SSRB: Direct Natural Language Querying to Massive Heterogeneous Semi-Structured Data

Xin Zhang[1,2], **Mingxin Li, Yanzhao Zhang, Dingkun Long, Yongqi Li**[2,*], **Yinghui Li**[3*]
Pengjun Xie, Meishan Zhang[1,*], **Wenjie Li**[2], **Min Zhang**[1], **Philip S. Yu**[4]
[1]Harbin Institute of Technology, Shenzhen [2]The Hong Kong Polytechnic University
[3]Tsinghua University [4]University of Illinois Chicago
Dataset: `https://hf.co/datasets/vec-ai/struct-ir`
Code: `https://github.com/vec-ai/struct-ir`

Abstract

Searching over semi-structured data with natural language (NL) queries has attracted sustained attention, enabling broader audiences to access information easily. As more applications, such as LLM agents and RAG systems, emerge to search and interact with semi-structured data, two major challenges have become evident: (1) the increasing diversity of domains and schema variations, making domain-customized solutions prohibitively costly; (2) the growing complexity of NL queries, which combine both exact field matching conditions and fuzzy semantic requirements, often involving multiple fields and implicit reasoning. These challenges make formal language querying or keyword-based search insufficient. In this work, we explore neural retrievers as a unified non-formal querying solution by directly index semi-structured collections and understand NL queries. We employ LLM-based automatic evaluation and build a large-scale semi-structured retrieval benchmark (SSRB) using LLM generation and filtering, containing 14M semi-structured objects from 99 different schemas across 6 domains, along with 8,485 test queries that combine both exact and fuzzy matching conditions. Our systematic evaluation of popular retrievers shows that current state-of-the-art models could achieve acceptable performance, yet they still lack precise understanding of matching constraints. While by in-domain training of dense retrievers, the performance can be significantly improved. We believe that our SSRB could serve as a valuable resource for future research in this area, and we hope to inspire further exploration of semi-structured retrieval with complex queries.

1 Introduction

Information access through natural language (NL) queries has been a fundamental need for a long time [1]. The ability to freely retrieve relevant information using language has profound implications for user experience and system accessibility. This is particularly crucial in the context of semi-structured data [2], which combines structured fields with unstructured content, offering a flexible representation for diverse information types [3, 4]. With the recent surge in LLM agents and RAG systems, semi-structured data has become increasingly prevalent and diverse [5], while accurate interaction with such data plays a crucial role in system performance. Traditional approaches to querying/searching semi-structured data typically fall into two categories: database-oriented formal methods (e.g., NL2SQL followed by database queries) and keyword-based full-text search [6, 4].

Driven by LLMs and AI advances, NL queries are becoming increasingly sophisticated and unconstrained, incorporating more diverse and challenging retrieval requirements that demand higher

*Corresponding Authors

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Track on Datasets and Benchmarks.

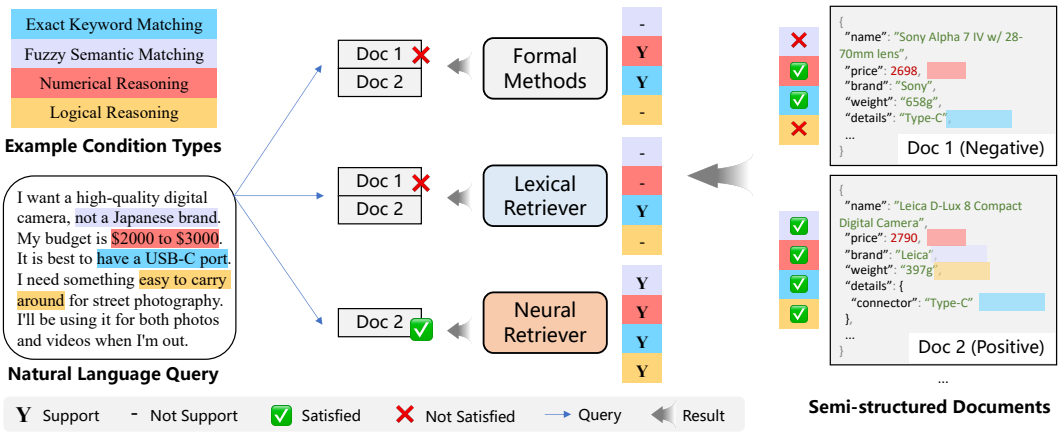

Figure 1: Task illustration. We evaluate the capabilities of current neural retrievers in understanding complex NL queries and semi-structured data. The queries involve diverse types of filtering conditions for structured objects, including exact and semantic matching, numerical and logical reasoning, or comprehensive understanding of multiple fields. The document structure can be dynamic, with potential missing fields and flexible structures (nested lists or dictionaries), making it challenging to query using fixed-schema database indexing. Current powerful LLM-based neural retrievers show promise in providing a unified solution to address the challenges present in this scenario.

levels of system intelligence [7, 8]. They may involve fuzzy semantic understanding, numerical reasoning, logical and contextual inference, or complex multi-field search conditions (Figure 1). These challenges require systems to possess advanced NL understanding and reasoning abilities for accurate data comprehension and constraint filtering. Additionally, the growing diversity of data structures, which may be incomplete or follow varying schema definitions, could be difficult to build conventional database indices and execute formal queries [2]. Hence, traditional formal methods and keyword-based approaches become inadequate, necessitating more promising solutions.

Given above challenges, we suppose that neural retrievers could offer a more promising solution, which, particularly when enhanced by LLMs, have demonstrated exceptional prowess in understanding and reasoning tasks [9, 10]. To harness the potential, we take the problem of processing complex NL queries over large-scale semi-structured data as a text retrieval task (§2), and develop a comprehensive benchmark to systematically evaluate the capabilities of neural dense retrievers.

We present the Semi-Structured Retrieval Benchmark (SSRB), encompassing 6 domains with 99 different data schemas, totaling 14M data objects, along with 8,485 NL queries of varying difficulty levels (3). Given the scarcity of public data, we build SSRB by LLMs in a three-stage data synthesis workflow (Figure 2): (1) schema generation, creating multiple schema definitions for six manually defined domains; (2) data triples generation, synthesizing $< query, positive, negative >$ triples for each schema using different query characteristic configurations to ensure diversity and quality; and (3) testset annotation, employing powerful LLMs to judge the relevance recalled candidates to queries as test labels. To validate the reliability of LLM annotations, we conduct human evaluation, demonstrating that LLM labels achieve high agreement with human labels (reaching 90% of human performance) while being significantly faster. And we train dense retrievers on the generated data, which shows improved performance, indicating the effectiveness as training data for future work.

Based on SSRB, we evaluate two main types of dense retrievers: 1) small-scale encoder-based models like InstructOR [11] and BGE [12], and 2) LLM-based ones such as E5-mistral [13]. We also include the BM25 lexical retriever for comparison. Our experiments reveal several key findings: 1) BM25 struggle with this task, 2) encoder-based models, benefiting from BERT-style backbones, provide better performance than BM25, and 3) LLM-based retrievers achieve notably better performance, highlighting the importance of LLM's powerful semantic understanding and reasoning capabilities in handling complex queries. However, their absolute performance remains relatively low, indicating the necessity for developing more task-specific retrievers. We conduct extensive analysis of results to understand model behaviors and checks to further verify the reliability of our benchmark. We believe

that our SSRB could serve as a valuable resource for future research, and we hope to inspire further exploration of semi-structured retrieval with NL queries.

2 Querying Task Formulation and Settings

We aim to evaluate the potential of neural information retrieval in addressing complex NL queries over semi-structured data. To facilitate a clear and systematic investigation, this section formulates the retrieval task and different querying settings.

2.1 Task Formulation

We start with the concept of *semi-structured data*, which has been discussed from different perspectives [2, 3, 4]. Synthesizing these viewpoints, we identify two key characteristics of semi-structured data: **(1) They follow defined schema** [2, 4], although the schema definition may be incomplete or contain fields that cannot be strictly typed, often incorporating natural language text; **(2) The structure is partial** [2, 3], meaning that while they follow schema definitions, specific data instances may have missing structural fields[2]. Data exhibiting these two characteristics constitute the semi-structured data investigated in this work.

Our another focus is on *complex natural language query*, which express data requirements through natural language descriptions containing **multiple filtering conditions**. A collection of semi-structured data forms the search space, from which users can retrieve desired data objects using natural language queries. Some methods convert these queries into formal languages (e.g., SQL) using database index to query the data, which is beyond the scope of this work. We explores neural IR to address such querying, leveraging pre-trained language models to construct *neural index* for semi-structured data and understand complex queries.

Specifically, we compute *semantic relevance* scores between a query q and different data objects d, rank all data in the collection D based on these scores, and return the top-k objects as query outputs. These outputs are identifed relevant documents that satisfy the conditions of the query q.

2.2 Querying Settings

In real-world applications, these semi-structured data may originate from either a single data source, where all objects follow the same schema definition, or multiple data sources, where the queried structured data encompasses different schemas. Similar to many existing benchmarks, we also consider two settings: **(1) In-schema retrieval**: the data objects in the collection share the same schema definition; **(2) Cross-schema retrieval**: the data objects in the collection are from different schema definitions, and the task is to retrieve relevant documents in the correct schema.

3 Semi-Structured Retrieval Benchmark

Guided by the task formulation in §2, we construct the benchmark, named SSRB, by a three-stage workflow, as shown in Figure 2. We describe the workflow in detail and provide data analysis in following subsections. Table 1 summarizes the statistics of our benchmark.

3.1 Data Generation

Given the scarcity and high cost of real-world public datasets, we leverage powerful LLMs to construct a dataset spanning diverse domains and schema structures. The LLM-based approach for generating retrieval data has been extensively studied and proven effective in previous research [13, 14, 15, 16]. Our data generation first generates diverse data schemas across different domains, and then produce numerous $< query, positive, negative >$ triplets for each schema definition, which follows previous works [13]. This enables us to generate large-scale heterogeneous semi-structured data with guaranteed query relevance, facilitating subsequent testset annotation.

[2]Such as Figure 1, one of the documents might be missing the "weight" field (*i.e.,* 'weight' is null), which would affect how relevant the document is to the example query.

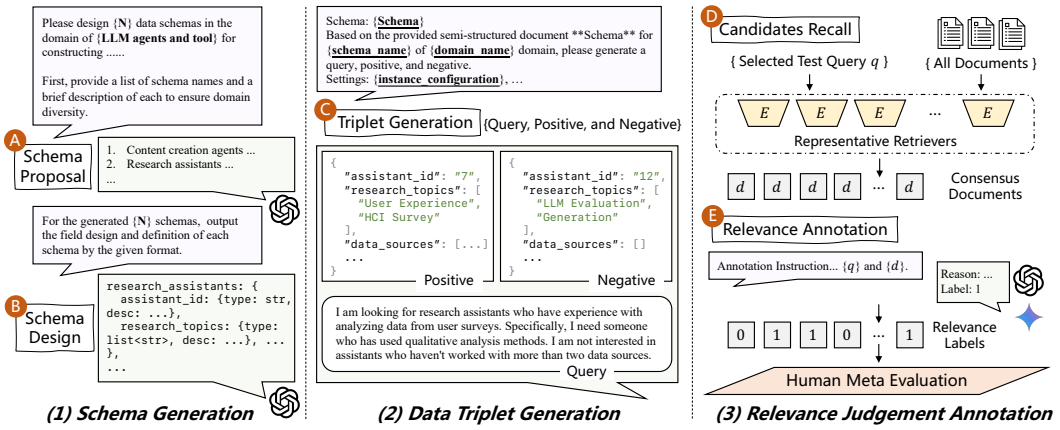

Figure 2: Benchmark construction workflow, which comprises three stages: **(1)** Schema generation: we prompt LLM with detailed instruction to generate 99 schemas across 6 domains. **(2)** Data triplet generation: for each schema, we setup multiple data configurations (*e.g.,* different number of exact or semantic matching conditions), and reptively prompt LLM to generate thousands <query, positive, negative> triplets. **(3)** Relevance judgment annotation: we select 8,000+ queries, employ several representative retrievers to recall possible positive documents, and then use LLMs to judge the relevance, where a human meta evaluation is conducted to measure the labeling accuracy.

Schema Generation We manually establish six domains (see Table 1). For each one, as shown in Figure 2 stage (1), we generate schemas by two steps: Ⓐ **Schema proposal**: Using a powerful LLM[3] to brainstorm N schema names and descriptions within each domain, ensuring schema diversity in the given domain. Ⓑ **Schema design**: Leveraging the context from step Ⓐ, we instruct the LLM to generate detailed schema definitions, specifying field names, data types, and descriptions for each schema. The prompt is in App. Figure 5. Additionally, for domains with potentially higher schema similarity[4] (*e.g.,* product or résumé search), during the schema proposal step Ⓐ, we let the LLM also design shared field definitions (prompt in App. Figure 6). This stage resulted in 99 distinct schemas across the six domains, with manual verification ensuring definition quality and coherence.

Triplet Generation Given a schema definition, as in Figure 2 stage (2), we instruct a strong but cheaper LLM[5] to generate one data triplet $< query, positive, negative >$ at a time [13], and obtain a large number of triplets by varying the generation configurations. This ensures natural semantic relationships and distinctiveness among the structured objects, enhances consistency, and facilitates direct selection of high-quality query-positive pairs for testing. To ensure query diversity, we designed five **configuration slots**: number of exact matching conditions, number of fuzzy semantic matching conditions, inclusion of scenario descriptions, use of common word abbreviations, and alternative representations for numbers and dates. The prompt is in App. Figure 7. By combining these slots with schema definitions, we create multiple prompt instances. For each one, we execute multiple LLM runs with high generation temperature to produce hundreds of different triplets.

Ultimately, we generate approximately 70k triplets per schema, resulting in a total collection of 14M documents across all schemas, as shown in Table 1.

3.2 Testset Annotation

After data generation, the next step is to establish a reliable testset for evaluation, which comprises two parts: *test queries* and *relevance labels*, *i.e.,* the identifiers of all positive documents for each query. We first randomly sample[6] around 86 queries for each schema as the testset, obtaining a total

[3]Here we use GPT-4o.

[4]For instance, product search schemas often show similarities as they share common fields (e.g., ID, name, price, brand) for better data management in practice.

[5]`https://huggingface.co/Qwen/Qwen2.5-32B-Instruct`

[6]Although one may consider that using some filters to select queries is better. As we want to keep the diversity of queries and not disgard hard ones, we do not apply any filters.

Table 1: Statistics of our dataset. `Pos.` denotes the positive document that meets the query.

Domain	#Schema	#Document	#Query	Avg. #Pos. per Query
Academic	19	2,734,920	1,896	20.12
Finance	10	1,438,706	993	12.76
Human Resources	10	1,440,000	802	15.13
Product	30	4,338,366	2,401	13.97
Résumé	10	1,439,978	798	11.81
LLM Tool & Agent	20	2,878,180	1,595	11.04
Total	99	14,270,150	8,485	14.33

of 8,485 queries. Then we identify the relevant documents from the retreival collection for these queries by two steps, as shown in Figure 2 stage (3).

Candidates Recall In our data, each triplet is generated independently, so documents from one triplet may also satisfy the query conditions of another triplet. These cross-triplet relevance could be false negatives if they are not selected, this is a common issue in retrieval benchmarks. To mitigate this and improve evaluation reliability, we perform an additional round of positive selection from the full corpus and annotate their relevance. For each query q, judging the relevance of all documents is impractical due to the large size of the collection (140k objects for each schema). We follow the common practice of using retrievers to recall candidate documents for labeling, which could significantly reduce the workload. We select several representative models from two major types of dense retrievers, *i.e.,* Encoder-based text embedding models and LLM-based ones, as the retriever committee. We retrieve top-100 documents from the in-schema collection for each q, and consider a document as a candidate only if it appears in the top-100 results of more than C retrievers. For each q we finally obtain about 50 candidates. The technical details is presented in Supplementary material.

LLM Relevance Judgment We utilize LLMs to judge the relevance label for each query-document pair, as recent studies have demonstrated that LLM-based annotation can achieve comparable performance to human annotation [17] while being more cost-effective and time-efficient. Specifically, we prompt LLMs to analyze the conditions in query q, compare them with d. The LLM need to provide reasoning content about their matching degree before generating a final relevance label (0 for irrelevant, 1 for relevant). The prompt is in App. Figure 8. In practice, we employ two strong LLMs[7] to annotate each query-document pair in parallel, and adopt an OR-gate strategy where the final label is 1 if either LLM assigns a positive label. This strategy leads to improved annotation accuracy, as demonstrated in our analysis in the following §3.3 and Table 2.

Human Annotation To validate and control the quality of LLM labels, we conducted additional human annotation. We sample over 5,500 query-candidate pairs and employed annotators following similar annotation principles and guidelines (the guideline and annotation interface are in Appendix §B.3). Specifically, we establish a two-tier annotation process: the first tier consists of primary annotators responsible for initial labeling, while the second tier comprises reviewers who examine and validate the assigned labels. The annotation is conducted in batches, with reviewers performing random sampling checks on each batch. If the inconsistency rate in the sampled annotations of a batch exceeded a threshold (95%), the entire batch would be returned to the annotators for revision. After the human annotation, we compare the labels from LLMs

Table 2: Annotation agreement scores (on 5,571 sampled query-doc pairs) of relevance judgments from LLMs and human annotators. *Denotes the inter-annotator agreement without label refinement, which is a fair comparison with LLMs.

Pred.	Ref.	Agreement	Ref. Recall		Pred. Precision	
			Pos.	Neg.	Pos.	Neg.
Gemini 2.0 Flash	Human	70.97	59.05	89.13	89.22	58.83
GPT-4o-mini	Human	67.10	50.49	92.39	91.00	55.06
Merged	Human	75.98	69.40	86.01	88.31	64.86
Human*	Human	86.25	-	-	-	-

[7]We use two powerful but affordable LLM services, *i.e.,* Gemini-2.0-Flash and GPT-4o-mini.

and human annotators. As shown in Table 2, the agreement rate between LLM labels and human annotations on these pairs reached 75.98%, while the inter-annotator agreement among humans was 86.25%. The acceptable gap between LLM and human performance indicates the reliability of our LLM-based annotation approach to a considerable extent.

It worth to note that, in Table 2, the Ref. Recall of Pos. and Pred. Precision of Neg. from Gemini and GPT are close to 50%, which is relatively low. The values reflect the fact that LLMs tend to be very conservative when predicting positive cases. They only label an instance as positive when they are highly confident, which results in very high positive precision (Pred. Precision Pos. nearly 90%). However, this cautious behavior also leads to many missed positives, resulting in relatively low positive recall (Ref. Recall Pos. around 50-60%). Overall, from the task performance view, the gap between LLM and human performance is substantial. We consider the much lower time and financial cost by LLMs and suppose it is acceptable.

Finally, to expand the testset coverage, we take the LLM relevance labels as the final relevance judgment, which enables us to have relevance labels for all 8,485 queries. Table 1 shows the statistics.

3.3 Dataset Summary and Analysis

Through the data construction workflow described in §3.1 and §3.2, we generate 99 schema definitions across 6 different domains, comprising a retrieval collection of 14M semi-structured objects. We annotate 8,485 test queries, with each query having approximately 14.33 relevant documents on average. Detailed domain-level statistics are presented in Table 1. We provide comprehensive schema-specific information, generation costs and annotation details in Supplementary material.

Training Data The rest part of generated $< query, positive, negative >$ triplets (that not selected as testset) could be used as training data for the development of better neural retrievers of this task, where we show it could achieve significant improvements in the experiment §4.2.

Query Distribution In data generation, we vary the number of exact and semantic filtering conditions to create diverse queries. Intuitively, queries with more filtering conditions would specify more precise search targets, thus corresponding to fewer relevant documents. We analyze the distribution of different query types and their relevance labels, as illustrated in Figure 3. The distribution aligns with our expectations: simpler queries with fewer filtering conditions yield more positive matches, while more complex queries with multiple filtering conditions result in fewer positives.

This diverse query distribution enables a comprehensive evaluation of neural retrieval performance across different difficulty levels, demonstrating the comprehensiveness of our benchmark.

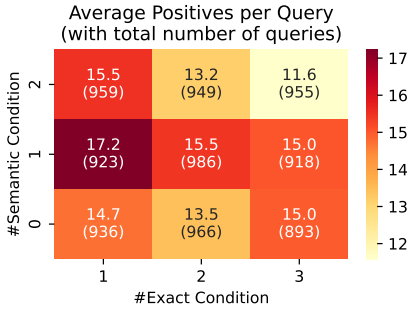

Figure 3: Query and labels distribution by condition numbers. Generally, queries with more filtering conditions yield fewer relevant documents.

4 Evaluation

In this section, we conduct the evaluation of the current popular retrievers (primarily dense retrievers) on SSRB, and analyze their behavior in complex NL queries and semi-structured data. Before the evaluation, we first introduce the research questions driving the experiment design and guide the smooth understanding, which could help us to logically solidify the findings and conclusions.

- **RQ1:** How do current retrievers perform on this querying task? To what extent can they accomplish this task effectively?
- **RQ2:** Does the data mixture from multiple schemas increase the difficulty of querying?
- **RQ3:** How well can models comprehend and process multiple, complex filtering conditions? How does their performance vary across different query distributions?
- **RQ4:** How reliable is our benchmark in evaluating model performance, and does it provide meaningful assessments?

Table 3: In-schema retreival performance of selected methods. We report Recall@20 (R@20) and nDCG@10 (N@10). Marco Avg. means the average across the scores of six domains.

	Marco Avg.		Academic		Finance		Résumé		HR		LLM		Product	
	R@20	N@10	R@20	N@10	R@20	N@10	R@20	N@10	R@20	N@10	R@20	N@10	R@20	N@10
BM25	2.01	2.07	1.68	2.63	2.35	1.90	0.75	0.72	3.05	2.86	2.88	2.40	1.38	1.93
InstructOR	4.06	4.95	4.37	6.29	4.22	4.36	2.34	1.94	5.68	8.73	4.72	4.53	3.03	3.83
$BGE_{large\text{-}v1.5}$	3.99	4.62	3.41	4.74	3.10	3.15	2.36	3.00	5.69	7.97	6.68	5.64	2.72	3.20
Jina-v3	5.85	7.00	6.66	10.34	5.16	5.56	3.53	3.98	8.05	9.68	7.33	6.84	4.33	5.57
Nomic-v2	6.06	8.10	7.86	12.63	4.50	5.24	3.48	4.13	7.31	11.33	6.76	6.62	6.46	8.68
$Drama_{1B}$	14.42	16.71	16.56	23.71	13.46	13.66	11.30	12.00	14.02	17.66	14.86	14.08	16.33	19.13
$GTE_{Qwen2\text{-}7B}$	13.66	16.35	17.05	24.31	12.17	12.21	11.53	12.15	12.67	19.26	14.84	13.78	13.68	16.39
$E5_{mistral\text{-}7b}$	14.66	17.94	18.34	26.96	12.48	13.77	10.83	11.82	15.05	21.93	14.40	14.20	16.87	18.97
$GritLM_{7B}$	13.65	16.66	16.03	24.51	11.10	12.34	12.92	14.35	11.37	15.33	13.18	13.09	17.31	20.36
NV-Embed-v2	14.52	17.26	16.49	24.66	12.92	13.22	12.54	14.30	13.68	18.39	13.68	13.52	17.77	19.50
Fine-Tuned														
Qwen2.5-0.5B	22.70	22.03	23.03	28.28	21.45	18.55	28.00	22.17	19.31	22.44	21.55	17.88	22.85	22.83
Qwen2.5-1.5B	24.17	22.09	23.79	28.77	22.70	20.47	28.92	22.68	23.04	21.18	22.96	18.76	23.60	20.66

4.1 Settings

Metrics As we formulate the task as a ranking problem, we adopt standard retrieval metrics to evaluate the performance, *i.e.,* Recall@K and nDCG@K. Generally, Recall@K measures the proportion of positives in the top-K results to the number of all labeled positives, while nDCG@K evaluates the ranking quality by considering the ranks of positives in the results. We set K to 20 for Recall (*i.e.,* R@20), and K to 10 (*i.e.,* N@10), which arer common choices in retrieval tasks.

Models We select several representative models for evaluation. The BERT-like pre-trained transformer **Encoder-based** dense retrievers are widely used in retrieval tasks due to their strong performance and efficiency. We select the following models: InstructOR [11], BGE [12], Jina-v3 [18], Nomic-v2 [19]. **LLM-based** retrievers have shown great potential in both common [20] and advcanced retrieval tasks, like the reasoning-intensive retrieval [9], by their powerful understanding of natural language and reasoning capabilities. We select the recent representative models: E5-mistral-7b [13], GritLM-7B [21], GTE-Qwen2-7B [22], NV-Embed-v2 [23], and Drama-1B [15].

Although we illustrated (Figure 1) that the lexical based retrievers might not be capable of handling complex semantic-based NL queries in this task, we still include them for comparison. We utilize the classic BM25 [24], by the implementation of `BM25s`[8] [25], as the lexical-based retriever.

We also fine-tune the smaller LLMs [26] on sampled 1M triplets from our dataset to demonstrate the effectiveness of our generated data for training better retrievers in this task, as well as the gap between the current generalist retrievers and task-specific ones. Appendix §C.2 provides the technical details.

4.2 Results

To RQ1: In-schema retreival. To fundamentally evaluate retrievers on SSRB, we first conduct in-schema retrieval experiments to isolate other influencing factors. Specifically, for each query we only search on the collection from the corresponding schema. The results are presented in Table 3. As expected, BM25 shows relatively poor performance due to its lack of semantic understanding capabilities. Neural models generally achieve higher scores, with LLM-based models outperforming encoder-based models thanks to their superior language understanding capabilities. Overall, the best recall scores among these models remain under 40%, which, while acceptable, indicates substantial room for improvement. Notably, models fine-tuned on our generated data achieve comparable or even superior performance to extensively pre-trained 7B models. This demonstrates the effectiveness of our training data in developing more powerful task-specific models.

To RQ2: In-domain cross-schema retreival. Next, we investigate the impact of schema mixing on querying performance. We combine data from different schemas within the same domain into a single collection for retrieval. The results are shown in Table 4, where a consistent but small

[8] `https://github.com/xhluca/bm25s`

Table 4: In-domain cross-schema retrieval performance. We report Recall@20 (R@20) and nDCG@10 (N@10). Marco Avg. means the average across the scores of six domains.

	Marco Avg.		Academic		Finance		Résumé		HR		LLM		Product	
	R@20	N@10	R@20	N@10	R@20	N@10	R@20	N@10	R@20	N@10	R@20	N@10	R@20	N@10
InstructOR	3.66	3.99	3.52	4.93	4.07	4.23	2.32	1.93	4.29	4.89	4.57	4.22	3.18	3.76
$BGE_{large\text{-}v1.5}$	3.76	4.33	3.04	4.28	2.91	3.00	2.30	2.92	5.47	7.57	6.34	5.26	2.48	2.96
Jina-v3	5.40	7.69	5.37	8.30	4.96	5.42	3.53	3.98	7.11	15.32	6.96	7.70	4.45	5.41
Nomic-v2	5.56	7.67	7.00	11.50	4.18	5.04	3.29	3.90	5.74	9.56	6.70	6.76	6.47	9.27
$Drama_{1B}$	13.98	17.18	15.73	22.61	13.34	13.52	10.98	11.69	12.97	20.80	14.71	13.81	16.17	20.68
$GTE_{Qwen2\text{-}7B}$	13.48	16.15	16.59	23.76	12.21	12.20	11.44	12.18	12.18	18.70	14.79	13.73	13.65	16.31
$E5_{mistral\text{-}7b}$	14.48	17.61	17.83	26.40	12.68	13.78	10.66	11.57	14.51	21.27	14.37	13.98	16.85	18.66
$GritLM_{7B}$	14.09	17.28	15.61	23.77	11.06	12.30	12.89	14.35	14.36	15.83	13.76	17.83	16.86	19.58
NV-Embed-v2	14.71	18.21	15.81	23.63	12.78	13.14	12.52	14.29	15.97	20.98	13.81	16.99	17.40	20.25
Fine-Tuned														
Qwen2.5-0.5B	22.46	20.19	23.94	26.72	21.34	18.55	27.95	22.17	18.20	16.92	21.73	17.99	21.58	18.78
Qwen2.5-1.5B	23.50	21.53	23.02	27.99	22.54	20.45	28.87	22.64	21.20	19.19	23.22	18.98	22.17	19.92

performance degradation is observed across all models. This indicates that despite the increased noise in schema-mixed collections, models can still generate distinctively different representations for data from various schemas, effectively distinguishing their semantic characteristics. The finding suggests the promising potential of using dense retrievers as a unified neural index for simultaneously handling querying tasks across multiple schemas.

4.3 Analysis

To RQ3: Query complexity and performance. We analyze the performance of different models on queries with varying numbers of filtering conditions. The results are shown in Figure 4. We observe that, generally, the performance increases with the number of filtering conditions, indicating that models can better leverage the information provided by more conditions. For open-source models, the performance strongly correlates with the number of semantic conditions (the x axis). While our fine-tuned (Qwen-1.5B) model improved overall performance, it exhibits unstable behavior across various condition types, suggesting that additional research is necessary to develop better retrievers. Overall, the results shows that, with the robust understanding ability of LLMs, we could process complex queries and provide accurate results.

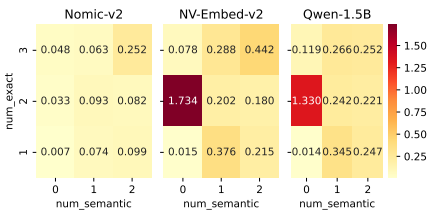

Figure 4: Performance on different query condition groups.

To RQ4: Evaluation reliability. As the testset relevance labels are from LLMs, one may question the reliability of our SSRB in evaluating model performance. To address this, we perform three fold checks: **(1) Label consistency:** as stated in §3.2, we conduct a meta-evaluation of the LLM labels by comparing them with human annotations on a sampled set of query-document pairs. The results are shown in Table 2, where we observe a high agreement rate of 75.98% between LLM and human labels, while human annotators show an agreement of 86.25% if we do not apply the refinement. We suppose the acceptable gap could demonstrates the reliability of LLM labels to a considerable extent. **(2) Performance consistency:** we compare the performance of several encoder and LLM models on SSRB with that on general text retrieval (BEIR [20]), as shown in Table 5. We observe that models within each category (encoder-based or LLM-based) demonstrate similar general text retrieval capabilities (comparable BEIR scores) due to their architectural limitations. This performance pattern is consistently reflected in our SSRB. Given that these models are evaluated in a zero-shot setting, this consistent trend is reasonable and indirectly validates the reliability of SSRB. **(3) LLM verification:** finally, we provide a direct verification of model performance by using our LLM judgment to label the retrieval results of these models. This is a more reliable way to evaluation, as in Table 2 the LLM judgment on positives are shown very high precision (88.31%). Table 5 shows the LLM positive rates, where we observe that they are consistent with the model performance on BEIR and SSRB. Combining these three checks, we believe that our benchmark could provide reliable evaluation.

Table 5: Comparison of model performance on text retrieval (BEIR) and our SSRB (in-schema), as well as LLM positive labeling rates on retrieval results.

Model	BEIR	SSRB	LLM Pos.
Jina-v3	53.17	7.00	8.99
Nomic-v2	52.86	8.10	11.05
$E5_{\text{mistral-7b}}$	57.07	17.94	19.99
$GTE_{\text{Qwen2-7B}}$	58.86	16.35	16.83
$GritLM_{7B}$	57.4	16.66	20.07
NV-Embed-v2	63.21	17.26	21.12

Table 6: Agreement scores of LLMs judgments to human labels. As a extension of Table 2, we tune the prompt and compare different LLMs.

Pred.	Agreement	Ref. Recall		Pred. Precision	
		Pos.	Neg.	Pos.	Neg.
Gemini 2.0 Flash	70.97	59.05	89.13	89.22	58.83
+ More strict to Neg.	60.26	37.64	94.70	91.54	49.93
GPT-4o-mini	67.10	50.49	92.39	91.00	55.06
GPT-4o	68.66	53.52	91.71	90.77	56.44
GPT-4.1	63.00	43.00	93.48	90.94	51.85
Claude-3.7-Sonnet	63.22	44.51	91.71	89.11	52.04
DeepSeek-R1	55.17	26.77	98.63	96.76	46.81

To RQ4: Tuning LLM Judgment Strategies Extending the analysis from Table 2, we explore optimization attempts for LLM judgment. First, we attempt to improve negative precision by making the LLM more strict in its negative judgment. As shown in the second row of Table 6, this approach proved unsuccessful, suggesting the need for more balanced instructions. Furthermore, we try different LLMs. Interestingly, as demonstrated in the second group of Table 6, more expensive models like DeepSeek-R1 do not necessarily yield better results. Thus, our current LLM judgment strategy is a reasonable choice, as it achieves a good balance between cost and performance.

Structure Representation: Json *v.s.* YAML In our main experiments, we encode structured objects as JSON-formatted strings as text input for the retriever models. Specifically, we use multi-line JSON strings with an indent of 4, which aligns better with human reading preferences. There are, of course, other approaches to converting structured documents into text, such as using non-indented single-line JSON strings or YAML format. We conducted a comparative study using the DRAMA model on a sampled small-scale dataset, as shown in Table 7. The results demonstrate that multi-line JSON strings, which are most aligned with human reading habits, yield the best performance.

Table 7: Comparison of different document string converting types on a subset.

R@20	Json	Json Compact	YAML
Drama-1B	19.50	16.58	15.39

5 Related Work

Benchmarking Text Retrieval. Initially, information retrieval (IR) datasets primarily focused on two typical application domains: Web retrieval and open-domain question answering. Web retrieval datasets included MS MARCO and the TREC series [27, 28], as well as question answering datasets [29, 30]. These datasets have provided comprehensive evaluations for both traditional and dense retrieval methods. As retrieval techniques evolved with the development of large language models, new evaluation capabilities have emerged. For example, testing a retrieval model's ability to follow instructions [31, 32] and reasoning [33, 9]. In this work, we introduce a new benchmark that focuses on the challenges of querying semi-structured data with natural language, which requires not only the understanding of structured data and following to query requirements but also involves deeper reasoning about numerical values and logical semantics.

Dense Retrieval. With the advent of language models like BERT [34], text retrieval shifted from traditional BM25-based inverted index retrieval to neural dense retrieval [35, 36, 37]. Dense retrieval was further advanced through techniques such as hard negative sample mining [38, 39] and improved pre-training designs [40, 41] in the following years. Recently, there has been a surge in open-source general-purpose dense retrieval models (or embedding models) [42]. Through training on large-scale datasets, both encoder-based [11, 18, 43, 19] and LLM-based retrieval models [44, 45, 13, 21, 16] have achieved significant performance improvements. We evaluate representative open-source dense retrieval models, such as E5 [13], GTE [22] and BGE [12], on our structured retrieval benchmark.

LLM-based Retrieval Data Synthesis. The acquisition of human-labeled data is exceedingly expensive and time-consuming. With the evolution of LLMs, there has been a growing interest in for data synthesis to improve the performance of retrieval models for specific tasks [13, 15, 10, 16]. For instance, Promptagator [46] introduced a data generation methodology based on few-shot learning,

[13] demonstrated the effectiveness of large-scale LLM-generated data for training retrievers. In addition to training data synthesis, there has been work on LLM-generated relevance judgments [17] and queries [47, 14] that showing the potential of LLMs in producing high-quality testsets for retrieval tasks. In this work, we further explore LLM-based data synthesis and automatic evaluation, demonstrating its effectiveness in generating high-quality data for semi-structured data retrieval tasks.

6 Discussion

For the differences between our task and from other retrieval tasks (e.g., passage retrieval, table QA, knowledge base QA), we provide the following discussion. Briefly, our task differs from traditional passage retrieval in two main ways: (1) our corpus consists of semi-structured data, whereas passage retrieval typically operates over unstructured text; and (2) our queries often contain multiple conditions that may require semantic reasoning, which is less common in standard passage retrieval settings. In contrast to QA tasks, our retrieval task focuses solely on identifying documents that are relevant to a given query, without generating or extracting an explicit answer from the documents— *i.e.,* there is no answer component involved.

For case studies, we provide several examples in Appendix §C.4. We observe that current retrievers may struggle with queries that require precise numerical comparisons (*e.g.,* "a salary between 80000 and 120000"), implicit conditions requiring semantic reasoning (*e.g.,* "strong project experience"), or complex logical conditions, leading to incorrect retrieval results.

7 Limitations

Despite the contributions of this study, we acknowledge its limitations: (1) A primary limitation is the reliance on LLMs for data generation and filtering. This dependency could introduce inherent biases found in LLMs, affecting both the benchmark dataset and the evaluations conducted. For example, LLMs might produce data and queries that reflect stereotypical viewpoints or overlook nuanced cultural contexts, thereby impacting the fairness and inclusivity of the retrieval process. (2) We do not conduct in-depth analyses of dense retrieval models or comparisons with approaches utilizing pure database queries similar to NL2SQL (which might be out of the scope of this paper, as we focus on evaluating retrieval models). These topics will be explored in detail in our future work. (3) Another limitation is the recall performance of the current state-of-the-art neural dense retrievers. While acceptable, it is not optimal for precise matching constraints, highlighting the need for further research to develop retrievers that better understand and manage the complexities of semi-structured data. With adequate training resources, larger dense retrieval models could be trained to investigate how model size scaling impacts semi-structured data retrieval using natural language queries.

8 Conclusion

In this work, we introduce SSRB, a novel benchmark addressing critical challenges in retrieving and interacting with vast heterogeneous semi-structured data through natural language queries. Our results highlights the limitations of current neural dense retrievers and underscores the need for models tailored to the specific demands of semi-structured data retrieval. By utilizing powerful LLMs, we crafted a substantial and diverse dataset that serves as a valuable resource for this area. Through systematic evaluation and insightful analyses, we establish a strong foundation for comprehending the complexities inherent in natural language-based querying of semi-structured data. Our findings emphasize the potential for innovative retrieval strategies that seamlessly combine exact field matching with fuzzy semantic requirements. We welcome researchers and practitioners to continue exploring new designs for dense retrievers in this domain.

Acknowledgments and Disclosure of Funding

We sincerely thank the anonymous reviewers and chairs for their valuable feedback and suggestions. This work receives partial support from the Natural Science Foundation of China (under Grants 624B2048 and 62336008), the Research Grant Council of Hong Kong (PolyU/15209724), and the Shenzhen Science and Technology Program (under Grant ZDSYS20230626091203008).

References

[1] David D Lewis and Karen Spärck Jones. Natural language processing for information retrieval. *Communications of the ACM*, 39(1):92–101, 1996.

[2] Serge Abiteboul. Querying semi-structured data. In *Proceedings of the 6th International Conference on Database Theory*, volume 1186 of *Lecture Notes in Computer Science*, pages 1–18, Delphi, Greece, January 1997. URL https://doi.org/10.1007/3-540-62222-5_33.

[3] Scott B. Huffman and Catherine Baudin. Toward structured retrieval in semi-structured information spaces. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, volume 2, pages 751–757, Nagoya, Japan, August 1997. URL https://aaai.org/papers/001-ws97-09-001.

[4] Stefanie Nadig, Martin Braschler, and Kurt Stockinger. Database search vs. information retrieval: A novel method for studying natural language querying of semi-structured data. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 1772–1779, Marseille, France, May 2020. URL https://aclanthology.org/2020.lrec-1.219.

[5] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=ehfRiF0R3a.

[6] Valentin Tablan, Danica Damljanovic, and Kalina Bontcheva. A natural language query interface to structured information. In *The Semantic Web: Research and Applications*, pages 361–375, Berlin, Heidelberg, 2008.

[7] Tianshu Wang, Xiaoyang Chen, Hongyu Lin, Xianpei Han, Le Sun, Hao Wang, and Zhenyu Zeng. Dbcopilot: Natural language querying over massive databases via schema routing. In *Proceedings 28th International Conference on Extending Database Technology, EDBT 2025, Barcelona, Spain, March 25-28, 2025*, pages 707–721, 2025. URL https://doi.org/10.48786/edbt.2025.57.

[8] Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. Text-to-sql empowered by large language models: A benchmark evaluation. *Proc. VLDB Endow.*, 17(5):1132–1145, January 2024. URL https://doi.org/10.14778/3641204.3641221.

[9] Hongjin SU, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han yu Wang, Liu Haisu, Quan Shi, Zachary S Siegel, Michael Tang, Ruoxi Sun, Jinsung Yoon, Sercan O Arik, Danqi Chen, and Tao Yu. BRIGHT: A realistic and challenging benchmark for reasoning-intensive retrieval. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=ykuc5q381b.

[10] Rulin Shao, Rui Qiao, Varsha Kishore, Niklas Muennighoff, Xi Victoria Lin, Daniela Rus, Bryan Kian Hsiang Low, Sewon Min, Wen-tau Yih, Pang Wei Koh, et al. Reasonir: Training retrievers for reasoning tasks. *arXiv preprint arXiv:2504.20595*, 2025. URL https://arxiv.org/abs/2504.20595.

[11] Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. One embedder, any task: Instruction-finetuned text embeddings. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1102–1121, Toronto, Canada, July 2023. URL https://aclanthology.org/2023.findings-acl.71/.

[12] Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 641–649, New York, NY, USA, 2024. URL https://doi.org/10.1145/3626772.3657878.

[13] Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 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, August 2024. URL https://aclanthology.org/2024.acl-long.642.

[14] Jianlyu Chen, Nan Wang, Chaofan Li, Bo Wang, Shitao Xiao, Han Xiao, Hao Liao, Defu Lian, and Zheng Liu. AIR-bench: Automated heterogeneous information retrieval benchmark. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 19991–20022, Vienna, Austria, July 2025. URL https://aclanthology.org/2025.acl-long.982/.

[15] Xueguang Ma, Xi Victoria Lin, Barlas Oguz, Jimmy Lin, Wen-tau Yih, and Xilun Chen. DRAMA: Diverse augmentation from large language models to smaller dense retrievers. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 30170–30186, Vienna, Austria, July 2025. URL https://aclanthology.org/2025.acl-long.1457/.

[16] Xin Zhang, Yanzhao Zhang, Wen Xie, Dingkun Long, Mingxin Li, Pengjun Xie, Meishan Zhang, Wenjie Li, and Min Zhang. Phased training for LLM-powered text retrieval models beyond data scaling. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=NC6G1KCxlt.

[17] Hossein A. Rahmani, Xi Wang, Emine Yilmaz, Nick Craswell, Bhaskar Mitra, and Paul Thomas. Syndl: A large-scale synthetic test collection for passage retrieval. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 781–784, Sydney NSW, Australia, 2025. URL https://doi.org/10.1145/3701716.3715311.

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

[19] Zach Nussbaum and Brandon Duderstadt. Training sparse mixture of experts text embedding models. *arXiv preprint arXiv:2502.07972*, 2025. URL https://arxiv.org/pdf/2502.07972.

[20] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems: Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=wCu6T5xFjeJ.

[21] Niklas Muennighoff, Hongjin SU, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. Generative representational instruction tuning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=BC4lIvfSzv.

[22] Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*, 2023. URL https://arxiv.org/pdf/2308.03281.

[23] Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. NV-embed: Improved techniques for training LLMs as generalist embedding models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=lgsyLSsDRe.

[24] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009. doi: 10.1561/1500000019. URL http://dx.doi.org/10.1561/1500000019.

[25] Xing Han Lù. Bm25s: Orders of magnitude faster lexical search via eager sparse scoring. *arXiv preprint arXiv:2407.03618*, 2024. URL https://arxiv.org/pdf/2407.03618.

[26] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/pdf/2412.15115.

[27] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*, 2016.

[28] Charles LA Clarke, Nick Craswell, and Ian Soboroff. Overview of the trec 2009 web track. In *Trec*, volume 9, pages 20–29, 2009.

[29] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. URL https://aclanthology.org/Q19-1026/.

[30] Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada, July 2017. URL https://aclanthology.org/P17-1147/.

[31] Orion Weller, Benjamin Chang, Sean MacAvaney, Kyle Lo, Arman Cohan, Benjamin Van Durme, Dawn Lawrie, and Luca Soldaini. FollowIR: Evaluating and teaching information retrieval models to follow instructions. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 11926–11942, Albuquerque, New Mexico, April 2025. URL https://aclanthology.org/2025.naacl-long.597/.

[32] Hanseok Oh, Hyunji Lee, Seonghyeon Ye, Haebin Shin, Hansol Jang, Changwook Jun, and Minjoon Seo. Instructir: A benchmark for instruction following of information retrieval models. *arXiv preprint arXiv:2402.14334*, 2024. URL https://arxiv.org/pdf/2402.14334.

[33] Chenghao Xiao, G Thomas Hudson, and Noura Al Moubayed. Rar-b: Reasoning as retrieval benchmark. *arXiv preprint arXiv:2404.06347*, 2024. URL https://arxiv.org/pdf/2404.06347.

[34] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, Minneapolis, Minnesota, June 2019. URL https://aclanthology.org/N19-1423/.

[35] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy, July 2019. URL https://aclanthology.org/P19-1612/.

[36] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 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, November 2020. URL https://aclanthology.org/2020.emnlp-main.550/.

[37] Yongqi Li, Wenjie Li, and Liqiang Nie. Dynamic graph reasoning for conversational open-domain question answering. *ACM Transactions on Information Systems*, 40(4):1–24, 2022.

[38] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=zeFrfgyZln.

[39] Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5835–5847, Online, June 2021. URL https://aclanthology.org/2021.naacl-main.466/.

[40] Luyu Gao and Jamie Callan. Condenser: a pre-training architecture for dense retrieval. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 981–993, Online and Punta Cana, Dominican Republic, November 2021. URL https://aclanthology.org/2021.emnlp-main.75/.

[41] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. SimLM: Pre-training with representation bottleneck for dense passage retrieval. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2244–2258, Toronto, Canada, July 2023. URL https://aclanthology.org/2023.acl-long.125/.

[42] Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia, May 2023. URL https://aclanthology.org/2023.eacl-main.148/.

[43] Xin Zhang, Yanzhao Zhang, Dingkun Long, Wen Xie, Ziqi Dai, Jialong Tang, Huan Lin, Baosong Yang, Pengjun Xie, Fei Huang, Meishan Zhang, Wenjie Li, and Min Zhang. mGTE: Generalized long-context text representation and reranking models for multilingual text retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1393–1412, Miami, Florida, US, November 2024. URL https://aclanthology.org/2024.emnlp-industry.103/.

[44] Niklas Muennighoff. Sgpt: Gpt sentence embeddings for semantic search. *arXiv preprint arXiv:2202.08904*, 2022. URL https://arxiv.org/pdf/2202.08904.

[45] Xin Zhang, Zehan Li, Yanzhao Zhang, Dingkun Long, Pengjun Xie, Meishan Zhang, and Min Zhang. Language models are universal embedders. In *Proceedings of the 1st Joint Workshop on Large Language Models and Structure Modeling (XLLM 2025)*, pages 252–265, Vienna, Austria, August 2025. URL https://aclanthology.org/2025.xllm-1.21/.

[46] Zhuyun Dai, Vincent Y Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith B Hall, and Ming-Wei Chang. Promptagator: Few-shot dense retrieval from 8 examples. *arXiv preprint arXiv:2209.11755*, 2022. URL https://arxiv.org/abs/2209.11755.

[47] Hossein A. Rahmani, Nick Craswell, Emine Yilmaz, Bhaskar Mitra, and Daniel Campos. Synthetic test collections for retrieval evaluation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2647–2651, Washington DC, USA, 2024. URL https://doi.org/10.1145/3626772.3657942.

Technical Appendices

A Boarder Impact

From a broader impact perspective, our benchmark and insights hold the potential to significantly influence future research directions in information retrieval. By offering a comprehensive dataset and analytical framework, we aim to advance the development of more robust retrievers that efficiently handle complex queries across diverse domains. Our contributions could enable more inclusive access to information, empowering diverse audiences to leverage massive semi-structured datasets.

We will have several safeguards to ensure the responsible release of our data. (1) We will provide a detailed description of the data generation process, including the prompts used for LLMs and the criteria for selection. This transparency will help researchers understand the limitations and potential biases in the data. (2) We will include a disclaimer in our dataset release, highlighting the potential biases and limitations of LLM-generated data. This will encourage users to critically evaluate the data and its applicability to their specific use cases. (3) We will actively seek feedback from the research community on our dataset and its impact, allowing us to address any concerns or issues that may arise.

B Dataset Detail

B.1 LLM Prompts

B.2 Candidates Recall

To build a retrieval test set, we need to annotate all relevant documents from the document collection for each query, which serve as test labels in the conventional sense. Specifically, for each query-document pair (q, d), we need their relevance label, where 0 indicates irrelevant and 1 indicates relevant. We then record the identifiers of all documents labeled as 1 to form the test set. However, examining all possible pairs is computationally expensive and impractical. Following common practice, we use a committee of multiple retrievers to recall potentially relevant documents. After reducing the number of candidates, we then proceed with relevance annotation either through LLM evaluation or human assessment.

We select several representative models from two major types of dense retrievers as the retriever committee. Ther are Drama-1B, GTE-Qwen2-7B, E5-mistral-7B, InstructOR-large, BGE-large-en-v1.5, and Jina-v3. To better capture the dataset characteristics, we also add our two fine-tuned retrievers (Qwen2.5-0.5B and Qwen2.5-1.5B) to the committee. We use each one of these eight retrievers to recall top-100 documents from the in-schema collection for each q. We consider a document as a candidate only if it appears in the top-100 results of more than $C = 4$ retrievers. For each q we finally obtain about 50 candidates.

B.3 Annotation Guideline

In the human annotation process, we provide one query-document pair and ask the annotator to label the pair as relevant (value: 1), irrelevant (value: -1), or undecided (value: 0). For the irrelevant and undecided labels, we further ask the annotator write a clear description about the reason. Bellow we provide the detailed guideline, and present the annotation interface screenshot in Figure 9.

Definitions: Query: A natural language search request that typically contains one or more constraints, such as specific date ranges or particular attributes. Candidate: A string representing a JSON object that may or may not meet the search requirements, requiring human judgment for annotation.

Prompt for Schema Generation

```
Please design {{N}} data schemas in the domain of {{DOMAIN}}. Each schema
needs to meet the following requirements:

1. **Output Format**: Adhere strictly to valid JSON format. Each schema
should be a single JSON object.
2. **Top-Level Structure**: The schema itself is a JSON object, with
top-level keys representing different fields or attributes of the document.
3. **Detailed Definition of Attributes**: Each value corresponding to a
top-level key must be a JSON object that details the attribute. This
description object must include the following three fields:
  - key_name: (string) The actual English name (key name) of the attribute,
  using snake_case naming convention.
  - data_type: (string) The data type of the attribute. To ensure diversity,
  actively try to use various types, such as: (1) Basic types: string,
  integer, float, boolean. (2) Date and time types: date (e.g., "YYYY-MM-DD"),
  datetime (e.g., "YYYY-MM-DDTHH:mm:ssZ") (3) Complex types: array (please
  specify the type of array elements if possible, e.g., array<string>,
  array<integer>, array<object>), object (used for nested structures).
  - description: (string) A clear and concise description that explains the
  purpose, meaning, or format of the attribute. The description should aid in
  understanding the field and generating such data.
4. **Attribute Quantity and Type**: The generated schema should include 5 to
20 or more different top-level attributes. The attributes can be of three
types: (1) Multiple schemas should share some common fundamental attributes
within the domain; (2) Each schema should have specific necessary attributes;
(3) Variable attributes of complex types for handling internal variations
within the schema.
5. **Considerations for Diversity and Authenticity**:
  - **Domain Diversity**: Each time a schema is generated, try to select a
  different real-world application or data sub-domain than the previous
  example.
  - **Structural Diversity**: Attempt to include varying levels of structural
  complexity in different generated results. For instance, some schemas may
  focus on basic types, while others may include `array` or `object` forms of
  nested data.
  - **Type Diversity**: Ensure that the combinations of `data_type` used are
  diverse and not consistently limited to `string` and `integer`.
  - **Logical Coherence**: The designed schemas should have internal logic
  and reasonably represent the typical structure of documents within the
  selected domain.
  - **Authenticity and Reasonableness**: The attributes in the designed
  schemas must be realistic and meaningful for data search in actual
  scenarios.

**Work Objectives**

First, provide a list of schemas and a brief description of each to ensure
domain diversity. Return a JSON array where each element is a string.


[[IN THE SAME THREAD]]

For the generated {{N}} schemas, thoughtfully consider and specify 3 to 15
required attribute fields (the number varying for each schema). Finally,
output the JSON definition of the schema, including the name of this schema
taken from the previous JSON array.
```

Figure 5: Prompt for Schema Generation

Prompt for Schema Generation with Shared Attributes

```
Please design {N} data schemas ......
......

**Work Objectives**

First, provide a list of schemas and a brief description of each to ensure
domain diversity. Return a JSON array where each element is a string.
Next, think deeply about the basic common attributes shared across these
schemas in the {{DOMAIN}} domain and return a JSON object.


[[IN THE SAME THREAD]]

For the generated {{N}} schemas, thoughtfully consider and specify 3 to 15
required attribute fields (the number varying for each schema). Finally,
output the JSON definition of the schema, including the name of this schema
taken from the previous JSON array and excluding the already determined basic
common attributes.
```

Figure 6: The different part of the prompt for Schema Generation with shared attributes.

Annotation Tasks 1: Overall Relevance Assessment For each data point, annotators must understand the constraints in the current query and evaluate whether the provided candidate meets these requirements. There are three possible labels:

- 1: Fully satisfies all conditions
- 0: Partially satisfies conditions (for multiple conditions, some conditions are missing or cannot be determined, while others are met)
- -1: Does not satisfy (if any single condition is not met among multiple conditions, it is deemed unsatisfactory. For example, in the above case, the candidate's field 'attributes page_count' value of 832 exceeds the query's requirement of 'under 500 pages')

Note: -1 (Does not satisfy) takes precedence; when both unsatisfactory and undeterminable conditions exist, the overall annotation should be "does not satisfy"

Annotation Tasks 2: Detailed Annotation If the overall annotation is "0 Partially satisfies" or "-1 Does not satisfy", two additional annotations are required:

- Missing Conditions (list of strings copied from the query): Typically occurs when either: (1) The candidate lacks corresponding condition information (2) The query condition is ambiguous and cannot be evaluated (e.g., "within the last 30 days" without specifying the current date)
- Failed Conditions (list of strings copied from the query, e.g., ["I want books that aren't too long - under 500 pages would be ideal"]) and their corresponding Failed Reasons (list of strings matching the order of failed conditions, e.g., ["The candidate's attributes page_count value of 832 exceeds the query requirement of under 500 pages"])

Important Notes: Each condition can generally be classified into three states: satisfied, not satisfied, or cannot be determined. Most conditions in the query should correspond to specific field(s) in the JSON object. If no correspondence can be found, the condition should be marked as "cannot be determined"

B.4 Statistics

We list the data statistics of each schema in Table 8.

Prompt for Data Triplet Generation

```
Schema:
{SCHEMA_DEF}

Based on the provided semi-structured document **Schema** for {{SCHEMA}} of
{{DOMAIN}} domain, please generate a JSON object containing the following
three essential key fields:

1. **query (string):**
   - A clear, natural language query directed at the schema structure of the
   document.
   - The query must include {{NUM_EXACT}} specific filtering condition (e.g.,
   searching for a particular attribute value, numerical range, boolean
   status, exclusion of a specific attribute, etc.). These conditions should
   target significant fields and be realistic.
   - The query must also include {{NUM_SEMANTIC}} vague or semantically based
   filtering condition. This condition requires the model to understand the
   meaning or intention behind words, not just keyword matching.
   {{QUERY_DEATIL}}

2. **positive (object):**
   - A **fully compliant** data instance according to the schema structure.
   - This instance must **strictly meet** all the filtering conditions
   defined in the query.

3. **negative (object):**
   - A data instance that **fully adheres** to the schema structure (or is
   very similar in structure).
   - This instance should be **relevant to the theme** of the query but
   **clearly fail** to meet at least one key filtering condition stated in
   the query.
   - The goal is to create a "hard negative" example that might be mistaken
   as a match but actually isn't.


**Ensure:**
* The query should be expressed clearly and naturally, its content should be
reasonable and fit real-world scenarios and user motivations. For example, in
a product search scenario, users typically won't set a price filter higher
than a certain amount; in a recruitment scenario, users usually prefer
candidates with lower salary expectations.
* The generated `positive` and `negative` objects should be based on
reasonable examples of the Schema.
* The relationships between `positive` and `negative` and the `query` should
be clear and meet the above definitions.
* Output only the JSON content without any explanations.
{{WORD_ABBREVIATION}}
{{DIFFERENT_WRITING_FOR_NUMBER_AND_DATE}}
```

Figure 7: The prompt for Data Triplet Generation.

Prompt for Query-Document Relevance Judgment

```
You are an expert evaluator. Your task is to determine if a given CANDIDATE
(a structured JSON object) fully satisfies a QUERY (a natural language
description of desired criteria). Output '<1>' if the CANDIDATE meets ALL
criteria in the QUERY; otherwise, output '<0>'. You must provide a brief,
clear "reason" for your decision.
Instructions & Constraints:
 1. Strict Matching: The CANDIDATE must satisfy all conditions in the QUERY
 to receive a label of '<1>'. If any condition is not met, the label should
 be '<0>'. Be precise in your evaluation of each condition.
 2. Temporal Reasoning: For time-related conditions in the QUERY, carefully
 evaluate any temporal constraints against time found within the CANDIDATE.
 These constraints may include: Relative time periods (e.g., "within last 30
 days"), Specific date ranges (e.g., "between 2020-01-01 and 2022-12-31"),
 Duration requirements (e.g., "lasting at least 6 months"). Identify the
 relevant time fields within the CANDIDATE and the given time in QUERY, and
 apply the appropriate temporal logic based on the specific requirements in
 the QUERY.
 3. Combined Criteria: Some criteria in the QUERY may require evaluating
 multiple fields in the CANDIDATE. Identify the fields needed to assess the
 criterion and combine their values or properties as required. For example,
 determining "substantial funding" might require considering multiple
 financial fields within the CANDIDATE or an array of funding_sources.
 4. Numerical Calculations: Perform any necessary numerical calculations
 (e.g., sums, averages) to evaluate conditions described in the QUERY.
 Clearly state the calculations performed in the "reason".
 5. Missing Information: If the CANDIDATE is missing information required to
 evaluate a specific condition in the QUERY, err on the side of caution and
 assume the condition is not met (label '<0>'). Clearly state what
 information is missing in the "reason". If a field exists but is null or
 empty, treat it as missing.


Output format:
reason: [brief explanation of your reasoning for your three output value]
label: [<0> or <1>]


QUERY:
{{QUERY}}

CANDIDATE:
{{CANDIDATE}}
```

Figure 8: The prompt for Query-Document Relevance Judgment.

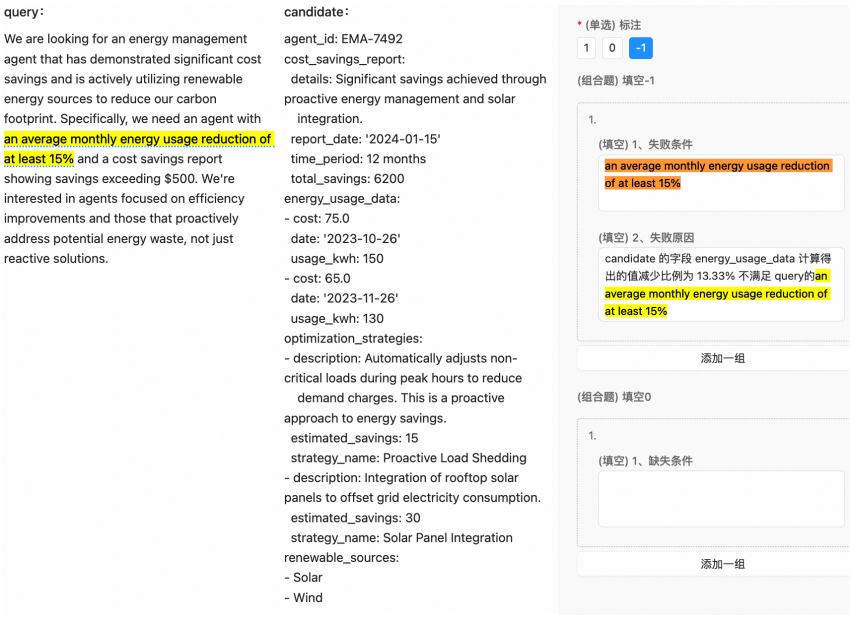

Figure 9: Annotation interface screenshot.

Table 8: Statistics of our dataset. `Pos.` denotes the positive document that meets the query.

Domain	Schema	#Documents	#Query	Pos. per Query
product_search	_	4338366	2401	13.97
human_resources	_	1440000	802	15.13
resume_search	_	1439978	798	11.81
llm_agent_and_tool	_	2878180	1595	11.04
Academic	_	2734920	1896	20.12
Finance_and_Economics	_	1438706	993	12.76
Detail				
Academic	academic_collabo	144000	100	21.77
Academic	academic_competi	143990	97	18.49
Academic	academic_journal	144000	100	27.00
Academic	academic_profile	143994	100	9.80
Academic	academic_worksho	144000	100	16.65
Academic	books_and_monogr	144000	100	23.84
Academic	conference_paper	143984	100	22.79
Academic	course_materials	144000	100	15.65
Academic	digital_learning	143998	100	13.28
Academic	educational_surv	143998	100	23.29
Academic	journal_articles	143032	100	19.06
Academic	laboratory_proto	144000	100	24.59
Academic	patents	143994	100	20.15
Academic	preprints	143948	100	20.62
Academic	research_data	144000	100	16.44
Academic	research_grants	144000	100	27.59
Academic	research_tutoria	144000	100	19.97
Academic	scholarly_events	143998	99	16.70
Academic	theses_and_disse	143984	100	24.60
Finance_and_Economics	banking_transact	144000	100	13.43
Finance_and_Economics	corporate_financ	143998	100	12.88
Finance_and_Economics	cryptocurrency_a	143992	99	14.09

Finance_and_Economics	economic_indicat	143986	97	10.25
Finance_and_Economics	financial_risk_a	144000	100	12.27
Finance_and_Economics	insurance_policy	144000	100	17.28
Finance_and_Economics	international_tr	142738	99	17.24
Finance_and_Economics	investment_portf	143992	99	10.14
Finance_and_Economics	personal_finance	144000	100	13.01
Finance_and_Economics	stock_market_dat	144000	99	6.94
human_resources	employee_benefit	144000	81	9.15
human_resources	employee_feedbac	144000	80	11.59
human_resources	employee_informa	144000	80	13.55
human_resources	hr_policies:_doc	144000	80	34.80
human_resources	leave_management	144000	80	13.70
human_resources	payroll:_financi	144000	80	7.68
human_resources	performance_revi	144000	80	10.34
human_resources	recruitment:_inf	144000	81	25.93
human_resources	time_tracking:_r	144000	80	10.73
human_resources	training_and_dev	144000	80	13.78
llm_agent_and_tool	api_usage_analyt	143978	80	10.63
llm_agent_and_tool	chat_history_ana	143028	80	19.85
llm_agent_and_tool	content_creation	144000	80	10.41
llm_agent_and_tool	content_generati	143984	80	12.15
llm_agent_and_tool	customer_support	144000	80	10.00
llm_agent_and_tool	data_security_an	144000	78	7.04
llm_agent_and_tool	e-commerce_recom	143944	80	6.56
llm_agent_and_tool	education_tutori	144000	80	8.74
llm_agent_and_tool	energy_managemen	144000	80	10.03
llm_agent_and_tool	error_and_feedba	143992	80	13.68
llm_agent_and_tool	financial_adviso	144000	79	9.94
llm_agent_and_tool	healthcare_suppo	143936	79	8.18
llm_agent_and_tool	legal_research_a	143930	80	15.08
llm_agent_and_tool	model_performanc	143976	80	13.28
llm_agent_and_tool	research_assista	143980	80	13.78
llm_agent_and_tool	sentiment_analys	144000	80	9.65
llm_agent_and_tool	training_data_ma	143994	79	13.15
llm_agent_and_tool	user_behavior_tr	143852	80	7.76
llm_agent_and_tool	version_control_	143586	80	13.94
llm_agent_and_tool	virtual_assistan	144000	80	6.81
product_search	art_and_craft_su	144000	80	8.73
product_search	automotive_searc	144000	80	6.99
product_search	baby_products_se	143948	80	13.15
product_search	beauty_products_	143998	80	14.68
product_search	books_search	144000	80	10.46
product_search	camera_search	143998	80	16.88
product_search	electronics_sear	163996	80	10.80
product_search	fashion_search	143992	80	15.93
product_search	furniture_search	144000	80	10.40
product_search	garden_supplies_	144000	80	10.13
product_search	gift_items_searc	143998	80	22.88
product_search	grocery_search	143968	80	10.60
product_search	health_products_	143858	81	20.12
product_search	home_appliances_	143996	80	10.49
product_search	hotel_and_travel	144000	80	16.80
product_search	jewelry_search	144000	80	12.69
product_search	junior_wearables	143998	80	19.38
product_search	medical_supplies	143988	80	24.01
product_search	mobile_devices_s	144000	79	8.23
product_search	music_and_videos	144000	80	13.73
product_search	office_supplies_	143998	80	18.60
product_search	pet_supplies_sea	143998	80	16.56

product_search	real_estate_sear	144000	80	18.58
product_search	restaurant_searc	144000	80	12.05
product_search	smart_devices_se	143942	80	19.08
product_search	software_product	143990	81	14.49
product_search	sports_equipment	142708	80	9.19
product_search	tools_and_hardwa	143992	80	16.21
product_search	toys_and_games_s	144000	80	7.05
product_search	watches_search	144000	80	10.20
resume_search	data_science:_re	143988	80	13.18
resume_search	education_&_teac	143998	80	12.44
resume_search	engineering:_res	143998	80	15.08
resume_search	finance_&_accoun	144000	80	10.44
resume_search	graphic_design:_	144000	80	9.71
resume_search	healthcare:_resu	143998	80	11.84
resume_search	human_resources:	144000	79	10.34
resume_search	project_manageme	143996	80	15.44
resume_search	sales_&_marketin	144000	80	7.89
resume_search	software_enginee	144000	79	11.70

C Evaluation Detail

C.1 Document Modeling

Here we use three examples to illustrate the difference between our tested three types of document types, *i.e.,* Json, Json Compact, and YAML.

JSON:

```
{
    "name": "Jack",
    "age": 42
    "tags": [
        "writer",
        "student"
    ]
}
```

JSON Compact:

```
{"name":"Jack","age":42,"tags":["writer","student"]}
```

YAML:

```
name: Jack
age: 42
tags:
  - writer
   - student
```

C.2 Qwen Fine-Tuning

To validate the quality of our generated data, we attempted to train retriever models using data that wasn't selected for the test set, and evaluated their performance. Specifically, we first sampled 1M triplets from the generated data as our training set.

We employ standard contrastive learning with InfoNCE loss for training, utilizing the generated positives and negatives as labels, and enabled in-batch negatives. Each batch is sampled from a single

domain to ensure that in-batch negatives wouldn't be too trivial. We set the batch size to 104 and learning rate to 1e-5, training for one epoch on four A800 GPUs. The models were initialized from Qwen2.5-1.5B-Instruct and Qwen2.5-0.5B-Instruct, using '<|endoftext|>' as the final token and its corresponding output state as the embedding for both training and retrieval.

C.3 Computing Resource

For data triplet generation, we run Qwen2.5-32B-Instruct model on multiple A800 GPUs via VLLM. Generating all data in Table 8 costs about 1400 GPU hours.

For Qwen2.5 model fine-tuning, the 0.5B model needs about 2 hours on 4 A800 GPUs, and 1.5B model needs 12 hours as we use gradcache to keep the same batch size.

For model evaluation, we run all models in FP16 precision. The 7B-level model takes 64 GPU hours to encode all documents. And the 1B-level model takes 16 GPU hours. All models totally cost about 340 GPU hours.

C.4 Case Study

We present some examples of retrieval results from different models.

Query: "Find a software engineer with experience in Python and JavaScript, seeking a salary between 80000 and 120000, and who is available to start within the next month. I'm looking for candidates with a strong project history. [[The current time is 2024-03-01]]"

Below are correct (relevant) docs from NV-Embed and fine-tuned Qwen2.5:


```
"NV-Embed-v2": {
  "_doc_id": "human_resources–software_enginee–p–44723",
  "technical_expertise": ["Python", "JavaScript", "React", "Node.js", "AWS", "Docker"],
   "name": "Alice Johnson", "current_position": "Software Engineer", "expected_salary":
110000,
  "relevant_projects": [
    { "name": "Cloud Migration Project", "description": "Migrated legacy applications to AWS,
improving scalability and reducing costs. Utilized Python for scripting and automation.",
"role": "Lead Developer" },
    { "name": "Real-time Dashboard", "description": "Developed a real-time dashboard using
React and Node.js for monitoring system performance.", "role": "Frontend Developer" } ],
  "experience": [ { "company": "Tech Solutions Inc.", "title": "Software Engineer", "duration":
"3 years" } ],
  "skills": ["Python", "JavaScript", "React", "Node.js", "AWS", "Cloud Computing", "Problem
Solving"], "availability": "2024-03-15",
  "objective": "Seeking a challenging software engineering role where I can leverage my skills
in Python, JavaScript, and cloud technologies to build innovative solutions."
}
```

```
"Qwen2.5_1.5B": {
 "_doc_id": "human_resources–software_enginee–p–69629",
 "technical_expertise": [ "Python", "JavaScript", "React", "Node.js", "SQL"],
 "name": "Alice Smith", "current_position": "Software Engineer", "expected_salary": 100000,
 "relevant_projects": [
    { "name": "E-commerce Platform", "description": "Developed a fully functional e-commerce platform using React, Node.js, and PostgreSQL.", "role": "Full-stack Developer" },
   { "name": "Data Analysis Tool", "description": "Created a tool for analyzing large datasets using Python and Pandas.", "role": "Data Engineer" } ],
"education": [ { "degree": "Bachelor of Science in Computer Science", "university": "Example University", "graduation_year": 2020 } ],
 "experience": [ { "company": "Tech Solutions Inc.", "title": "Software Engineer", "duration": "2020-Present" } ],
 "skills": [ "Python", "JavaScript", "React", "Node.js", "SQL", "Git", "Agile Development"],
 "certifications": ["AWS Certified Developer - Associate"],
 "objective": "Seeking a challenging software engineering role where I can utilize my skills and experience to contribute to innovative projects.",
 "languages": ["English"], "availability": "2024-03-15"
}
```


Then irrelevant doc results from Drama (“strong project” in query is not meet) and Jina-v3 (“a salary between 80000 and 120000” in query is not meet):


```
"drama": {
 "_doc_id": "human_resources–software_enginee–n–48888",
 "technical_expertise": [ "Python", "C++", "JavaScript", "SQL"],
 "name": "Jane Doe", "current_position": "Software Engineer", "expected_salary": 110000, "github_profile": null,
 "relevant_projects": [{ "name": "Web Application Development", "description": "Developed a web application using JavaScript and React.", "role": "Frontend Developer" } ],
 "education": [ { "degree": "Bachelor of Arts in Computer Science", "university": "State University", "graduation_year": 2021 } ],
 "experience": [ { "company": "Beta Inc", "title": "Software Engineer", "duration": "2021 - Present" } ],
 "skills": [ "Python", "JavaScript", "Web Development", "SQL"],
 "objective": "Looking for a growth opportunity in software engineering.",
 "languages": ["English", "Spanish"], "availability": "2024-04-15"
}
```



```
"Jina-v3" : {
 "_doc_id": "human_resources–software_enginee–n–1030",
 "technical_expertise": [ "Python", "Django", "JavaScript", "React", "SQL"],
 "name": "Bob Williams", "expected_salary": 160000,
 "experience": [ { "title": "Software Engineer", "company": "StartupX", "duration": "2020 - Present", "description": "Developed web applications using Python and Django. Focused on front-end development and user interface design." } ],
   "skills": [ "Web Development", "Front-End Development", "Python", "Django", "JavaScript"],
 "objective": "Seeking a challenging software engineering position where I can utilize my skills in web development and contribute to innovative projects."
 "availability": "2024-04-01",
}
```

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Discussed in section 7

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: This paper does not include theoretical results

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See section 3, 4 and Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: See submission details for the links of data and code.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: See section 4.1 and Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: This is a benchmark paper and all retrieval experiments are determinstic.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.

- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conform with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: See section A.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: See section A.

    Guidelines:

    - The answer NA means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All assets used in the paper are properly credited.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: See the huggingface dataset page.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: See Appendix.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [Yes]

    Justification: We have obtained IRB approval for data curation and annotation.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: See section 3.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.