TARGET: Benchmarking Table Retrieval for Generative Tasks

Xingyu Ji*[1], Parker Glenn[2], Aditya G. Parameswaran[1], Madelon Hulsebos[3]
[1]UC Berkeley [2]Capital One [3]CWI

Abstract

The data landscape is rich with structured data, often of high value to organizations, driving important applications in data analysis and machine learning. Recent progress in representation learning and generative models for such data has led to the development of natural language interfaces to structured data, including those leveraging text-to-SQL. Contextualizing interactions, either through conversational interfaces or agentic components, in structured data through retrieval-augmented generation can provide substantial benefits in the form of freshness, accuracy, and comprehensiveness of answers. The key question is: how do we retrieve the right table(s) for the analytical query or task at hand? To this end, we introduce TARGET: a benchmark for evaluating **TA**ble **R**etrieval for **GE**nerative **T**asks. With TARGET we analyze the retrieval performance of different retrievers in isolation, as well as their impact on downstream tasks. We find that dense embedding-based retrievers far outperform a BM25 baseline which is less effective than it is for retrieval over unstructured text. We also surface the sensitivity of retrievers across various metadata (e.g., missing table titles), and demonstrate a stark variation of retrieval performance across datasets and tasks. TARGET is available at `https://target-benchmark.github.io`.

1 Introduction

Large Language Models (LLMs) have become an indispensable tool in the knowledge worker's arsenal, providing a treasure trove of information at one's fingertips. Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) further extends the capabilities of these LLMs by grounding generic dialog using information from external data stores. Despite progress in long-context LLMs, RAG still provides benefits in cost and inference time (Li et al., 2024b; Yu et al., 2024). Moreover, it allows us to augment generic, off-the-shelf LLMs with proprietary data they haven't been trained on. Progress on RAG has largely been enabled by benchmarks that help exhaustively evaluate the effectiveness of various methods (Yang et al., 2024; Muennighoff et al., 2023).

While RAG has been extensively explored for free-form text, this is unfortunately not the case for structured data, stored either in relational databases or otherwise. Prior work has shown that structured data is of a different nature, for example regarding data types and dimensionality, requiring dedicated research (Cong et al., 2023). Moreover investigating retrieval of structured data for RAG is important: contextualizing LLMs using frequently updated statistical data sources, such as Data Commons (Guha et al., 2023), or using proprietary relational databases within organizations, can yield rich dividends (Radhakrishnan et al., 2024), all underscoring the need for better models, approaches and evaluation for retrieval over structured data.

Another important motivation for research on table retrieval stems from research on LLM-powered interfaces and agentic systems for processing and querying structured data. Most research in this direction, e.g., for question answering (Nan et al., 2022) or text-to-SQL (Gao et al., 2024), assumes that a table or relational database is provided, while identifying the relevant table is, in fact, a non-trivial task for a user (or agent). Figure 1 depicts an end-to-end pipeline as we envision: starting with a natural language query (which can be a "lookup" or analytical question), the first step is to interpret and augment the query, for which the retrieval component identifies the relevant tabular data needed to generate a response (which can be in code, natural language, or other format). We find that table retrieval in end-to-end (analytical) query systems is an understudied area, motivating the creation of a benchmark.

*Correspondence to madelon.hulsebos@cwi.nl and jixy2012@berkeley.edu

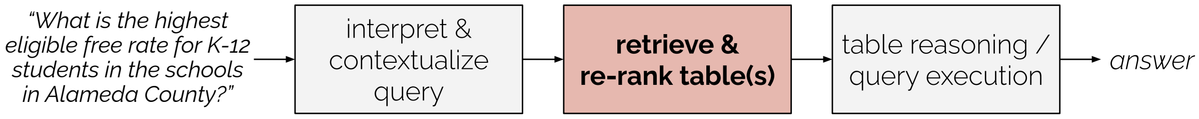

"What is the highest eligible free rate for K-12 students in the schools in Alameda County?"
interpret & contextualize query
retrieve & re-rank table(s)
table reasoning / query execution
answer

Figure 1: Pipeline of "open domain" question answering over tabular data, in which no tables containing "evidence" for the question are provided. Most research considers the "closed domain" setting, and focuses on query interpretation and augmentation, or table reasoning or query generation (e.g. text-to-SQL). With TARGET we intend to stimulate and facilitate research on the critical retrieval step in the "open domain" setting.

While there has been initial work exploring open-domain question answering on public table corpora such as Wikipedia (Chen et al., 2021; Herzig et al., 2021), this does not represent the full spectrum of data characteristics and tasks for structured data retrieval. The development of a broad and comprehensive benchmark covering diverse tasks and datasets of varying difficulty is therefore key in advancing retrieval systems for structured data.

In this paper, we present TARGET: *the first benchmark evaluating Table Retrieval for Generative Tasks*. With TARGET we provide a consistent and comprehensive framework for evaluating models and pipelines for table retrieval in isolation, as well as end-to-end for downstream tasks. We use TARGET to analyze retrieval methods based on sparse lexical representations (Chen et al., 2021), dense embeddings of metadata (Liu, 2022), dense table embeddings (Zhang et al., 2025), and dense row embeddings (Kumar et al., 2023). We find that sparse lexical representations are far less effective for retrieval over tabular data as it is found to be for rich free-form text (Muennighoff et al., 2023). In our analysis with TARGET, we find that dense table- and row- embeddings (Zhang et al., 2025) outperform baselines but still show high variation in performance across tasks and datasets. Finally, we highlight the sensitivity of retrievers to the provided metadata inputs (e.g., web page titles) and table data availability (e.g., embedding full tables, column names only, or generated table summaries). Our findings identify a performance gap in retrieval accuracy and robustness across data and tasks, emphasizing the need for more research in this area for which TARGET is an instrumental stepping stone.

2 Related Work

Representation Learning and LLMs for Tables Tables have recently become a modality of interest for representation learning and generative models for tasks such as table understanding (Hulsebos et al., 2019; Deng et al., 2022), fact verification (Herzig et al., 2020; Zhang et al., 2020), and question answering (Herzig et al., 2020), and more recently text-to-SQL (Gao et al., 2024). These models either deploy LLMs out-of-the-box for tabular data, or develop tailored architectures to capture the properties of tables, which pose specific challenges (Cong et al., 2023). These models typically take one or more tables and a query as input to generate an answer, however, the relevant tables per query can be difficult to identify. TARGET is intended to close this gap and facilitate research on end-to-end querying over tabular data such as text-to-SQL and question answering.

Table Retrieval Retrieval of structured data has been studied across use-cases in data management and machine learning. Dataset search where the objective is to find a dataset for a given task (e.g. training a machine learning model or doin data analysis) is a well studied topic in the data management literature (Halevy et al., 2016; Castelo et al., 2021). These table retrieval systems typically take a semantic description of the data as input and return the relevant tables. In TARGET we focus on retrieval components embedded into end-to-end query systems, where input queries are natural language queries and the task is to provide an accurate response based on relevant data that first needs to be retrieved in an end-to-end manner. Such pipelines have mainly been studied for open-domain question answering, typically over web table corpora (Chen et al., 2021; Herzig et al., 2021; Wang and Castro Fernandez, 2023). We include OTTQA (Chen et al., 2021), a sparse lexical retriever, as a baseline for open-domain QA. We also integrate two commonly used datasets for open domain table QA (FeTaQA (Nan et al., 2022) and OTTQA (Chen et al., 2021)) into TARGET. We introduce two new end-to-end query tasks: fact verification and text-to-SQL, which are typically not considered in the "open-domain" setting but assume the relevant data is provided by a user.

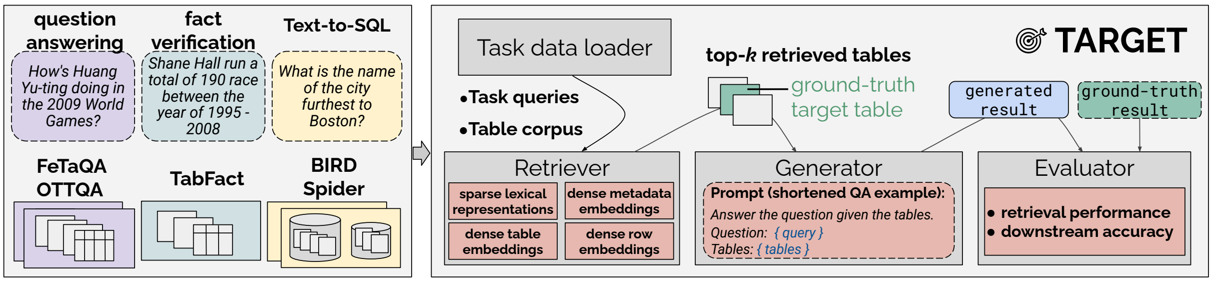


Figure 2: Overview of the TARGET benchmark for evaluating table retrieval methods and downstream generation for various datasets across three downstream tasks: tabular question answering, fact verification, and Text-to-SQL.

Benchmarks and Datasets To develop stronger rerievers and advance research on LLM-driven tasks on structured data, benchmarks and datasets are essential. The MTEB and CRAG benchmarks (Muennighoff et al., 2023; Yang et al., 2024) have been instrumental in benchmarking text embedding quality and RAG over rich text documents. We need similar benchmarks for retrieval systems and embedding models for structured data. In prior research, useful datasets were introduced to evaluate various tasks for relational data, such as TabFact (Chen et al., 2020), FeTaQA (Nan et al., 2022), and Spider (Yu et al., 2018). These datasets focus on evaluating methods for a specific downstream task only, i.e., given a table or database, answer natural language queries about it, without integrating the critical task of retrieval. TARGET addresses this gap by focusing on the evaluation of table retrieval performance while incorporating existing task-specific datasets.

3 The TARGET Benchmark

We describe the datasets, tasks, metrics, and retrievers that make up the TARGET benchmark. All resources for use and extension of TARGET are available at `https://anonymous.4open.science/r/target-B782`[1].

3.1 Benchmark Design

The pipeline of TARGET aligns with typical RAG pipelines (Figure 2). TARGET takes as inputs the corpus with tables/databases and queries (a natural language question or statement). Data loading and evaluation are abstracted away such that custom core components of RAG pipelines, i.e., the `Retriever` and `Generator` can easily be evaluated when aligned with the TARGET API. The retriever, which can be basic or advanced (Gao et al., 2023), identifies the relevant table(s)/database(s) for an input query. Depending on needs, retrievers can either manage corpus embedding independently or leverage vector databases (Malkov and Yashunin, 2018; Qdrant) integrated in TARGET. Given the tables and query, the generator yields a response which is then evaluated with respect to the ground-truth.

Table 1: Tasks and Evaluation Metrics in TARGET.

Task	Evaluation Metrics
Table retrieval	recall (R@k), capped recall (CR@k) avg. retrieval time (s)
Question answering	SacreBleu (SB))
Fact verification	precision (P), recall (R), F1-score (F1)
Text-to-SQL	execution accuracy (EX)[2]

3.2 Tasks & Metrics

Per source dataset, we combine all tables and any available metadata into a retrieval corpus. For all tasks, e.g., question answering, we evaluate the retriever and generator outputs using metrics from the original dataset papers or that are widely adopted. An overview of the tasks and metrics in TARGET can be found in Table 1.

Table Retrieval Table retrieval task assesses retrieval performance in isolation and is the first step for end-to-end downstream evaluation. Retrieval performance is measured with recall@top-k, reflecting the successful retrieval of the ground-truth table within the top-k retrieved tables. In the text-to-SQL setting, however, standard recall may yield unintuitive results, as multiple ground-truth tables might be needed to generate the valid SQL query. With T_i representing the ground-truth tables for the ith query, we correct for situations where $k \ll |T_i|$ and follow Thakur et al. (2021) in evaluating capped recall by setting our denominator to $min(k, |T_i|)$. Additionally, we include the average retrieval time per query.

[1] The URL of the project page with links to datasets and code will be linked upon publication.

Question Answering Given the retrieved tables contents and the input question, an answer is generated and evaluated against the ground-truth natural language answer for accuracy and comprehensiveness. We report SacreBleu (Post, 2018) to reflect syntactic similarity across generated tokens.

Fact Verification Given the retrieved tables, the generator either accepts or refutes a natural language statement, or acknowledges that insufficient information is provided. Here, the accuracy is measured through precision, recall and F1.

Text-to-SQL We adapt a prompt template from Talaei et al. (2024), which incorporates the natural language question and the schemas of the retrieved tables along with generation instructions. The prompt instructs the generator to output a concise "chain-of-thought" reasoning trace (Nye et al., 2021; Wei et al., 2022) to support more robust query generation. Additionally, since the retrieved tables may belong to different databases, the generator is required to include the selected database alongside the SQL query to ensure proper execution. The execution result from the generated SQL are then compared to that of the ground-truth SQL. We report the execution accuracy, aggregated across query complexity categories, following the implementation in BIRD (Li et al., 2024a)[2].

3.3 Datasets

Data and Label Sources The datasets of each task in TARGET can be found in Table 2. All publicly available splits of each dataset are included except for BIRD's train split. We use the test splits of included datasets for our evaluations. For OTTQA and BIRD, where test splits are unavailable, validation splits are used.

To ensure consistency across datasets, e.g. for consistent data processing, we standardize the schemas of the files holding the datasets. Each dataset has a "corpus" and a "queries" file. The "corpus" files contain the table contents and table identifiers (IDs), wherein each entry corresponds to a single table and includes a "context" field for metadata, if available. For instance, in the text-to-SQL datasets, the context field contains primary key, foreign keys, and other table schema information. The "queries" files contain the queries, query IDs, and the ground-truth table ID(s).

To evaluate retrieval for text-to-SQL, we extract all the tables referenced in the ground-truth query using sqlglot and consider them as ground-truth.

Table 2: Dimensions of included tabular datasets per task across splits in TARGET.

Task	Dataset	Split	Corpus size	# queries
Question Answering	OTTQA	train	8.1K tables	41.5K
		val	789 tables	2.2K
	FeTaQA	train	7.3K tables	7.3K
		val	1K tables	1K
		test	2K tables	2K
Fact Verif.	TabFact	train	13.2K tables	92.3K
		val	1.7K tables	12.8k
		test	1.7K tables	12.8K
Text-to-SQL	Spider	train	146 DBs (2K tables)	8.7K
		val	20 DBs (1K tables)	1K
		test	40 DBs (1K tables)	2.1K
	BIRD	val	11 DBs (75 tables)	1.5K

Data Complexity Tables across datasets differ significantly in size. For example, text-to-SQL datasets feature significantly larger tables compared to other datasets in TARGET. Although BIRD's validation split contains fewer tables and databases overall, the large size of each table poses a significant challenge for retrieval systems. Specifically, the average number of rows per table in BIRD is 52.4k, nearly 10x compared to 5.3k rows per table in Spider. In contrast, tables in FeTaQA, OTTQA, and TabFact range from 10 to 50 rows. The distributions of row and column counts per dataset can be found in Appendix A.

Another distinction across datasets is the availability of metadata. Unlike text-to-SQL datasets, which feature descriptive table names and database schema, FeTaQA and TabFact does not provide informative table titles (for example, "2-1570274-4.html.csv" from TabFact) or grouping by databases. This requires retrieval methods to effectively use tabular data contents or devise data augmentation methods.

3.4 Retrievers

We present our analysis with TARGET for retriever methods that reflect common design principles in research and industry. We evaluate dense semantic embeddings and sparse lexical representations, and vary the inputs provided: tables or rows, with or without table metadata, and metadata-only. Text-to-SQL has small changes to the retriever and generator, as explained in Appendix B.

[2] See evaluation_ex.py

No Context baseline LLMs are capable of memorizing facts from the data that they were trained on (Mallen et al., 2023). To understand the influence of memorization on downstream task responses, the LLM-based generator is asked to respond based soely on its internal knowledge without any retrieved tables provided. We refer to this setting as the "No Context" baseline.

Sparse Lexical Representation The Sparse Lexical Representation retriever resembles the OTTQA approach (Chen et al., 2021). It constructs a TF-IDF matrix of the corpus , which may use TF-IDF term weights or BM25. It takes as input the column names, table rows, and, table metadata such as the (Wikipedia) page title. On retrieval, a query is converted into a TF-IDF-weighted vector for which the dot product is calculated with the table representations to find the k-most similar tables.

Dense Metadata Embedding While metadata such as titles and descriptions can provide context for retrieval, they are either uninformative (e.g. "8c4c-4f0d.csv") or entirely absent in many tables. To this end, the Dense Metadata Embedding retriever creates table summaries following three steps, ① generate a table name and summary of each table with GPT-4o-mini[3] using the column names and first 10 rows of the table, ② embed the table metadata with `text-embedding-ada-002`, and ③ retrieve relevant tables based on the cosine similarity between natural language query and metadata embedding. We use the open-source LlamaIndex library, commonly used in practice, to store the embeddings in an in-memory key-value index and retrieve using cosine similarity (Liu, 2022).

Dense Table Embedding We compare three dense embedding models: `text-embedding-3-small` (OpenAI, 2024), `stella_en_400M_v5` (Zhang et al., 2025), and `multilingual-e5-large-instruct` (Wang et al., 2024)[4]. The latter two are open-weight models available on HuggingFace. We evaluate the performance for embeddings of only column names versus column names along with 100 rows. While formatting tables as json appeared better for GPT-3.5 (Singha et al., 2023), markdown formatting yields better results. Each row of the table is formatted in markdown's tabular syntax and sequentially appended to form a single concatenated string for embedding. For retrieval, the input query is embedded with the same model, and the top-k tables are retrieved based on cosine similarity.

Dense Row-level Embedding The input query might semantically correspond to values of certain rows within tables. Alternative approaches, therefore, devise retrieval through row-level embeddings (Zhang et al., 2023; Kumar et al., 2023; Wang and Castro Fernandez, 2023). In this baseline, each row is serialized into a sentence following the template "[column name]$_i$ is [cell value]$_i$, [column name]$_j$ is [cell value]$_j$" (Zhang et al., 2023), for example, "first name is John, last name is Doe". The serialized rows are embedded using the relatively small and effective `stella_en_400M_v5` embedding model (435M parameters). Upon retrieval, the input query is embedded with the same model and used for retrieving rows with the highest cosine-similarity with the input query embedding. Based on the retrieved rows, the corresponding top-k tables are retrieved. Row-wise retrieval via dense embeddings can become impractical for very large tables with hundreds of thousands of rows, for example those included in BIRD. Therefore, this baseline is not evaluated for the BIRD dataset.

3.5 Generators

We use basic LLM prompts for downstream tasks to evaluate the GPT-4o-mini model[3] in our experiments (Hurst et al., 2024). However, we design the TARGET API to enable evaluations of other language models and advanced generation pipelines.

The `Instruction` prompt takes in: ① task instructions, ② the top-k retrieved table(s) or database schemas of retrieved tables (for text-to-SQL), and ③ the query. Unless otherwise specified, we serialize all tables in prompts to markdown strings. An example prompt for the question answering task is provided below. The full prompt templates can be found in Appendix B.

```
Use the provided table(s) to answer the question. Yield
a concise answer to the question.
If none of the tables provide relevant information, use
your knowledge base to generate an answer - but only if
you are confident in the answer's factuality.
If the neither the tables nor your knowledge can be used
to answer the question reliably, say that not enough
information is provided.
Tables: {table_contents}
Question: {query}
```

[3] gpt-4o-mini-2024-07-18

[4] Experiments with the `tapas-large` model (Herzig et al., 2020) illustrated that this BERT-based table-specialized embedding model is not competitive as-is for retrieval.

4 Results

Table 3 presents the performances of the evaluated retrievers with k set to 10. Figure 3 illustrates the average retrieval recall over various values of k across datasets. For the Sparse Lexical Retriever, only the performance using BM25 is included as its performance is similar to TF-IDF.

4.1 Retrieval Insights

How do different table representations perform? We find that table retrieval based on sparse lexical representations such as BM25 (OTTQA) are less effective, across tasks and datasets, than they are for text (Muennighoff et al., 2023), even with increased k (Table 3). The strong performance of the sparse lexical retrievers with table title on the OTTQA dataset (recall@10 of 0.967 and 0.963) can be attributed to the high correspondence between Wikipedia table titles and the questions, as manually verified[5]. The performance drops for BM25 and TF-IDF to 0.592 and 0.583 respectively, if the table title is not included. The importance of descriptive metadata for retrievers based on lexical representations is confirmed by their low performances on FeTaQA and TabFact, where descriptive table titles are not available. LLM-generated table summaries with dense metadata embeddings can significantly improve retrieval performance as illustrated by the Dense Metadata Embedding baseline.

Dense Table Embeddings (with column names and rows included in the embeddings) generally yield the best performance. Different embedding models demonstrate similar performance across datasets, with `stella_en_400M_v5` achieving the best results, showing itself to be a viable open-source, lightweight, and efficient option (Table 4). Notably, for both text-to-SQL datasets, the effect of including data rows is minimal, with differences within $\pm 5\%$ in recall. Inspection confirms that (analytical) queries in text-to-SQL datasets typically have high resemblance with schemas (column names). In contrast, for the question answering and fact verification tasks, retrieval performances are significantly curtailed when only the column names are embedded.

The Dense Row-level Embedding method exhibits comparable performances to dense embeddings of tables with sampled rows. On Question Answering datasets, row-level retrieval does not improve performance compared to dense table embeddings, while for Text-to-SQL and Fact Verification it lightly outperforms other baselines. However, due to large size tables for Text-to-SQL datasets, the vast search space significantly hinders retrieval efficiency. With relatively small performance gains, row-level embeddings may not be practical for large-scale table retrieval.

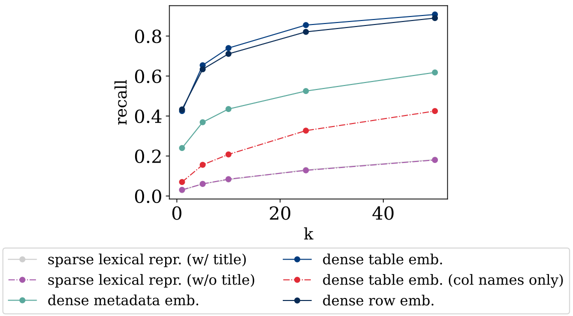


Figure 3: Influence of k on retrieval performance with various baselines on the FeTaQA dataset, confirming the expectation that performance gradually increases with k, most significantly for dense embedding approaches.

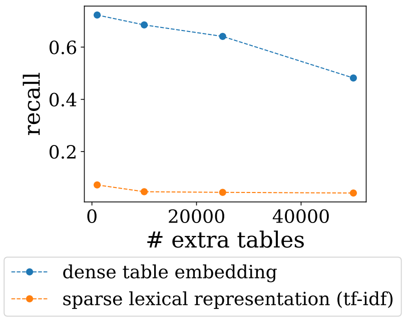


Figure 4: Influence of corpus size on retrieval, illustrating the sensitivity in retrieval performance of dense retrievers when the corpus reaches a large scale.

How important is table metadata for retrieval? From our analysis of the retrieval results of methods based on sparse lexical representations (OTTQA TF-IDF and BM25), we conclude that descriptive metadata (e.g. table summaries or titles) can be key for lexical retrievers. We observe a similar sensitivity for lexical representations for semantic metadata on the Text-to-SQL tasks when table names are not included, which is further confirmed with results on FeTaQA, where the provided table titles are not descriptive (e.g. "example-10461") and including them does not enhance performance. The importance of metadata is also highlighted in the strong performance of the dense metadata embedding method compared to the dense table

[5] The queries and tables can be explored at: `https://ott-qa.github.io/explore.html`

Table 3: Results with TARGET for table retrieval with k=10. R@k stands for recall@k, CR@k stands for capped recall @k (Thakur et al., 2021), and s for average retrieval time in seconds. For the Dense Table Embedding baseline, we report the best performing model `stella_en_400M_v5`. Best scores are in **bold**, second-best underlined.

	Question Answering				**Fact Verification**		**Text-to-SQL**			
	OTTQA		**FeTaQA**		**TabFact**		**Spider**		**BIRD**	
Method	R@10	time (s)	R@10	time (s)	R@10	s	CR@10	time (s)	CR@10	time (s)
Sparse Lexical Repr. (BM25)	**0.967**	0.001	0.082	0.001	0.338	0.001	0.544	0.001	0.700	0.001
w/o table title	0.592	0.001	0.084	0.001	0.331	0.001	0.491	0.001	0.616	0.001
Sparse Lexical Repr. (TF-IDF)	0.963	0.001	0.083	0.001	0.336	0.001	0.541	0.001	0.586	0.001
w/o table title	0.583	0.001	0.039	0.001	0.322	0.001	0.489	0.001	0.613	0.001
Dense Metadata Embedding	0.820	0.297	0.436	0.396	0.469	0.354	0.621	0.024	0.940	0.014
Dense Table Embedding	0.963	0.001	**0.741**	0.001	0.824	0.001	0.657	0.001	**0.961**	0.003
column names only	0.658	0.001	0.208	0.001	0.506	0.001	0.648	0.001	0.932	0.003
Dense Row-level Embedding	0.951	0.267	0.711	0.394	**0.848**	0.384	**0.665**	6.077	N/A	N/A

Table 4: Table Retrieval Performances of Dense Table Embedding with Various Text Embedding Models with k=10. Best scores are in **bold**, second-best underlined.

	Question Answering		**Fact Verif.**	**Text-to-SQL**	
	OTTQA	**FeTaQA**	**TabFact**	**Spider**	**BIRD**
Method	R@10	R@10	R@10	CR@10	CR@10
text-embedding-3-small	0.950	0.722	0.779	0.618	0.858
column names only	0.601	0.184	0.452	0.635	0.908
stella_en_400M_v5	**0.963**	**0.741**	**0.824**	**0.657**	**0.961**
column names only	0.658	0.208	0.506	0.648	0.932
multilingual-e5-large-instruct	0.918	0.655	0.741	0.620	0.894
column names only	0.549	0.188	0.430	0.613	0.909

embedding method for text-to-SQL.

How does scale affect retrieval performance? First, we assess the impact of the number of retrieved tables, i.e. by increasing k. As Figure 3 shows, average recall gradually increases with k for all retrievers, which is expected. The lexical retrievers do not gain significant performance improvements upon retrieving more tables.

Another influential variable is the size of the retrieval corpus. To analyze this, we evaluate the retrieval performance as corpus size increases, by appending tables from the GitTables dataset (Hulsebos et al., 2023). Here we zoom in on the FeTaQA dataset, which initially consists of 2K tables. We study the impact of corpus size on retrieval performances of the sparse lexical baseline based on TF-IDF and the dense table embedding baseline. As Figure 4 shows, retrieval performance decreases as the corpus size grows. For the dense table embedding baseline, which generally exhibits the best performance across tasks, the drop becomes progressively more noticeable once the corpus exceeds 10K added tables. Performance degradations on large corpora illustrates a need for developing table retrievers that remain robust at scale.

4.2 Generator insights

Can LLMs execute tabular tasks from memory? In general, the "No Context" baseline performs significantly lower without having relevant tables provided (Table 5). An exception to this is the low performance of sparse lexical retrievers on FeTaQA, which we discuss in the next section. Without grounding LLMs in relevant structured data to answer domain-specific questions, factuality and quality of generation becomes unreliable. Additionally, Table 5 also emphasizes that database schemas for text-to-SQL are critical to generate accurate SQL queries, as the "No Context" baseline yields an accuracy of 0.

Does generation benefit from table retrieval? The low performance of all retrievers on the OTTQA dataset is notable (all SacreBleu scores are below 1), which we hypothesize is due to the relatively short answers in OTTQA versus longer generated answers despite prompting for conciseness. In comparison to the "No Context" baseline, where the model is asked to generate answers solely based on its knowledge base, providing retrieved tables in context increases downstream performances notably, as exemplified by results for FeTaQA and TabFact. Due to the stronger retriever performance of dense embeddings, we find that dense retrievers generally yield best downstream performance across datasets. Meanwhile, the poor retrieval performance of sparse lexical representations on FeTaQA seems to distract the generator with irrelevant tables, leading to a significant decrease in SacreBleu scores compared to the "No Context"

Table 5: Results with TARGET for downstream tasks corresponding upfront table retrieval with k=10. SB stands for SacreBleu, EX for execution accuracy aggregated over all query complexity categories. P/R/F1 reflect precision, recall, and f1 scores. For Dense Table Embedding, we report the results of the best performing embedding model `stella_en_400M_v5` Best scores are in **bold**, second-best underlined.

	Question Answering		Fact Verification	Text-to-SQL	
	OTTQA (SB)	**FeTaQA (SB)**	**TabFact (P/R/F1)**	**Spider (EX)**	**BIRD (EX)**
No Context	0.146	6.761	0.59/0.19/0.25	0	0
Sparse Lexical Repr. (BM25)	0.475	1.618	0.64/0.25/0.33	0.444	0.076
Sparse Lexical Repr. (TF-IDF)	**0.510**	1.586	0.64/0.25/0.33	0.440	0.183
Dense Metadata Embedding	0.476	11.027	0.63/0.34/0.40	0.556	0.266
Dense Table Embedding	0.486	12.569	0.64/0.47/0.49	0.588	**0.291**
Dense Row-level Embedding	0.469	**13.231**	**0.64/0.48/0.50**	**0.599**	N/A

baseline. Ensuring the inclusion of relevant tables in the LLM's context is crucial for reliable downstream generation quality, highlighting the need for robust retrieval methods.

Can long-context LLMs replace table retrieval? An alternative for retrieval-augmented generation (RAG) is to exhaust the context of LLMs by including vast amounts of tables from the corpus without fine-grained retrieval, and rely on the LLM to extract the answer from a large set of tables. To understand the limitations of LLM context for table comprehension tasks, we explore the relationship between the rank of the ground-truth table in the retrieval results and downstream task performance in Figure 5. Treating instances where the ground-truth table failed to appear in the top-10 retrieval results as the lowest rank, we see a strong negative correlation (average Spearman's ρ = -0.85) between retriever performance and downstream task performance[6]. These results 1) motivate work on improved table retrieval and reranking, and 2) indicate that careful attention in crafting table retrievers is more effective than relying on providing a large number of tables into long-context LLMs.

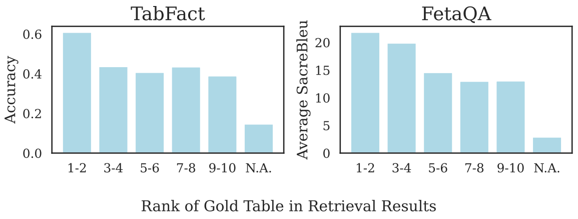


Figure 5: As the rank of the ground-truth table increases (i.e. the table is deeper in the prompt), the performance on downstream tasks tends to decrease. "N.A." indicates that the ground-truth table was not retrieved.

5 Conclusion

Retrieval is key in LLM-powered query systems over structured data as well as for grounding dialog and interactions with LLMs in up-to-date, factual, structured data. With both categories of use-cases in mind, we present TARGET: the first benchmark for Table Retrieval for Generative Tasks. With TARGET we extend the "open-domain" query setting beyond question answering, i.e. for fact verification and text-to-SQL, to evaluate end-to-end query systems that can encompass basic retrieval and generation, as well as more complex, potentially agentic, multi-step pipelines. We evaluate various retrieval methods based on different representations of tabular data, and find that sparse lexical representations are not as robust as found for retrieval over text. Instead, dense embeddings of tables or their metadata are critical for identifying the relevant tables for given queries, with the impact of different types of metadata being an important aspect to study further. We evaluate end-to-end query performance as well, and find that table retrieval significantly improves the accuracy of LLM generations across tasks and datasets. Through deeper analysis with TARGET we surface the importance for more robust table retrievers as retrieval performance declines for large corpus sizes, whereas generation accuracy is affected by the position of the relevant table provided in the context.

[6] Due to the multi-table retrieval setting of text-to-SQL, we only consider Question Answering and Fact Verfication here.

6 Acknowledgments

We acknowledge support from the Dutch Research Council (NWO) grant NGF.1607.22.045, a BIDS Accenture Fellowship, the National Science Foundation (grants DGE-2243822, IIS-2129008, IIS-1940759, IIS-1940757), and EPIC lab sponsors (G-Research, Adobe, Microsoft, Google).

7 Limitations

TARGET does not incorporate all existing table retrieval methods due to lacking source code and tuned models, but the baselines included reflect the main method categories, varying in representation (e.g. dense or sparse) and inputs (e.g. with and without table metadata, or metadata-only). We make it straightforward to evaluate custom retrieval methods with the TARGET API, which is inspired by the MTEB benchmark for text embedding evaluation (Muennighoff et al., 2023). We encourage researchers to directly integrate their retrieval methods into TARGET to establish a comprehensive and consistent ground for evaluation.

We also note that relational databases can be large, hence, necessitate more fine-grained retrieval (i.e. row and column selection (Chen et al., 2024)). In TARGET, we extract ground-truth labels of the relevant database and tables for text-to-SQL datasets enabling two-step retriever evaluations (retrieving database first, then the relevant tables within that database). While ground-truth column- and row-level labels of relevance for a given query are lacking, 'in-table' retrieval can be evaluated indirectly by assessing generation performance for retrieved table fragments.

Finally, upon inspecting the generation results we observed a discrepancy between the conciseness of the ground-truth answers (e.g. in OTTQA (Chen et al., 2021)) and the comprehensiveness of the generated answers by GPT-4o, despite instructing for concise responses. The SacreBleu score can be sensitive to such differences. The BERTScore (Zhang* et al., 2020) metric is commonly used for evaluating long-form QA systems as it measures semantic similarity, but our evaluations with this metric provided no discriminative signal across retrievers, as the generated answer might semantically overlap regardless of being correct with the "pointwise" answer, this motivates further development of suitable metrics to evaluate long-form answers.

References

Sonia Castelo, Rémi Rampin, Aécio Santos, Aline Bessa, Fernando Chirigati, and Juliana Freire. 2021. Auctus: A dataset search engine for data discovery and augmentation. *Proceedings of the VLDB Endowment*, 14(12):2791–2794.

Si-An Chen, Lesly Miculicich, Julian Martin Eisenschlos, Zifeng Wang, Zilong Wang, Yanfei Chen, Yasuhisa Fujii, Hsuan-Tien Lin, Chen-Yu Lee, and Tomas Pfister. 2024. TableRAG: Million-token table understanding with language models. *Thirty-eighth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Yang Wang, and William W Cohen. 2021. Open question answering over tables and text. In *International Conference on Learning Representations*.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020. TabFact: A large-scale dataset for table-based fact verification. *International Conference of Learning Representations*.

Tianji Cong, Madelon Hulsebos, Zhenjie Sun, Paul Groth, and H. V. Jagadish. 2023. Observatory: Characterizing embeddings of relational tables. *Proceedings of VLDB*, 17(4).

Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. 2022. Turl: Table understanding through representation learning. *ACM SIGMOD Record*, 51(1).

Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, and et al. 2024. Text-to-sql empowered by large language models: A benchmark evaluation. *Proceedings of VLDB*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Ramanathan V Guha, Prashanth Radhakrishnan, Bo Xu, Wei Sun, Carolyn Au, Ajai Tirumali, Muhammad J Amjad, Samantha Piekos, Natalie Diaz, Jennifer Chen, et al. 2023. Data commons. *arXiv preprint arXiv:2309.13054*.

Alon Halevy, Flip Korn, Natalya F Noy, Christopher Olston, Neoklis Polyzotis, Sudip Roy, and Steven Euijong Whang. 2016. Goods: Organizing google's datasets. In *Proceedings of the 2016 International Conference on Management of Data*, pages 795–806.

Jonathan Herzig, Thomas Mueller, Syrine Krichene, and Julian Eisenschlos. 2021. Open domain question answering over tables via dense retrieval. In *Proceedings of NAACL*.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos.

2020. Tapas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Madelon Hulsebos, Çagatay Demiralp, and Paul Groth. 2023. Gittables: A large-scale corpus of relational tables. *Proceedings of the ACM on Management of Data*, 1(1).

Madelon Hulsebos, Kevin Hu, Michiel Bakker, Emanuel Zgraggen, Arvind Satyanarayan, Tim Kraska, Çagatay Demiralp, and César Hidalgo. 2019. Sherlock: A deep learning approach to semantic data type detection. In *Proceedings of the 25th ACM SIGKDD*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Vishwajeet Kumar, Yash Gupta, Saneem Ahmed Chemmengath, Jaydeep Sen, Soumen Chakrabarti, Samarth Bharadwaj, and Feifei Pan. 2023. Multi-row, multi-span distant supervision for table+ text question answering. In *The 61st Annual Meeting Of The Association For Computational Linguistics*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, et al. 2024a. Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. *Advances in Neural Information Processing Systems*, 36.

Zhuowan Li, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. 2024b. Retrieval augmented generation or long-context llms? a comprehensive study and hybrid approach. *arXiv preprint arXiv:2407.16833*.

Jerry Liu. 2022. LlamaIndex.

Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4).

Alex Troy Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *The 61st Annual Meeting Of The Association For Computational Linguistics*.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. Mteb: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037.

Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryściński, Hailey Schoelkopf, Riley Kong, Xiangru Tang, et al. 2022. Fetaqa: Free-form table question answering. *Transactions of the Association for Computational Linguistics*, 10.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. 2021. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*.

OpenAI. 2024. text-embedding-3-small. https://platform.openai.com/docs/guides/embeddings. Embedding Model.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Qdrant. Qdrant: Vector database for the next generation of ai. Vector database engine.

Prashanth Radhakrishnan, Jennifer Chen, Bo Xu, Prem Ramaswami, Hannah Pho, Adriana Olmos, James Manyika, and RV Guha. 2024. Knowing when to ask-bridging large language models and data. *Data Commons*.

Ananya Singha, José Cambronero, Sumit Gulwani, Vu Le, and Chris Parnin. 2023. Tabular representation, noisy operators, and impacts on table structure understanding tasks in llms. *Table Representation Learning workshop at NeurIPS*.

Shayan Talaei, Mohammadreza Pourreza, Yu-Chen Chang, Azalia Mirhoseini, and Amin Saberi. 2024. Chess: Contextual harnessing for efficient sql synthesis. *arXiv preprint arXiv:2405.16755*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. 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)*.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Multilingual e5 text embeddings: A technical report. *Preprint*, arXiv:2402.05672.

Qiming Wang and Raul Castro Fernandez. 2023. Solo: Data discovery using natural language questions via a self-supervised approach. *Proceedings of the ACM on Management of Data*, 1(4):1–27.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Daniel Gui, Ziran Will Jiang, Ziyu Jiang, et al. 2024. Crag–comprehensive rag benchmark. *arXiv preprint arXiv:2406.04744*.

Tan Yu, Anbang Xu, and Rama Akkiraju. 2024. In defense of rag in the era of long-context language models. *arXiv preprint arXiv:2409.01666*.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887*.

Dun Zhang, Jiacheng Li, Ziyang Zeng, and Fulong Wang. 2025. Jasper and stella: distillation of sota embedding models. *Preprint*, arXiv:2412.19048.

Hongzhi Zhang, Yingyao Wang, Sirui Wang, Xuezhi Cao, Fuzheng Zhang, and Zhongyuan Wang. 2020. Table fact verification with structure-aware transformer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1624–1629.

Tianping Zhang, Shaowen Wang, Shuicheng Yan, Li Jian, and Qian Liu. 2023. Generative table pretraining empowers models for tabular prediction. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14836–14854.

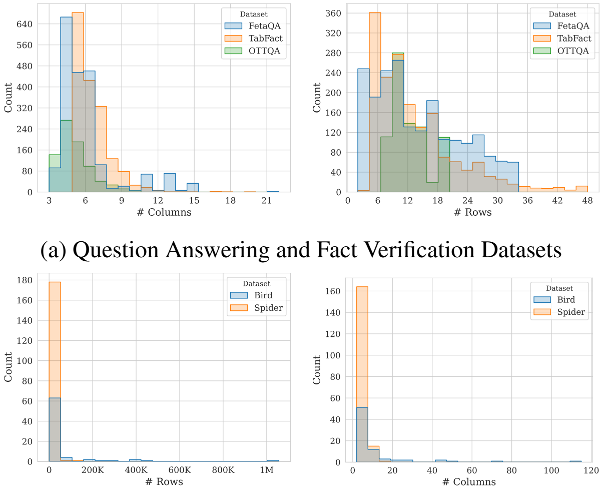
Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Alex Zhuang, Ge Zhang, Tianyu Zheng, Xinrun Du, Junjie Wang, Weiming Ren, Stephen W Huang, Jie Fu, Xiang Yue, and Wenhu Chen. 2024. Structlm: Towards building generalist models for structured knowledge grounding. *Conference on Language Modeling*.

Appendix

A Data characteristics

Besides differentiation in task queries and corpus sizes, the datasets in TARGET present various distinctive properties in terms of data distributions such as table dimensions as shown in Figure 6.



(a) Question Answering and Fact Verification Datasets

(b) Text-to-SQL Datasets

Figure 6: Comparison of column and row count distributions across datasets. The x-axis shows the number of columns / rows per table, and the y-axis represents the number of tables with that column / row count. Text-to-SQL datasets are separated from the Question Answering and Fact Verification datasets due to the significant differences in row count distributions.

B Generator Prompt Templates

Question Answering These prompts were adapted from (Zhuang et al., 2024).

System Prompt:

```
You are a data analyst who reads
tables to answer questions.
```

Instruction Prompt:

```
Use the provided table(s) to answer
the question. Yield a concise
answer to the question.
If none of the tables provide
relevant information, use your
knowledge base to generate an answer
– but only if you are confident in
the answer's factuality.
If the neither the tables nor your
knowledge can be used to answer the
question reliably, say that not
enough information is provided.

Tables: {table_contents}
Question: {query}
```

Fact Verification These prompts were adapted from (Zhuang et al., 2024).

System Prompt:

```
You are an expert in evaluating statements on factuality given the provided tables.
```

Instruction Prompt:

```
Given the following evidence which may take the form of sentences or a data table,determine whether the evidence supports or refutes the following statement.
If none of the tables provide relevant information, refer to your knowledge base – but only if you are confident your answer's factuality. If the neither the evidence nor your knowledge can be used to verify the statement reliably, state that there is not enough information.
Assign the statement one of three labels: True, False, Not Enough Information. Do not include anything else in your answer.

Tables: {table_contents}
Statement: {query}
```

Text-to-SQL Prompts These prompts were adapted from CHESS (Talaei et al., 2024).

To ensure the generated SQL query can be easily parsed from the generator's response (which includes both Chain of Thought Reasoning and the generated SQL), we use OpenAI's structured output API to enforce output in JSON format.

System Prompt:

```
You are an expert and very smart data analyst.
```

Instruction Prompt:

```
Below, you are presented with a database schema and a question.
Your task is to read the schema, understand the question, and generate a valid SQLite query to answer the question.
Before generating the final SQL query, think step by step on how to write the query.
Database Schema: {database_schema}
This schema offers an in-depth description of the database's architecture, detailing tables, columns, primary keys, foreign keys, and any pertinent information regarding relationships or constraints.

Question: {query}

Take a deep breath and think step by step to find the correct SQLite SQL query. If you follow all the instructions and generate the correct query, I will give you 1 million dollars.
```

No Context Instruction For "No Context" baseline evaluations, we provide the following message to the generator in place of the {table_contents} field in the instruction prompts.

```
Some or all tables are not available. Don't acknowledge the lack of information in your response. Please use your knowledge base and answer to the best of your ability, without producing false information.
```