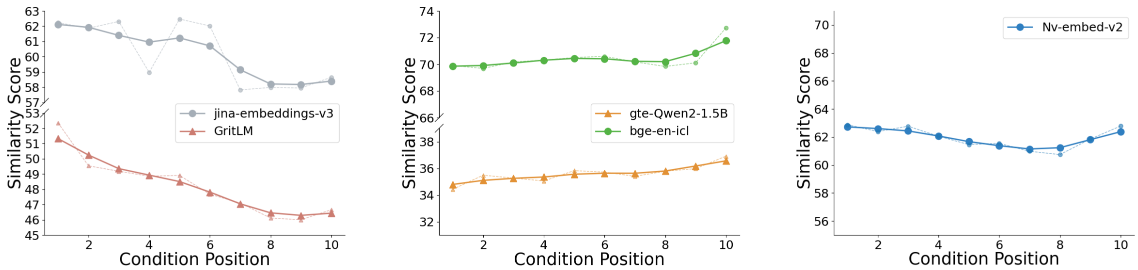
5.2 Condition Position Impact on Focus

Our experimental findings indicate that the position of a condition significantly influences the model's subsequent similarity judgment. This phenomenon



(a) Retrieval Performance when padding to 512 words (b) Retrieval Performance when padding to 512 words

Figure 5: Retrieval performance when padding the document set to 512 words and 1024 words. Rerankers are highly sensitive to increases in document length, showing rapid performance degradation, whereas retrievers remain comparatively robust.



(a) Mean pooling (b) <EOS> pooling (c) Latent attention layer pooling

Figure 6: Impact of condition position on different pooling methods. The condition is placed at different positions (1–10) in the query, with other positions filled by ten "[unused0]" tokens (example from Table 8). The dashed line represents the original data, while the solid line shows the Gaussian-smoothed trend (kernel size = 1) for clarity.

was observed consistently across both retriever and reranker models.

To illustrate position effects in retriever models, we conducted a targeted study on how relocating a single condition within the query influences its similarity score. We selected representative retriever models employing distinct pooling strategies: mean pooling, <EOS> pooling, and latent layer pooling. The results, as depicted in Fig.6, revealed that models utilizing mean pooling tend to weight the early tokens most heavily: similarity drops steadily as the condition is shifted toward the tail of the query. <EOS> shows the opposite bias, emphasising the final tokens. Latent layer pooling heightened focus on both the beginning and end of the query, with comparatively less focus on the middle.

Similarly, for reranker models, we selected a cross-encoder model (bge-reranker-m3) to visualize the attention heatmap. Fig.7 in Appendix F.4 shows a non-uniform distribution of attention across different token positions. This implies that these rerankers tend to assign differential focus to specific conditions or tokens within the concatenated query-document input, rather than distributing their attention uniformly across all elements.

6 Conclusion

In this work, we introduced MULTICONIR, a novel benchmark designed to rigorously evaluate information retrieval models in realistic multi-condition scenarios, a critical area where existing evaluation frameworks are lacking. Through three specifically designed tasks—complexity robustness, relevance monotonicity, and query format sensitivity—conducted across five diverse domains. Experiments revealed that existing models struggle with multi-condition retrieval, with their performance degrading as the number of conditions increases; rerankers excel for single-condition queries but fail in multi-condition scenarios. Notably, rerankers are more sensitive to complex inputs. GPT-4o outperforms specialised IR systems, exposing a performance gap in handling complex information needs.

Our findings highlight an urgent need for new modeling approaches and training paradigms specifically tailored for robust multi-condition understanding. MULTICONIR serves as a valuable resource to drive this research, benchmark progress, and ultimately propel information retrieval systems towards a more sophisticated, human-like comprehension of complex information needs.

Limitations

While MULTICONIR provides a novel benchmark for evaluating retrieval models in multi-condition scenarios, several limitations should be acknowledged. First, our dataset relies on automated query generation and hard negative creation, which may introduce biases in condition representation despite efforts to ensure accuracy. These biases could affect retrieval models' ability to distinguish fine-grained differences. Second, our evaluation focuses on retrieval tasks and does not cover reasoning-based retrieval or interactive search scenarios. Real-world systems often incorporate reranking, user feedback, and hybrid retrieval, which are not explicitly modeled. Lastly, our dataset does not fully consider query reformulation strategies or multi-turn retrieval, limiting its applicability to dynamic search environments. These limitations highlight the need for further research into multi-condition retrieval, particularly in addressing dataset biases, expanding evaluation scopes, and integrating retrieval with realistic user interactions.

Ethics Statement

This study adheres to ethical standards in AI research, ensuring transparency and reproducibility in dataset construction and model evaluation while exclusively using publicly available pre-trained models for experiments. Dataset Considerations: MULTICONIR is built from publicly available sources and does not contain sensitive or personally identifiable information. Given its inclusion of medical and legal documents, we apply strict data filtering and safety measures to respect model safety constraints and prevent the generation of harmful or misleading content. Additionally, we recognize that automatically generated queries and hard negatives may introduce biases. Therefore, during dataset construction, we take measures to minimize the impact of inherent language model biases on retrieval tasks. MultiConIR aims to advance multi-condition retrieval research while ensuring data fairness and ethical compliance. We encourage future research to further explore bias detection strategies in retrieval dataset, enhancing model fairness and reliability in diverse corpus environments.

References

Akari Asai, Timo Schick, Patrick Lewis, Xilun Chen, Gautier Izacard, Sebastian Riedel, Hannaneh Hajishirzi, and Wen-tau Yih. 2023. Task-aware retrieval with instructions. *ACL Findings*.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. Llm2vec: Large language models are secretly powerful text encoders. *Preprint*, arXiv:2404.05961.

Adam Berger, Rich Caruana, David Cohn, Dayne Freitag, and Vibhu Mittal. 2000. Bridging the lexical chasm: statistical approaches to answer-finding. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 192–199.

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

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2022. LexGLUE: A benchmark dataset for legal language understanding in english. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4310–4330. Association for Computational Linguistics.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *Preprint*, arXiv:2402.03216.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Thomas Palmeira Ferraz, Kartik Mehta, Yu-Hsiang Lin, Haw-Shiuan Chang, Shereen Oraby, Sijia Liu, Vivek Subramanian, Tagyoung Chung, Mohit Bansal, and Nanyun Peng. 2024. Llm self-correction with decrim: Decompose, critique, and refine for enhanced following of instructions with multiple constraints. *arXiv preprint arXiv:2410.06458*.

Qianyu He, Jie Zeng, Qianxi He, Jiaqing Liang, and Yanghua Xiao. 2024. From complex to simple: Enhancing multi-constraint complex instruction following ability of large language models. *arXiv preprint arXiv:2404.15846*.

HPE AI Solutions. 2023. Medical cases classification tutorial dataset.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2024. Nv-embed: Improved techniques for training llms as generalist embedding models. *arXiv preprint arXiv:2405.17428*.

Chaofan Li, Zheng Liu, Shitao Xiao, and Yingxia Shao. 2023a. Making large language models a better foundation for dense retrieval. *Preprint*, arXiv:2312.15503.

Chaofan Li, MingHao Qin, Shitao Xiao, Jianlyu Chen, Kun Luo, Yingxia Shao, Defu Lian, and Zheng Liu. 2024. Making text embedders few-shot learners. *Preprint*, arXiv:2409.15700.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023b. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Zhuoyan Li, Hangxiao Zhu, Zhuoran Lu, and Ming Yin. 2023c. Synthetic data generation with large language models for text classification: Potential and limitations. *arXiv preprint arXiv:2310.07849*.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2023. Fine-tuning llama for multi-stage text retrieval. *Preprint*, arXiv:2310.08319.

Sameer S. Mahajan. 2017. People wikipedia data.

Quan Mai, Susan Gauch, and Douglas Adams. 2024. Setbert: Enhancing retrieval performance for boolean logic and set operation queries. *arXiv preprint arXiv:2406.17282*.

Niklas Muennighoff, Hongjin Su, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. 2024. Generative representational instruction tuning. *Preprint*, arXiv:2402.09906.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

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.

Jinming Nian, Zhiyuan Peng, Qifan Wang, and Yi Fang. 2024. W-RAG: Weakly Supervised Dense Retrieval in RAG for Open-domain Question Answering. *arXiv preprint arXiv:2408.08444*.

Hanseok Oh, Hyunji Lee, Seonghyeon Ye, Haebin Shin, Hansol Jang, Changwook Jun, and Minjoon Seo. 2024. Instructir: A benchmark for instruction following of information retrieval models. *arXiv preprint arXiv:2402.14334*.

OpenAI. 2024. Hello gpt-4o. https://openai.com/index/hello-gpt-4o/. Accessed: 2025-05-19.

Jay M Ponte and W Bruce Croft. 2017. A language modeling approach to information retrieval. In *ACM SIGIR Forum*, 2, pages 202–208. ACM New York, NY, USA.

Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023. Rankzephyr: Effective and robust zero-shot listwise reranking is a breeze! *Preprint*, arXiv:2312.02724.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. 2024. Infobench: Evaluating instruction following ability in large language models. *arXiv preprint arXiv:2401.03601*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

Jonathan Robischon. 2018. Wikipedia movie plots.

Elvin Rustamov. 2021. Books dataset.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. 2024. Ai models collapse when trained on recursively generated data. *Nature*, 631(8022):755–759.

Saba Sturua, Isabelle Mohr, Mohammad Kalim Akram, Michael Günther, Bo Wang, Markus Krimmel, Feng Wang, Georgios Mastrapas, Andreas Koukounas, Nan Wang, et al. 2024. jina-embeddings-v3: Multilingual embeddings with task lora. *arXiv preprint arXiv:2409.10173*.

Hongjin Su, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han-yu Wang, Haisu Liu, Quan Shi, Zachary S Siegel, Michael Tang, et al. 2024. Bright: A realistic and challenging benchmark for reasoning-intensive retrieval. *arXiv preprint arXiv:2407.12883*.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2023. Improving text embeddings with large language models. *arXiv preprint arXiv:2401.00368*.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024a. Improving text embeddings with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11897–11916, Bangkok, Thailand. Association for Computational Linguistics.

Yifei Wang, Jizhe Zhang, and Yisen Wang. 2024b. Do generated data always help contrastive learning? *Preprint*, arXiv:2403.12448.

Orion Weller, Benjamin Chang, Sean MacAvaney, Kyle Lo, Arman Cohan, Benjamin Van Durme, Dawn Lawrie, and Luca Soldaini. 2024a. Followir: Evaluating and teaching information retrieval models to follow instructions. *arXiv preprint arXiv:2403.15246*.

Orion Weller, Dawn Lawrie, and Benjamin Van Durme. 2024b. Nevir: Negation in neural information retrieval. *Preprint*, arXiv:2305.07614.

Orion Weller, Benjamin Van Durme, Dawn Lawrie, Ashwin Paranjape, Yuhao Zhang, and Jack Hessel. 2024c. Promptriever: Instruction-trained retrievers can be prompted like language models. *arXiv preprint arXiv:2409.11136*.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-pack: Packaged resources to advance general chinese embedding. *Preprint*, arXiv:2309.07597.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1503–1512.

Wenhao Zhang, Mengqi Zhang, Shiguang Wu, Jiahuan Pei, Zhaochun Ren, Maarten de Rijke, Zhumin Chen, and Pengjie Ren. 2024a. Excluir: Exclusionary neural information retrieval. *arXiv preprint arXiv:2404.17288*.

Xinghua Zhang, Haiyang Yu, Cheng Fu, Fei Huang, and Yongbin Li. 2024b. Iopo: Empowering llms with complex instruction following via input-output preference optimization. *arXiv preprint arXiv:2411.06208*.

Zongmeng Zhang, Jinhua Zhu, Wengang Zhou, Xiang Qi, Peng Zhang, and Houqiang Li. 2024c. Boolquestions: Does dense retrieval understand boolean logic in language? *Preprint*, arXiv:2411.12235.

Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Haonan Chen, Zheng Liu, Zhicheng Dou, and Ji-Rong Wen. 2023. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107*.

A Details of Models

For each model used in this paper, Table 3 provides details on model size, architecture, maximum input context length, and whether instructions is included. The GPT-4o model utilized for dataset generation and ranking in this work was the "2024-07-01-preview" version.

Model	Size	Architecture	Instruction	Max length	Pooling Method
Sparse Retriever					
BM25 (Robertson and Zaragoza, 2009)	N/A	Sparse	No	N/A	N/A
Dense Retriever					
jina-embeddings-v3 (Sturua et al., 2024)	572M	Encoder	No	4K	Mean
gte-large-en-v1.5 (Li et al., 2023b)	434M	Encoder	No	8K	<GLS>
NV-Embed-v2 (Lee et al., 2024)	7.8B	Decoder	Yes	32K	Latent Attention Layer
bge-en-icl (Li et al., 2024)	7.1B	Decoder	Yes	32K	<EOS>
gte-Qwen2-7B-instruct (Li et al., 2023b)	7.6B	Decoder	Yes	131K	<EOS>
gte-Qwen2-1.5B-instruct (Li et al., 2023b)	1.5B	Decoder	Yes	131K	<EOS>
e5-mistral-7b-instruct (Wang et al., 2024a)	7.1B	Decoder	Yes	32K	<EOS>
GritLM-7B (Muennighoff et al., 2024)	7.2B	Decoder	Yes	4K	Mean
LLM2Vec (BehnamGhader et al., 2024)	7.5B	Decoder	Yes	8K	Mean
Fine-tuned Reranker					
bge-reranker-v2-m3 (Chen et al., 2024)	568M	Cross-Encoder	No	8k	N/A
bge-reranker-v2-gemma (Chen et al., 2024)	2.5B	Decoder	Yes	8K	N/A
followIR (Weller et al., 2024a)	7.2B	Decoder	Yes	4K	N/A
RankZephyr (Pradeep et al., 2023)	7B	Decoder	Yes	32K	N/A
Zero-shot LLM for Ranking					
GPT-4o (OpenAI, 2024)	N/A	Decoder	Yes	128K	N/A

Table 3: **Details of models used in experiments.** We list the number of parameters of each model except the sparse model (BM25). Regarding the model architecture, we distinguish between sparse models, dense models, and rerankers. Dense models are further classified as Encoders or Decoders. Rerankers are categorized into Cross Encoders and Decoders (LLM-based generative relevance scoring). The Instruction column indicates whether instructions are included in the retrieval process. Max length denotes the maximum input length used for each model in the experiments. The Pooling Method represents the approach used by the model to obtain embeddings.

For Dense Retrieval models that require instructions (NV-Embed-v2, bge-en-icl, gte-Qwen2-7B-instruct, gte-Qwen2-1.5B-instruct, e5-mistral-7b-instruct, GritLM-7B, and LLM2Vec), we use the following instruction:

"Given a domain retrieval query, retrieve documents that meet the specified conditions."

For LLM-based rerankers (bge-reranker-v2-gemma and followIR), we adopt the model's default prompt. For example, bge-reranker-v2-gemma uses the following prompt:

"Given a query A and a passage B, determine whether the passage contains an answer to the query by providing a prediction of either 'Yes' or 'No'."

For models that do not require instructions, we directly input the query and document, such as jina-embeddings-v3, gte-large-en-v1.5, and bge-reranker-v2-m3.

B Prompt Templates For Constructing MULTICONIR DATASET

Table 4, 5, and 6 present the prompts used in Steps 1 to 3 for constructing our **MULTICONIR** dataset.

For placeholders, {*domain*} ∈ {*People, Books, Movies, Medical Case, Legal Document*}. {*domain_features*} specifies key attributes within a particular domain. In the medical case domain, {*domain_features*} ∈ { *patient symptoms, clinical diagnosis, drug allergies, family medical history, surgical details, postoperative outcomes, hospitalization duration, recovery status.*} In the legal document domain, {*domain_features*} ∈ { *case type, involved parties, court ruling, legal provisions, evidence summary, defense strategy.* } In the movies domain, {*domain_features*} ∈ { *summary, lead actors, release date, release area, genre, detailed plots.* } In the books domain, {*domain_features*} ∈ { *author, publication* *year, genre, main content, detailed plots.* } In the people domain, {*domain_features*} $\in$ { *profession, nationality, notable achievements, social impact, related events.* }

Task	Prompt
Step 1: Condition Sentence Extraction	I will provide you a document of {domain}, you should extract ten detailed sentences that represent the key conditions the document satisfies. Please adhere to the following guidelines: - Extract fine-grained condition-related sentences relevant to {domain}, such as {domain_features}. - Do not paraphrase; use the original sentences from the document. - Ensure each sentence is semantically intact and not conflict with the context. - Format the output as an array, e.g., ["sentence1", "sentence2", ..., "sentence10"]. Here is the document: {domain_document}. Return array only.

Table 4: Prompt for GPT-4o to extract condition sentence (Step 1).

C Benchmark Quality Evaluation

To guarantee the reliability of MULTICONIR, we applied a two–stage audit that examines both *query realism* and *document–label validity*. Figure 3 gives a visual outline; full numbers appear in Table 1.

Query realism. From each of the five domains (*People*, *Books*, *Movies*, *Medical*, *Legal*) we randomly sampled 100 multi-condition queries, yielding a 500-item evaluation set. Ten trained graduate annotators independently rated every query for *linguistic naturalness*, *precision of constraints*, and *contextual plausibility* (realistic vs. unrealistic). Inter-annotator agreement reached 93.7% with Fleiss' $\kappa = 0.84$, indicating near-perfect consensus that the automatically generated queries resemble genuine information needs.

Document–label validity.

1. **LLM filtering.** We applied *GPT-4o* to the *entire* corpus. For every positive document d^+ the model verified that all ten conditions in $C(d)$ were satisfied; for each hard-negative document d_k^- it checked that *exactly* $k-1$ conditions held. Instances failing these criteria (false positives or false negatives) were discarded, reducing domain sizes to: People (420), Books (482), Movies (500), Medical (479), Legal (426).

2. **Human spot-check.** To confirm the LLM filter, we randomly drew another 100 document–query pairs per domain (500 in total). Two independent annotators judged whether the labelled number of satisfied conditions was correct; disagreements were resolved by a third adjudicator. The residual error rate was 2.4%, implying that the automatic filter removed the vast majority of mis-labelled items.

D Discussion: *Win Rate* vs. Traditional IR Metrics

Why introduce Win Rate? MULTICONIR poses *multi-condition* queries for which models must sense fine-grained semantic differences. We focus on two abilities: **(i)** discriminating the *positive* document from a hard negative as the number of query conditions grows (**Task 1**); **(ii)** preserving a *monotonic ordering* in which a document satisfying k conditions outranks one satisfying $k-1$ under the *same* query (**Task 2**).

Task	Prompt
Step 2: Query Generation (Instruction-style)	I will provide you {num} condition-related sentences; formulate a retrieval query for me. Here are a few examples for reference: - {Instruction-style example 1} - {Instruction-style example 2} Please adhere to the following guidelines: - Each sentence represents a condition; with {num} sentences, the number of conditions is {num}. - The query should be instruction-style, explicitly listing all conditions. - Each condition should be around 10 words. - Make conditions concise, summarizing each sentence. - You can paraphrase and modify keywords while maintaining meaning. Here are the sentences: {info}. Return one query only. Do not include extra information.
Step 2: Query Generation (Descriptive-style)	I will provide you {num} condition-related sentences; formulate a retrieval query for me. Here are a few examples for reference: - {Descriptive-style example 1} - {Descriptive-style example 2} Please adhere to the following guidelines: - Each sentence represents a condition; with {num} sentences, the number of conditions is {num}. - The query should be descriptive-style, integrating and describing all conditions in natural language. - Each condition should be around 10 words. - Make conditions concise, summarizing each sentence. - You can paraphrase and modify keywords while maintaining meaning. Here are the sentences: {info}. Return one query only. Do not include extra information.

Table 5: Prompt for GPT-4o to generate queries with varying conditions (Step 2).

Limitations of conventional metrics.

- **Precision@1** coincides with Win Rate in Task 1 (one positive vs. one HN) but, in Task 2, observes only the *top* result and ignores the intended hierarchy $d^+ \succ \mathrm{HN}_1 \succ \cdots \succ \mathrm{HN}_{10}$.
- **NDCG@k** introduces graded relevance but still weights absolute rank more than pairwise consistency, thus blurring step-wise violations of the monotonic order.
- **Recall** proved even less informative in early pilot runs dominated by easy negatives: high recall was achievable without respecting the semantic precision that MULTICONIR is designed to test.

How Win Rate fills the gap. Win Rate computes the proportion of pairwise comparisons in which a document that fulfils more conditions is ranked above one that fulfils fewer. Hence, it *matches Precision@1* in the degenerate Task 1 case, yet remains sensitive to every local inversion in the graded Task 2 ladder, offering a sharper lens on a model's ability to capture incremental semantic constraints.

Task	Prompt
Step 3: Hard Negative Sentence Making (For Books, Movies, Medical Case, and Legal Document Datasets)	I will provide you one query and one sentence, generate a modified sentence for me. Here are a few examples for reference: Query: - {query} Sentence: - {condition sentence} Modified: - {hard negative sentence} Please adhere to the following guidelines: - Modify the sentence so that its meaning no longer aligns with the query. - Keep key terms unchanged. - Ensure the new sentence is semantically different from the original. Here is the query: {query}. Here is the Sentence: {information}. Return only the modified sentence.
Step 3: Hard Negative Sentence Making (For People Dataset)	I will provide you one query and one sentence, generate a modified sentence for me. Here are a few examples for reference: Query: - Who is the American artist that went to RISD? Sentence: - He went to RISD for graduate school. Modified: - He went to ACCA for graduate school, but his sister went to RISD. Please adhere to the following guidelines: - Modify the sentence so that its meaning no longer aligns with the query. - Keep key terms unchanged, but introduce dummy information to mislead the retrieval model. For example, if the original sentence states, “He went to RISD for graduate school,” you can modify it to, “He went to ACCA for graduate school, but his sister went to RISD,” where the key term (RISD) remains but is assigned to an irrelevant entity (his sister). - Ensure the new sentence is semantically different from the original by using different wording and synonymous substitution. - The changed sentence should prevent the query from retrieving it as relevant information. Here is the query: {query}. Here is the sentence: {information}. Return only the modified sentence.

Table 6: Prompt for GPT-4o to modify the condition sentence to hard negative sentence (Step 3).

E Examples Of The MULTICONIR Dataset

Tables 8, 9, 7, 10, and 11 illustrate examples from their respective domains.

F Complete Results

F.1 Complete Results Of Task 2

Table 12 presents the experimental results of Task 2, where Win Rate reflects the success rate between documents that satisfy different numbers of conditions under a multi-condition query (query10, which

Query 3	Positive	HN 1
Find a notable individual who meets these criteria: 1. Studies plasma melatonin to assess biological rhythm disorders. 2. Identified 25-hour circadian rhythms in totally blind individuals. 3. **Worked at NIMH in Bethesda, Maryland before 1981.**	Relying on a very precise assay for plasma melatonin, a hormone that has a clearly defined 24-hour pattern of secretion, biological rhythm disorders can be assessed and their treatment can be monitored. Totally blind individuals have 25-hour circadian rhythms, drifting an hour later each day unless they take a melatonin capsule at a certain time every day. **Prior to moving to Oregon in 1981, Lewy was at the National Institute of Mental Health (NIMH) in Bethesda, Maryland, working with senior colleague Thomas Wehr.** In Oregon, he has worked closely with Robert L. Sack. He describes his research as follows: 'My laboratory studies chronobiologic sleep and mood disorders.' Alfred J. Lewy, aka Sandy Lewy, graduated from University of Chicago in 1973 after studying psychiatry, pharmacology, and ophthalmology. As of December 2005, he had 94 publications available on PubMed. He is a full professor and vice-chair of the Department of Psychiatry at OHSU (Oregon Health Science University) and holds an MD and PhD. Current research is focused on developing bright light exposure and melatonin administration as treatment modalities for these disorders. These disorders include winter depression, jet lag, maladaptation to shift work, and certain types of sleep disturbances.	Relying on a very precise assay for plasma melatonin, a hormone that has a clearly defined 24-hour pattern of secretion, biological rhythm disorders can be assessed and their treatment can be monitored. Totally blind individuals have 25-hour circadian rhythms, drifting an hour later each day unless they take a melatonin capsule at a certain time every day. After to moving to Oregon in 1981, Lewy was at the National Institute of Mental Health (NIMH) in Bethesda, Maryland, working with senior colleague Thomas Wehr. In Oregon, he has worked closely with Robert L. Sack. He describes his research as follows: 'My laboratory studies chronobiologic sleep and mood disorders.' Alfred J. Lewy, aka Sandy Lewy, graduated from University of Chicago in 1973 after studying psychiatry, pharmacology, and ophthalmology. As of December 2005, he had 94 publications available on PubMed. He is a full professor and vice-chair of the Department of Psychiatry at OHSU (Oregon Health Science University) and holds an MD and PhD. Current research is focused on developing bright light exposure and melatonin administration as treatment modalities for these disorders. These disorders include winter depression, jet lag, maladaptation to shift work, and certain types of sleep disturbances.

Table 7: An example in domain of People

contains ten conditions), i.e., d_k vs. d_{k-1}.

F.2 Complete Results Of Task3

Table 13 presents the complete results of Format Invariance.

Query 5	Positive	HN 1
Find a notable individual who meets these criteria: 1. Details Bernie Madoff's $65B Ponzi collapse. 2. Covers the impact on national media. 3. Investigates Madoff's history of fraud. 4. Offers deep insight into Madoff's family. 5. **Contains exclusive news and material.**	The collapse of Bernie Madoff's Ponzi scheme led to the instant evaporation of $65 billion of wealth. The effects of Madoff's brazen fraud were felt most closely in New York and Palm Beach but the story was, and continues to be, front page news across the country. Brian Ross and his team of investigators shed an unyielding light onto Madoff's scheme–how he got started, how he succeed for so long, who helped him, and who shielded him from early investigations. This is an incisive and voyeuristic look into this first family of financial crime. **The Madoff Chronicles includes a vast array of news and material that readers won't find anywhere else.** Contains a reproduction of Bernie's Little Black Book. Ross has also secured Madoff's calendar for the past three years and other never-before-seen documents from inside the Madoff empire, straight from his desk. Read key details of how Madoff carried out his scam and the revelation that he began the fraud from almost the first day, in the 1960s. Extensive cooperation by Madoff's personal assistant, Eleanor Squillari. Contains incriminating connections between Madoff and certain members of the SEC.	The collapse of Bernie Madoff's Ponzi scheme led to the instant evaporation of $65 billion of wealth. The effects of Madoff's brazen fraud were felt most closely in New York and Palm Beach but the story was, and continues to be, front page news across the country. Brian Ross and his team of investigators shed an unyielding light onto Madoff's scheme–how he got started, how he succeed for so long, who helped him, and who shielded him from early investigations. This is an incisive and voyeuristic look into this first family of financial crime. The Madoff Chronicles includes a vast array of news and material. Contains a reproduction of Bernie's Little Black Book. Ross has also secured Madoff's calendar for the past three years and other never-before-seen documents from inside the Madoff empire, straight from his desk. Read key details of how Madoff carried out his scam and the revelation that he began the fraud from almost the first day, in the 1960s. Extensive cooperation by Madoff's personal assistant, Eleanor Squillari. Contains incriminating connections between Madoff and certain members of the SEC.

Table 8: An example in domain of Book

F.3 Complete Results Of Document Length

Table 14 and Table 15 present the effect of document length on retrieval performance, with documents padded to 512 and 1024 words, respectively. We use repeated filler text, such as *"The grass is green. The sky is blue. The sun is yellow. Here we go. There and back again."*, following the setting in Wang et al. (2023). The filler text is inserted between the original document sentences until the total text length reaches 512 or 1024 words.

Query 8	Positive	HN 1
Find a movie that matches all conditions: 1. Originated from American. 2. Plot: Mrs. Lowe and Black were lovers. 3. Plot: Terry carelessly spends money. 4. Plot: Terry promoted to ship foreman. 5. Cast: Mary Miles Minter, Allan Forrest. 6. Plot: Julia Deep works behind exchange desk. 7. Director: Lloyd Ingraham. 8. **Plot: Terry's spending takes a toll.**	Origin/Ethnicity: American Meanwhile, it is revealed Mrs. Lowe and Black were once lovers. He is spending his money carelessly and doesn't put any time in paying the bills, much to the dislike of the department store owner Timothy Black. Soon, Terry is promoted to a foreman on a ship. Cast: Mary Miles Minter, Allan Forrest Julia Deep is a young woman working behind the exchange desk at a department store. Director: Lloyd Ingraham **After a while, Terry's money spending takes its toll.** Lottie gets distracted and does not notice Terry and Julia at the park. Release Year: 1918	Origin/Ethnicity: American Meanwhile, it is revealed Mrs. Lowe and Black were once lovers. He is spending his money carelessly and doesn't put any time in paying the bills, much to the dislike of the department store owner Timothy Black. Soon, Terry is promoted to a foreman on a ship. Cast: Mary Miles Minter, Allan Forrest Julia Deep is a young woman working behind the exchange desk at a department store. Director: Lloyd Ingraham **Eventually, Terry's frugality leads to financial growth.** Lottie gets distracted and does not notice Terry and Julia at the park. Release Year: 1918

Table 9: An example in domain of Movie

F.4 Attention heatmap of cross-encoder model

Figure 7 shows the attention-score heat map produced by the cross-encoder reranker (bge-reranker-m3) for the input query–document pair. As we progressively add conditions to the query and compute the attention distribution between each condition and its corresponding segment in the document, we observe that the cross-encoder allocates attention unevenly across positions. Fig.7

G Findings In Constructing MULTICONIR Dataset

G.1 The Use Of LLM-generated Data in Retrieval

In recent years, artificial datasets generated by LLMs have become a common practice for training and evaluating retrieval models (Su et al., 2024; Lee et al., 2024; Weller et al., 2024a). For instance, E5-Mistral (Wang et al., 2024a) rely entirely on LLM-generated datasets for fine-tuning. While this approach can significantly expand training corpora, prior studies have highlighted its potential drawbacks, including introducing inherit linguistic biases of the underlying LLMs (Shumailov et al., 2024), potentially constraining the retrieval model's performance and generalizability. Furthermore, purely artificial data often lacks the contextual richness and complexity found in real-world retrieval scenarios (Li et al., 2023c; Wang et al., 2024b), making it difficult to capture the actual needs of users' queries accurately.

During our dataset construction, we observed similar issues. When using LLM-generated transformations to modify positive documents into hard negatives, the model often restructured expressions to fit its learned patterns, even when explicitly instructed to modify only a few condition-related words while keeping the rest unchanged. For example, in the legal documents dataset, a positive sentence like: *"The defendant was convicted of fraud under Section 420 of the Penal Code and sentenced to five years in prison."* was frequently modified by the LLM into a generic pattern, such as: *"The defendant was found guilty of fraud and received a prison sentence."*

Similarly, in medical case documents, a sentence like: *"The patient reported experiencing persistent*

Query 7	Positive	HN 1
Find a case where the patient: 1. Underwent ascending aortic arch angiogram. 2. Had left common carotid artery angiogram. 3. Received right common carotid artery angiogram. 4. Undergone left subclavian artery angiogram. 5. Had right iliac angiogram with runoff. 6. Performed bilateral cerebral angiograms. 7. **Experienced TIA and moderate carotid stenosis.**	PROCEDURE PERFORMED: 1. Selective ascending aortic arch angiogram. 2. Selective left common carotid artery angiogram. 3. Selective right common carotid artery angiogram. 4. Selective left subclavian artery angiogram. 5. Right iliac angio with runoff. 6. Bilateral cerebral angiograms were performed as well via right and left common carotid artery injections. **INDICATIONS FOR PROCEDURE: TIA, aortic stenosis, postoperative procedure. Moderate carotid artery stenosis.** ESTIMATED BLOOD LOSS: 400 ml. After obtaining informed consent, the patient was brought to the cardiac catheterization suite in postabsorptive and nonsedated state. Using modified Seldinger technique, a 6-French sheath was placed into the right common femoral artery and vein without complication.	PROCEDURE PERFORMED: 1. Selective ascending aortic arch angiogram. 2. Selective left common carotid artery angiogram. 3. Selective right common carotid artery angiogram. 4. Selective left subclavian artery angiogram. 5. Right iliac angio with runoff. 6. Bilateral cerebral angiograms were performed as well via right and left common carotid artery injections. INDICATIONS FOR PROCEDURE: TIA, aortic stenosis, postoperative procedure. Severe carotid artery stenosis. ESTIMATED BLOOD LOSS: 400 ml. After obtaining informed consent, the patient was brought to the cardiac catheterization suite in postabsorptive and nonsedated state. A 6-French sheath was used in the left femoral artery and vein with minor complications, employing the modified Seldinger technique.

Table 10: An example in domain of Medical Case

chest pain and shortness of breath, leading to a diagnosis of angina." was often transformed into a standardized version: *"The patient was diagnosed with a heart condition after reporting chest pain."*

These modifications eroded the diversity and long-tail characteristics of real-world data, reducing the fine-grained variability necessary for retrieval tasks. Instead of preserving rich domain-specific details, LLM-generated transformations tended to normalize distinct cases into overly generic patterns, which could misrepresent real-world retrieval challenges.

Empirical results further confirm the limitations of fully LLM-generated training data. The E5-Mistral model, which relies entirely on synthetic data, performs the worst on MULTICONIR. In Task 1, as shown in Table 2, it exhibits the highest performance decline (16.93%) among retrieval models, and in Task 2, as shown in Table 12, its average win rate (60.36%) is the lowest among retrieval models, trailing the second-worst model (Jina-Embeddings-V2) by 5%. These results reinforce the generalization challenges posed by fully synthetic datasets in retrieval tasks, highlighting the importance of incorporating real-world document structures and constraints in training data.

To mitigate this, our pipeline minimizes document-wide modifications, instead restricting LLM interventions to condition sentences only. This targeted approach preserves real-world data authenticity while introducing controlled semantic perturbations, ensuring that retrieval models are trained on meaningful and realistic hard negatives rather than fully synthetic documents.

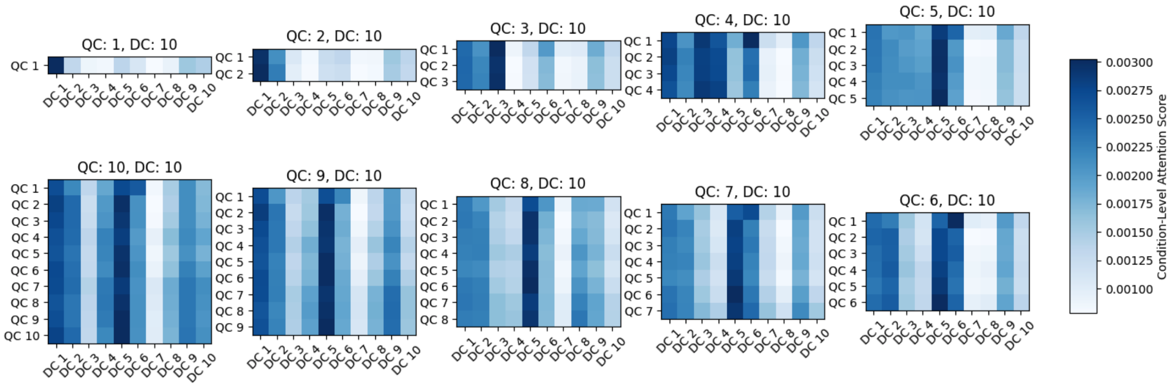


Figure 7: Attention heatmap of cross-encoder model (bge-reranker-m3).

G.2 Impact of Different Hard Negative Construction Strategies

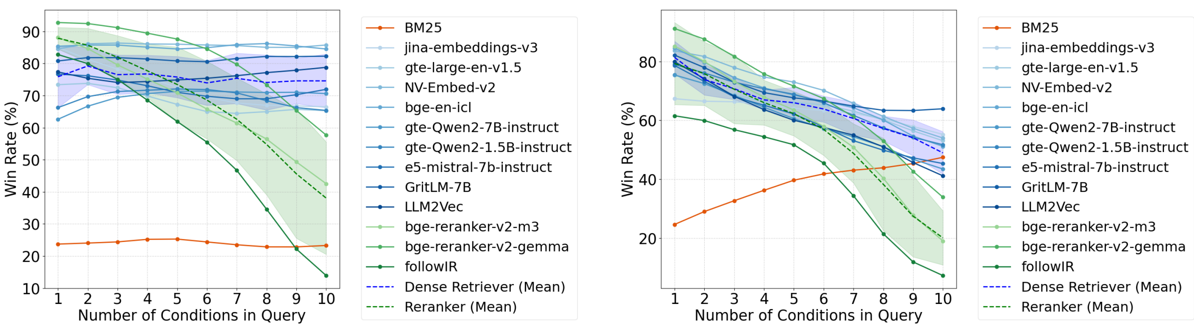
To systematically examine the impact of hard negative sentence (HNS) construction on retrieval models, we experimented with two distinct approaches: (1) Key Information Modification – altering critical details while maintaining overall sentence structure (applied to books, movies, medical cases, and legal documents). (2) Keyword Retention with Dummy Information – keeping all original keywords intact while injecting irrelevant dummy information (used for the people dataset).

A key objective of this study was to investigate how different HNS construction strategies affect retrieval difficulty. Our initial hypothesis was that the second approach (retaining keywords but adding dummy information) would pose a greater challenge for retrieval models, particularly Dense Retrievers, since hard negatives in this setting contain all the key terms present in positive documents.

However, our experimental results contradicted this expectation. As shown in Fig.8 on the people dataset, Dense Retrieval models remained highly stable, demonstrating a strong ability to differentiate semantic nuances even when all keywords were retained. This suggests that Dense Retrieval primarily relies on contextual embeddings rather than simple keyword matching, allowing it to distinguish between truly relevant documents and distractors with superficial lexical overlap.

In contrast, Reranker models exhibited a significant performance drop when dealing with dummy-information-based HNS. This suggests that Rerankers are more sensitive to this type of negative construction, likely due to their cross-encoder or generative architectures, which process both the query and document jointly. Since Rerankers typically assign scores based on fine-grained textual relevance, the presence of keyword overlap without genuine semantic alignment may mislead them more than Dense Retrieval models.

These findings highlight important considerations for hard negative sampling in multi-condition retrieval. While Dense Retrievers appear robust to surface-level keyword retention, Rerankers are more vulnerable to semantically misleading negatives, suggesting that future retrieval pipelines should adapt negative sampling strategies based on the target retrieval model architecture.



(a) Task1 on Peole dataset

(b) Task1 on Legal dataset

Figure 8: Impact of different HNS construction strategies.

Query 10	Positive	HN 1
Find a case where: 1. Michigan Legislature enacted a statute in 1987. 2. Petitioners challenged the statute under Contract Clause and Due Process Clause. 3. The statute affected workers injured before March 31, 1982. 4. Petitioners argued a 1981 law allowed reduction of workers' compensation benefits. 5. The Michigan Supreme Court accepted petitioners' interpretation in 1985. 6. Legislature introduced a bill to overturn the court's decision. 7. House Bill 5084 was introduced in October 1985. 8. The bill became law on May 14, 1987. 9. Petitioners were ordered to refund nearly $25 million. 10. **Michigan Supreme Court upheld the statute for lacking vested rights and rational purpose.**	In 1987, the Michigan Legislature enacted a statute that had the effect of requiring petitioners General Motors Corporation (GM) and Ford Motor Company (Ford) to repay workers' compensation benefits GM and Ford had withheld in reliance on a 1981 workers' compensation statute. Petitioners challenge the provision of the statute mandating these retroactive payments on the ground that it violates the Contract Clause and the Due Process Clause of the Federal Constitution. The benefit coordination provision did not specify whether it was to be applied to workers injured before its effective date, March 31, 1982. Petitioners took the position that the 1981 law allowed them to reduce workers' compensation benefits to workers injured before March 31, 1982, who were receiving benefits from other sources. In 1985, petitioners' interpretation was accepted by the Michigan Supreme Court. Chambers v. General Motors Corp., decided together with Franks v. White Pine Copper Div., Copper Range Co., 422 Mich. 636, 375 N.W.2d 715. The Michigan Legislature responded almost immediately by introducing legislation to overturn the court's decision. On October 16, 1985, before the Michigan Supreme Court had ruled on the motion for rehearing in Chambers, House Bill 5084 was introduced. The amended Senate bill passed into law on May 14, 1987. 1987 Mich.Pub.Acts No. 28. As a result of the 1987 statute, petitioners were ordered to refund nearly $25 million to disabled employees. **The Michigan Supreme Court upheld the statute against these challenges, on the ground that the employers had no vested rights in coordination for Contract Clause purposes, and that the retroactive provisions furthered a rational legislative purpose.** 436 Mich. 515, 462 N.W.2d 555 (1990).	In 1987, the Michigan Legislature enacted a statute that had the effect of requiring petitioners General Motors Corporation (GM) and Ford Motor Company (Ford) to repay workers' compensation benefits GM and Ford had withheld in reliance on a 1981 workers' compensation statute. Petitioners challenge the provision of the statute mandating these retroactive payments on the ground that it violates the Contract Clause and the Due Process Clause of the Federal Constitution. The benefit coordination provision did not specify whether it was to be applied to workers injured before its effective date, March 31, 1982. Petitioners took the position that the 1981 law allowed them to reduce workers' compensation benefits to workers injured before March 31, 1982, who were receiving benefits from other sources. In 1985, petitioners' interpretation was accepted by the Michigan Supreme Court. Chambers v. General Motors Corp., decided together with Franks v. White Pine Copper Div., Copper Range Co., 422 Mich. 636, 375 N.W.2d 715. The Michigan Legislature responded almost immediately by introducing legislation to overturn the court's decision. On October 16, 1985, before the Michigan Supreme Court had ruled on the motion for rehearing in Chambers, House Bill 5084 was introduced. The amended Senate bill passed into law on May 14, 1987. 1987 Mich.Pub.Acts No. 28. As a result of the 1987 statute, petitioners were ordered to refund nearly $25 million to disabled employees. **The Michigan Supreme Court found the statute invalid on the grounds that the retroactive provisions did not further a rational legislative purpose and that the employers had vested rights in coordination for Contract Clause purposes.** 436 Mich. 515, 462 N.W.2d 555 (1990).

Table 11: An example in domain of Legal Document

Model	d_1_vs_d_0	d_2_vs_d_1	d_3_vs_d_2	d_4_vs_d_3	d_5_vs_d_4	d_6_vs_d_5	d_7_vs_d_6	d_8_vs_d_7	d_9_vs_d_8	d_{10}_vs_d_9	Avg.
Sparse Retriever											
BM25	13.91	16.50	16.81	18.14	22.10	29.04	37.78	38.87	39.93	40.19	25.90
Dense Retriever											
jina-embeddings-v3	73.43	70.52	67.45	66.66	65.32	63.40	63.15	62.82	65.13	60.35	65.82
gte-large-en-v1.5	76.74	73.85	72.70	69.91	70.32	68.05	67.39	64.09	65.14	62.58	69.08
NV-Embed-v2	82.57	76.39	74.45	72.10	73.27	69.15	69.48	66.74	68.75	71.57	72.45
bge-en-icl	79.40	70.58	69.36	68.13	64.80	63.12	63.01	61.72	61.31	63.69	66.51
gte-Qwen2-7B-instruct	79.84	74.02	70.57	69.97	65.44	60.54	61.35	59.42	59.55	60.40	66.11
gte-Qwen2-1.5B-instruct	74.30	71.80	72.28	68.49	69.32	65.69	67.08	64.97	63.46	65.09	68.25
e5-mistral-7b-instruct	75.11	67.88	62.73	58.61	56.87	54.52	55.26	54.03	56.68	61.94	60.36
GritLM-7B	79.59	77.73	73.40	74.71	75.56	72.15	73.52	71.87	72.01	75.21	74.58
LLM2Vec	83.50	74.25	73.43	72.24	70.36	67.21	66.99	67.07	66.49	67.48	70.90
Fine-tuned Reranker											
bge-reranker-v2-m3	76.08	68.06	63.83	62.06	60.65	58.35	54.79	50.60	48.57	44.96	58.80
bge-reranker-v2-gemma	87.98	82.08	78.07	77.10	76.21	72.13	68.84	65.63	62.32	56.13	72.65
followIR	61.99	59.82	60.87	60.76	59.91	56.41	51.63	47.93	44.71	43.52	54.76
RankZephyr	90.20	86.04	83.96	83.14	82.23	79.41	76.07	72.43	70.26	66.84	79.06
Zero-shot LLM for Ranking											
GPT-4o	93.37	92.76	92.20	91.16	90.51	89.55	88.38	86.82	85.96	85.26	89.60

Table 12: Average Win Rate Comparison Between Documents in Task 2

Model	People	Books	Movies	Medical	Legal	Avg.
Sparse Retriever						
BM25	14.86	17.88	16.84	19.25	12.14	16.19
Dense Retriever						
jina-embeddings-v3	10.55	8.65	10.24	14.72	13.10	11.45
gte-large-en-v1.5	11.74	8.84	12.96	15.70	15.16	12.88
NV-Embed-v2	10.17	8.80	7.52	10.17	8.94	9.12
bge-en-icl	13.48	12.74	15.18	19.81	14.44	15.13
gte-Qwen2-7B-instruct	12.71	15.56	13.62	16.37	17.56	15.16
gte-Qwen2-1.5B-instruct	12.38	13.26	10.48	16.81	12.51	13.09
e5-mistral-7b-instruct	9.17	9.92	8.20	10.75	12.25	10.06
GritLM-7B	8.52	5.35	8.32	8.98	9.86	<u>8.21</u>
LLM2Vec	12.81	7.93	9.56	8.12	10.49	9.78
Fine-tuned Reranker						
bge-reranker-v2-m3	42.40	32.22	34.82	28.35	31.24	33.81
bge-reranker-v2-gemma	27.50	18.94	16.52	13.42	24.41	20.16
followIR	35.81	31.60	23.70	25.07	28.43	28.92
RankZephyr	20.21	16.34	14.86	11.53	25.31	17.65
Zero-shot LLM for Ranking						
GPT-4o	7.21	4.22	6.78	6.32	10.35	**6.98**

Table 13: Flip Rate for query format shift (Task 3). The Flip Rate reflects the win rate reversal when switching the query format from instruction-style to descriptive-style.

Model	d_1_vs_d_0	d_2_vs_d_1	d_3_vs_d_2	d_4_vs_d_3	d_5_vs_d_4	d_6_vs_d_5	d_7_vs_d_6	d_8_vs_d_7	d_9_vs_d_8	d^+_vs_d_9	Avg.
Sparse Retriever											
BM25	11.91	14.34	14.75	14.45	15.99	24.75	36.58	37.68	38.02	39.34	24.78
Dense Retriever											
jina-embeddings-v3	64.8	60.11	58.61	58.31	57.83	56.36	56.42	57.20	58.56	58.59	58.68
gte-large-en-v1.5	66.63	61.15	57.66	59.44	56.26	55.10	53.64	53.14	51.34	54.51	56.89
NV-Embed-v2	68.83	62.87	60.97	62.16	61.78	61.80	62.04	61.10	62.75	64.78	62.91
bge-en-icl	70.55	62.18	59.35	60.33	59.70	60.19	59.28	58.95	60.16	60.23	61.09
gte-Qwen2-7B-instruct	68.63	64.87	61.91	60.28	61.98	58.70	59.28	59.06	60.28	59.69	61.47
gte-Qwen2-1.5B-instruct	69.51	66.38	63.47	61.89	60.13	58.76	58.57	57.36	58.62	62.11	61.68
e5-mistral-7b-instruct	69.98	63.50	60.30	59.3	56.77	54.52	53.05	53.14	51.47	53.99	57.60
GritLM-7B	73.46	70.37	69.19	70.36	68.36	67.05	68.34	61.55	58.38	59.57	66.66
LLM2Vec	70.57	68.37	67.10	67.10	66.35	58.73	36.04	36.81	35.37	37.03	54.35
Fine-tuned Reranker											
bge-reranker-v2-m3	73.65	66.55	63.28	61.65	54.60	38.54	28.42	27.92	28.14	28.02	47.08
bge-reranker-v2-gemma	82.39	77.63	71.80	70.00	65.10	56.91	41.75	32.01	28.84	29.96	55.64
followIR	50.48	51.78	49.64	46.27	37.60	25.22	24.41	24.51	25.67	24.71	36.03
RankZephyr	85.41	79.91	72.91	70.41	64.41	52.91	47.91	45.41	42.30	42.56	60.41
Zero-shot LLM for Ranking											
GPT-4o	89.41	88.15	87.50	87.80	86.30	84.90	83.20	82.95	81.10	80.35	85.17

Table 14: Effect of document length on retrieval performance (padded to 512 words).

Model	d_1_vs_d_0	d_2_vs_d_1	d_3_vs_d_2	d_4_vs_d_3	d_5_vs_d_4	d_6_vs_d_5	d_7_vs_d_6	d_8_vs_d_7	d_9_vs_d_8	d^+_vs_d_9	Avg.
Sparse Retriever											
BM25	12.25	14.32	14.86	14.81	15.97	24.48	36.00	37.53	38.85	39.53	24.86
Dense Retriever											
jina-embeddings-v3	64.56	59.74	58.13	57.20	55.47	55.90	55.33	53.70	54.75	54.81	56.96
gte-large-en-v1.5	68.62	61.61	58.47	54.77	54.97	54.52	54.44	49.88	39.09	38.34	53.47
NV-Embed-v2	59.23	61.26	62.58	62.81	64.57	63.95	62.98	63.49	65.65	67.55	63.41
bge-en-icl	66.08	61.93	60.83	59.63	59.34	61.04	61.08	51.67	36.00	35.95	55.36
gte-Qwen2-7B-instruct	66.06	63.23	63.36	61.35	59.63	58.56	56.62	57.97	36.61	35.96	55.94
gte-Qwen2-1.5B-instruct	68.46	63.66	62.02	62.21	59.47	60.78	59.38	58.71	35.01	34.91	56.46
e5-mistral-7b-instruct	66.97	61.11	59.05	54.53	54.40	54.47	53.02	54.55	53.88	53.57	56.56
GritLM-7B	71.47	67.97	69.41	56.02	54.49	52.58	54.21	56.35	54.87	57.26	59.46
LLM2Vec	74.81	72.05	71.55	26.85	26.68	26.35	25.24	26.16	26.04	27.27	40.30
Fine-tuned Reranker											
bge-reranker-v2-m3	76.56	70.75	56.83	18.33	19.86	19.09	18.55	20.75	19.88	21.78	34.24
bge-reranker-v2-gemma	83.48	77.02	66.34	19.76	20.54	19.81	20.64	19.34	20.65	21.46	36.90
followIR	52.36	51.43	19.70	18.35	17.15	17.11	17.78	16.94	17.77	18.75	24.73
RankZephyr	84.34	76.84	66.64	35.14	33.34	34.04	31.54	32.64	30.74	31.24	45.65
Zero-shot LLM for Ranking											
GPT-4o	88.80	86.50	85.80	86.10	83.50	82.00	80.20	79.50	78.30	77.63	82.83

Table 15: Effect of document length on retrieval performance (padded to 1024 words).