Enhancing and Evaluating Tabular Models *on the fly* via Synthetic Question-Answer Generation

Jorge Osés Grijalba[1,3], Luis Alfonso Ureña-López[1], Eugenio Martínez Cámara[1], Jose Camacho-Collados[2]
[1]SINAI Research Group, CEATIC, University of Jaén, Spain
[2]Cardiff NLP, Cardiff University, UK [3]Law Business Research, UK
jorgeosesgrijalba@gmail.com, {laurena, emcamara}@ujaen.es,
camachocolladosj@cardiff.ac.uk

Abstract

Question Answering (QA) over Tabular Data has been traditionally a challenging task, but LLMs have recently shown the ability to respond to questions related to this type of structured data. However, current tabular QA datasets are skewed toward Wikipedia tables and SQL-style answers composed of human-crafted question-answer pairs. This limits the evaluation of LLMs on this task to a narrow genre of data and language, while also requiring extensive human effort for dataset or benchmark creation. To address this, we introduce SYNTABQA, a methodology for the automatic generation of synthetic question-answer pairs from any unannotated table. SYNTABQA defines a detailed question typology, enabling fine-grained evaluation and facilitating the creation of diverse QA datasets. Our approach not only provides an automated test bed for any tabular dataset but can also be used in few-shot settings to supply LLMs with tailored examples, improving their focus and accuracy. We validate SYNTABQA on two large, manually constructed tabular QA benchmarks of different nature.

Keywords: Tabular data, question answering, LLMs, synthetic data generation

1. Introduction

Recent advances in LLMs have significantly broadened the scope of NLP models through zero-shot and few-shot learning (Radford et al., 2019; Brown et al., 2020). These general-purpose models now handle applications ranging from sentiment analysis (Deng et al., 2023; Zhang et al., 2024b) and summarization (Zhang et al., 2024a) to more complex tasks in specialized domains, including agentic behaviors and structured reasoning (Yang et al., 2024a; Wei et al., 2022; Kang et al., 2023; Zhao et al., 2024). A particularly promising application is question answering (QA) over tabular data, where models are expected to interpret and reason over structured sources such as relational databases, spreadsheets, and web tables.

Unlike traditional QA over text (Voorhees, 2001; Rajpurkar et al., 2016; Khashabi et al., 2020), tabular QA requires understanding column semantics, combining heterogeneous data types, and leveraging implicit relationships across rows and fields (Fang et al., 2024).

Tabular QA requires a special type of data resources for model training, instruction, validation and evaluation. Despite progress, developing robust LLM systems for tabular QA remains challenging. Datasets in this area often depend on expensive human annotation, are typically limited to a domain (*e.g.* financial) or language (*e.g.* SQL), or are automatically constructed with LLMs (Chen et al., 2021b; Nan et al., 2022a; Kweon et al., 2023; Chen, 2023; Wu et al., 2025), which limits their application in unseen datasets at scale. Therefore, there is a lack of a domain-agnostic methodology to generate question-answer pairs that can be used to improve and validate LLMs in arbitrary datasets and domains. Furthermore, LLMs rely heavily on well-designed few-shot examples or code-based prompts to generalize effectively in this setting. However, manually producing these examples is time-consuming and requires familiarity with the table structure and reasoning patterns required.

In this paper, we introduce SYNTABQA, a novel method to automatically generate diverse and high-quality question-answer (QA) pairs for tabular data. These QA pairs can be used flexibly across different model interaction paradigms by leveraging key underlying structures of the data like column types and question patterns. The method is also language- and interface-agnostic, meaning it is applicable to a wide range of programming languages and execution environments used to query or reason over data requiring minimal adjustments. Our approach generates relatively simple questions that span different data types, incorporating multiple columns and realistic patterns of data interaction.

As illustrated in Figure 1, we evaluate SYNTABQA on two fundamentally different table QA benchmarks: OpenWikiTables (Kweon et al., 2023), based on Wikipedia tables, and DabaBench (Osés-Grijalba et al., 2024b), a diverse benchmark of real-world datasets. In particular, we demon-

Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026), pages 5389–5413
11–16 May 2026.

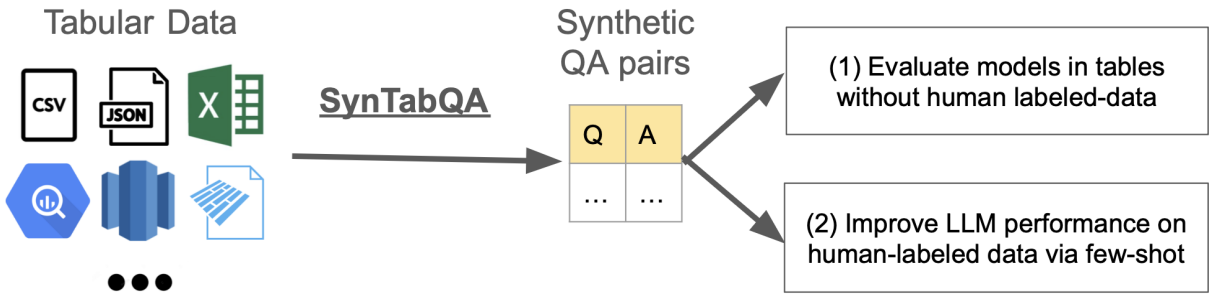

Figure 1: Overview of the two main experiments in which SynTabQA will be showcased.

strate that the generated QA pairs improve LLM performance when used as part of the model few-shot prompt. We also show how these QA can be used for online LLM evaluation and its results strongly correlated with those from the human-crafted questions from the original datasets.

Contributions. Our contributions are threefold. First, we propose SynTabQA, a novel, automatic, domain-agnostic and low-resource method to generate question-answer pairs for tabular data. Second, we demonstrate the pairs resulting from SynTabQA enhance the performance of LLMs over human-made QA pairs via custom few-shot in both in-context and code-based approaches, independently of the programming language or interface used. Third, our framework enables the instant evaluation of any table QA models on the fly.

2. Related Work

Until recently, there has been little research on processing tabular data. Nonetheless, the interest has surged in recent years due to the availability of textual data stored in tables, and the new possibilities enabled by language models to answer questions on non-textual data (Jin et al., 2022; Lu et al., 2025).

Most of the works related to tabular QA are skewed to processing data from Wikipedia, in particular the WikiTableQuestionDataset (Pasupat and Liang, 2015a), which is still used in recent work (Mouravieff et al., 2024). Other related datasets were also compiled with data from WikiPedia, as FeTaQA (Nan et al., 2022b), OpenWikiTables (Kweon et al., 2023) and WikiSQL (Zhong et al., 2017a). The release of datasets in a different domain from Wikipedia started in 2024 with the publication of the ICTQA dataset (Min et al., 2024), which is a dataset grounded in documents of information and communication technology products; DataBench (Osés-Grijalba et al., 2024b), which is a large collection of datasets of tabular data on real scenarios stored in CSV; and the multimodal MMSci dataset in the scientific domain (Yang et al., 2025). We evaluate our method on textual tabular data (Wikipedia and CSV files), excluding datasets designed for table description (*e.g.*, ICTQA) or multimodal data (*e.g.*, MMSci).

Synthetic data. Simple synthetic question generation methods have previously been used in this task, as shown in works such as Pasupat and Liang (2015b), primarily to decompose SQL queries into simpler components that could be more easily processed by earlier SQL parsers. QA datasets for tables, including (Pasupat and Liang, 2015a; Zhong et al., 2017b; Nan et al., 2022b; Zhong et al., 2017a; Pasupat and Liang, 2015c; Kweon et al., 2023), have been effective in improving model performance. However, they are limited by the fact that their questions and answers are tied to specific datasets. In contrast, we propose an agnostic approach that can be used ad-hoc in arbitrary user data.

Table QA methods. Three main approaches have been proposed in the literature: (1) representing the tables as knowledge graphs, (2) leveraging SQL sentences to generate the answers, and (3) based on the use of LLMs. Regarding the first type of approach, Pasupat and Liang (2015a) use a semantic parser for representing tables into a knowledge base. Similarly, TAG-QA Zhao et al. (2023) built upon a graph neural network a parsing method to also represent tables in a knowledge graph. In the second group, there are several works: Zhong et al. (2017b) propose a deep learning method to transform questions into SQL queries to generate the response; Chemmengath et al. (2021) follow a similar approach but in this case leveraging BERT to generate the SQL queries; finally, Pal et al. (2023a) propose MultiTabQA, a sequence to sequence model to generate SQL queries to answer the questions with tables. MultiTabQA is able to answer questions from more than one table, since it is based on SQL and relational databases. Recent advancements have also explored improving robustness and scalability in table QA. Ashury-

Example Dataset			Example Questions for Columns A, B, C		
A	**B**	**C**	**Question**	**Column Types**	**Answer**
1	2.3	apple	What is the first value of A where B $>$ 1?	number, number	1
2	0.8	banana	What is the most common value of C if B $<$ 1?	category, number	banana
3	1.5	carrot	List the values of C where B is greater than 2	category, number	[apple, banana]
4	0.4	apple	What are all of the values in A where C is 'apple'?	number, category	[1, 4]
5	2.1	banana	What is the sum of B for rows where C = 'banana'?	number, category	2.9

Table 1: Illustration of a small dataset (left) and several automatically generated question-answer pairs using SYNTABQA (right) for given columns *A*, *B*, and *C*.

Tahan et al. (2025) introduce a benchmark to evaluate model consistency across table formats, while Wang et al. (2024) enhance reasoning through evolving tables. Additionally, Su et al. (2024) propose multimodal models that handle ambiguous tables, and Yang et al. (2024b) focus on pretraining protocols to improve generalization for tabular tasks. Osés-Grijalba et al. (2024b) test the ability of LLMs to answer questions that can involve several columns of the same table. They also show the capacity of LLMs to generate programming code to response questions that require complex mathematical reasoning. Min et al. (2024) leverage LLMs to transform tables into text, which is a more suitable representation to use again LLMs to response the question with a plain text answer.

We propose a method to generate synthetic pairs of questions and answers from tabular data that can be used to improve performance in tabular QA tasks through few-shot learning. This is, to our knowledge, the first attempt of its kind that has been tested within the task of tabular QA. There are some contributions in the task of question generation (Guo et al., 2024b). For instance, Schmidt et al. (2024) proposes a method grounded in LLMs to generate synthetic questions for *textual* question answering. Instead, our question and answer generation proposal is only focused on tabular data, is independent from any LLMs or programming language, and is able to involve several columns in order to generate questions that require to deeply understand the question and to leverage a reasoning process to generate the answer.

3. SYNTABQA: Synthetic Table Question-Answer Generation Method

Tabular data consists of records (rows) defined by shared attributes (columns). Our question-answer generation approach leverages this structure and focuses on two main components: column types common to all data formats and filtering method that are common to those types. Examples of questions generated using our method from a toy dataset are included in Table 1.

Column Typing. Our method leverages a typing scheme based on standard database practices. Specifically, we use a human-adapted version of Apache Parquet's type system[1], which allows for the automatic classification of table columns without manual intervention. We rely on the open-source implementation of Lector[2]. Identifying the data type of each column is essential, since many questions in Tabular QA require specific operations such as mathematical computations, conditional filtering, or date comparisons. The column types we consider include numerical, categorical, date, URL, text, boolean, and list columns containing any of the aforementioned types.

Data Preprocessing and Ingestion. To standardize inputs across heterogeneous sources, SYNTABQA utilizes the `Lector` framework for automated schema inference. Unlike standard parsers that often perform destructive casting (e.g., converting large integer IDs to floats), `Lector` implements a multi-step detection pipeline: (1) Dialect Detection, which identifies CSV delimiters and encodings; (2) Preamble Removal, which strips metadata lines to locate the actual table header; and (3) Type Inference, which uses PyArrow's compute functions to map columns to precise types (e.g., `timestamp`, `category`, or `boolean`). This ensures that the question-generation templates are only instantiated using validated data structures, directly preventing the "nonsensical queries" (e.g., calculating the average of a URL column) that arise from raw string-matching approaches. Template examples generated for a given dataset can be found in Appendix C.

Question Typing. The type of question that can be asked depends on the data types involved in both the condition and the target of the query. To enable consistent evaluation and analysis, we label each question based on the expected type of its answer. This may be a single numeric value, a categorical value, or a list of either. These labels are applied to both the synthetic QA pairs generated

[1] https://arrow.apache.org/

[2] https://www.github.com/graphext/lector

by our system and the human-written questions used for validation.

3.1. Automatic Generation

SYNTABQA leverages the structural regularities of tabular data to automatically generate diverse and type-consistent question-answer pairs. Given a set of column *types*, the system constructs pairs of synthetic questions and answers that are contextually meaningful and semantically coherent with the data schema.

Question-answer Formulation. We define a question-answer generator as a function τ that receives a set of structured arguments and returns a pair of automatically constructed `(question, answer)`, independent of the specific mechanism used to compute the answer. Unlike existing LLM-assisted synthetic methods that rely on the model to "guess" an answer from the table, SYNTABQA utilizes a deterministic dual-generation approach (R5). For every question, the system simultaneously generates a symbolic execution trace (the Python/SQL code). The ground-truth answer is then derived by executing this code against the actual data, ensuring 100% accuracy in the synthetic supervision signal.

Considering two columns A and B as sets of $n \in \mathbb{N}$ elements, and a parameter $m \in \mathbb{N}$ with $1 \leq m \leq n$, we can formalize:

$$\tau\colon (A, B, sel, fil, b) \rightarrow (question, answer)$$

$$bool = \{1, 0\}$$

$$fil : (B, b) \rightarrow bool^n$$

$$sel : (A, bool^n) \rightarrow A \cup \mathbb{R} \cup A^m \cup \mathbb{R}^m$$

Here, the *filter* (fil) acts as a conditional operator applied to a reference column B and one of its elements $b \in B$, determining which instances of A contribute to the answer. Entries marked as 1 are retained, while those marked as 0 are excluded.

The *selector* (sel) then operates on this filtered subset of A, yielding either a single element, a subset of m elements, a numeric value, or a vector of m numeric values. These formally defined return types correspond to our column representations: *category*, *number*, *list[category]*, and *list[number]*.

Parameter Assignment. Each QA pair is generated by sampling argument values for τ from predefined yet extensible candidate sets. For every dataset, column pairs are dynamically selected. Columns containing only null values are excluded, after which one non-null value from the conditioning column and suitable operations (*selectors* and *filters*) are chosen based on the inferred column types.

Selector and Filter Operations. SYNTABQA supports a rich collection of operators to maximize compositional variety and semantic coverage across different data types.

For *numeric columns*, selectors such as `average`, `sum`, `count`, `maximum`, `minimum`, `median`, and `standard deviation` enable the generation of quantitative summaries and comparisons. Corresponding filters include `greater than`, `less than`, `equal to`, `not equal to`, `between`, as well as ranking-based filters such as `top-`k and `bottom-`k, which select subsets based on ordered magnitude. For *categorical columns*, SYNTABQA employs selectors such as `unique`, `first`, `last`, `mode`, and `count per category`, allowing aggregation and frequency-based reasoning. The associated filters include `is`, `is not`, `in set`, `starts with`, and `contains`, which capture equality and substring-based constraints. When operating on *temporal* (date or time) columns, SYNTABQA can apply selectors like `earliest`, `latest`, `duration`, and `average date`, supporting chronological or interval-based queries. Temporal filters include `before`, `after`, `on`, `between dates`, and `in year/month`, enabling flexible temporal conditioning. Finally, for *text columns*, the framework provides selectors such as `longest`, `shortest`, `most frequent term`, and `contains keyword`, which facilitate linguistic and descriptive reasoning. Complementary filters like `contains`, `does not contain`, `matches pattern`, and `starts with` allow fine-grained selection based on text patterns or substrings.

Together, these operator sets allow SYNTABQA to instantiate a wide range of semantically diverse questions, spanning descriptive, comparative, and conditional reasoning across heterogeneous data modalities. The complete and extensible catalog of implemented selectors and filters, including their parameter configurations and examples, will be provided in the appendices upon release.

For example, given columns *salary* (numeric) and *department* (categorical), using selector *average* and filter *is equal to* for the value *'Engineering'*, the generated question becomes:

What is the average salary of employees where department is equal to 'Engineering'?

The corresponding answer, derived from the same argument set, might be *35*. In parallel, SYNTABQA produces the executable code snippet associated with this computation, which can support code generation or serve as a supervision signal:

```
employees[employees["department"] ==
"Engineering"]["salary"].mean()
```

Although most examples use two-column interactions, the formulation generalizes naturally to multi-column settings and more complex relational dependencies.

Col. Type	#DataBench	#O. WikiT.	Example
number	788	4100	55
category	548	8964	apple
date	50	47	1970-01-01
text	46	966	A red fox...
url	31	5	example.org
boolean	18	0	True
list[number]	14	0	[1,2,3]
list[category]	5	112	[cat, dog]
list[url]	8	0	[x.com,y.org]
Columns	1615	14086	
Datasets	65	2262	

Table 2: Data types in the columns present over the DataBench and Open Wikitables table collections.

Scalability of Question Generation. SYNTABQA can generate a large and diverse set of question-answer pairs from a given dataset with minimal human intervention and low computational cost.

Formally, let N denote the total number of columns in dataset D and c the average number of non-null values per column. Assuming s selectors and f filters are available, and considering all ordered column pairs, the total number of baseline QA pairs is approximated by:

$$N_{qa}(D) = N \times (N-1) \times c \times s \times f.$$

This quantity scales combinatorially with the number of operations and columns, allowing SYNTABQA to efficiently produce extensive QA corpora for training or evaluation of table reasoning models.

4. Experimental Setup

The experimentation has a twofold aim. First, it will show that the generated question-answer pairs (see section 3) can be used to evaluate LLMs, and second, that those very same pairs can be useful to inject this kind of knowledge via few-shot learning to enhance the ability of LLMs to respond to answers from tabular data. In the following, we present the datasets involved in our experimental setup (section 4.1), along with the chosen small open-source models (section 4.2), the evaluation of the utility of the synthetic question-answers in subsection 4.3 and how we design the few-shot learning with the new synthetic pairs (section 4.4).

4.1. Data

We use two standard, diverse, and large benchmarks to evaluate the quality and utility of the generated question-answer pairs. These benchmarks differ in nature: the first, Open WikiTables, requires answer retrieval via SQL database queries, while the second, DataBench, involves generating Python code to access the data and produce the answers. General statistics for both benchmarks, including data types, are provided in Table 2.

Open WikiTables. The data released from Kweon et al. (2023) comprises the test set. It is built upon a collection of 2,262 tables extracted from Wikipedia articles (Zhong et al., 2017a; Pasupat and Liang, 2015b). The suite does not provide an answer typing system, but answers can be classified into either numbers or categorical data, which we have done in order to perform a subsequent analysis. As originally envisaged, we tackle this task with a *SQL Query Generation* approach.

DataBench. DataBench (Osés-Grijalba et al., 2024b) provides a collection of 66 datasets extracted from various sources and contains associated QA sets over them. Although the benchmark is approach-agnostic, we will approach it as a code generation task in Python, as performed in the original evaluation. The answers fall into an array of types ranging from booleans to lists of elements, and thus present a wider variety than answers from the Open Wikitables benchmark.

4.2. Models

The focus of our study will be small open-source models that can be run with comparatively modest resources. We have selected five of the most popular smaller language models, including both specialised coding models such as `codellama` (Rozière et al., 2024) or `deepseek-coder` (Guo et al., 2024a) and other general-purpose ones such as zephyr (Tunstall et al., 2023), mistral (Jiang et al., 2023a) and `openchat` (Wang et al., 2023). All of our models are GPTQ versions of 7 billion parameters with no additional fine-tuning, and we use code-generation tasks in order to help make their reasoning more understandable. Specific parameters, prompts and more detailed information can be found in Appendix A and the related GitHub repository[3].

4.3. Evaluation on Synthetic Data

For Experiment 1, we generate a synthetic set of question-answer (QA) pairs for each dataset in our collections, following the procedure described in section 3. Each model is then evaluated using the previously defined non–few-shot prompt on these automatically generated QA pairs. The generation process is highly flexible and can produce thousands of QA pairs per dataset (see subsection 3.1). Since using SYNTABQA on the whole collections from Open WikiTables and DataBench yields excessively large outputs, we limit production to 30

[3] https://github.com/jorses/syngen-tables

synthetic QA pairs per dataset. To account for variability in the generation process, we create three independent sets of 30 synthetic QA pairs per dataset. In total, for Open WikiTables (226 datasets), we produce 67,860 QA pairs across the three sets, and for DataBench (65 datasets), we produce 1,980 QA pairs.

4.4. Few-shot Evaluation

In Experiment 2, we initially benchmarked all language models with a zero-shot prompt approach over real data, which will be used as the basis for the comparison and baseline in the few-shot experiment. The basic prompt has the necessary information for the models to answer the questions, including a basic dataset representation that contains the columns and first few rows, and information on the expected format of the answer.

We then assess whether the 30 synthetic QA pairs per dataset generated in Experiment 1 can improve the model's ability to answer human-authored QA pairs. Specifically, we conduct an incremental few-shot evaluation where we inject an increasing number of synthetic examples, starting from 5 and growing in increments of 5 up to 30 per prompt, into the model's context. Each larger few-shot configuration includes all examples from the previous one. The process followed is illustrated in Figure 2.

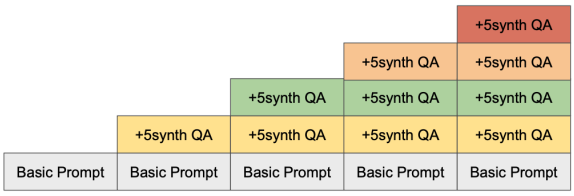

Figure 2: Incremental prompting approach followed in Experiment 2 in order to produce the results seen in Figure 3.

This setup enables us to analyze how the number of synthetic QA pairs provided as in-context examples influences model performance on real, human-created QA tasks, and to determine the optimal few-shot sample size for knowledge injection.

5. Results

In this section, we present the main results of Experiment 1 and Experiment 2.

Model	Synth. QA Pairs O. WikiTables	Synth. QA Pairs DataBench
codellama	80.60 ± 1.14	47.17 ± 0.64
deepseek	83.43 ± 0.90	63.59 ± 0.45
mistral	82.20 ± 0.14	28.65 ± 0.26
openchat	25.07 ± 0.37	36.26 ± 1.02
zephyr	28.13 ± 1.65	31.67 ± 0.39

Table 3: **Experiment 1**: Average accuracy ($\pm$ std) for each model on synthetically generated QA pairs from each table collection.

5.1. Experiment 1: Evaluation on Synthetic Data

The main results of evaluating LLMs on our synthetically-generated question-answer pairs are presented in Table 3.

Open Wikitables. The performance of comparison models is quite diverse. The models that perform better are `codellama`, `deepseek` and `mistral`, while `openchat` and `zephyr` perform much worse. These Wikipedia-based datasets have been extensively used in the table QA domain and elsewhere, which may explain the high performance of recent LLMs.

DataBench. The results seem to indicate that the synthetic questions are harder to answer in the context of code-completion than those in the Wikipedia case, so we could expect to see a higher improvement in these types of dataset in the few-shot scenario. The best model, `deepseek`, is also the best model at answering the real questions, as we will further analyse in subsection 6.1. We highlight the performance degradation of `mistral` in this case where the nature of the tables differs from traditional relational tables as the ones found in Open WikiTables.

5.2. Experiment 2: Few-Shot Evaluation on Real Data

Figure 3 shows the few-shot results for the given models and datasets in both Open Wikitables and DataBench. The zero-shot baseline is indicated by 0, and it contributes the baseline we compare our experiments to. In all cases, providing examples to the models in a few-shot fashion help, with results generally picking with five or ten example. In terms of benchmarks, DataBench seem to benefit the most from the few-shot setting in comparison to Open Wikitables. Expanded results can be found in Appendix B.

While not the main focus of the evaluation, we observe that the models that are intended for code generation like deepseek-coder tend to do better.

Open Wikitables. Figure 3 shows the performance across models has indeed benefited from few-shot. We can see that `codellama` has improved

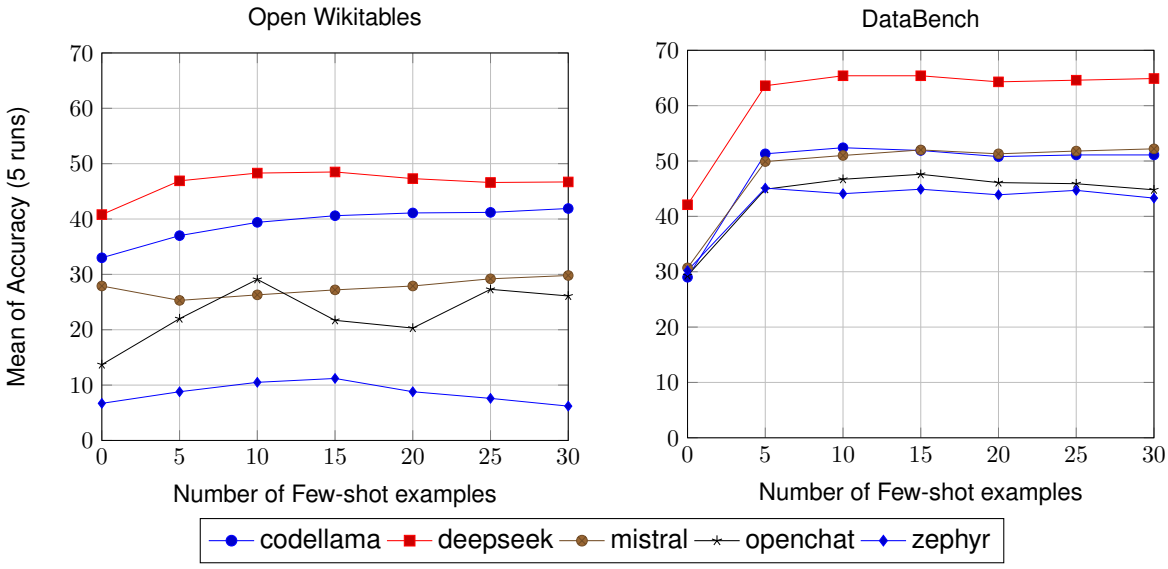

Figure 3: **Experiment 2**: Accuracy of evaluating LLMs in Open Wikitables and DataBench over human-made QA sets using zero or more synthetically generated examples in the prompt over the original human-made evaluation sets for each table collection.

from 33% to 42%, `deepseek` from 40% to 48%. `Mistral` does not seem to be considerably improving its results, only by 3%, while the smaller models seem to be benefiting the most by the strategy, doubling their previous performance and being on-par with models that performed better in the baseline. This seems to indicate that the information injected can be used to boost the performance of models that are not specialised on code or on a given task, putting them closer to specialist models that are more or less better prepared for that task. Those trained as general chat agents (`mistral`;`openchat`:`zephyr`) seem unable to fully learn from this approach in our setup. `Zephyr`'s low accuracy is mainly due to a pattern of faulty response by repetition of the same SQL query both in the baseline and few-shot runs.

DataBench. Improvements are larger than in the Wikipedia case, presenting an improvement up to 65% for `deepseek`, and both `mistral` and `codellama` getting up the 50% range, as we expected based on the results only on synthetic QA pairs (see Table 3). Most models here seem to peak around 5 to 10 questions, and performance stagnates later. It is worth noting that the task of code generation in Python is still better captured by these non-specific models than the task of answering a SQL query.

5.2.1. Question type analysis

Building on our typing system, we now analyze how different question types in the few-shot setting influence model responses. For each model and benchmark, we focus on the 10-question few-shot configuration in Figure 3, as this appears to yield the largest performance gains across all models.

DataBench. Table 5 shows that there are gains across the board for most models. Improvements are on average of 20%, with both types of lists and boolean questions (those which answer is yes or no) having much better improvements than categories and numbers. This could be because categories and numbers are on average much better results in the first place, and in fact remain on top after all improvements are made. This could indicate that there is a theorical ceiling of improvement from our approach that especially benefits those questions on which the model is underperforming, but further research is needed.

Open WikiTables. In the case of WikiTables (Table 4), overall gains are much lower on average, of 7.2%. The largest gains are achieved by OpenChat, which did worse than the others in the initial evaluation on synthetic data, and Zephyr. Since the type component is not less rich in this data suite, and there are not significant differences between types for SQL, this is not entirely surprising. Type differences are in general much smaller here than in DataBench. It remains to be seen if tackling the *Open Wikitables* set with Python instead of SQL would result in a larger type difference. The effect that is also observed here is that the larger the initial accuracy, the smaller the gain.

6. Validation of Automatic Question and Answer Generation

In section 5.2, we showed how our method can yield improvements when used in a few-shot fash-

	category	number	overall
deepseek	46.99 (+6.8)	49.74 (+6.1)	48.11 (+6.5)
codellama	33.46 (+3.2)	39.43 (+1.8)	35.84 (+2.7)
mistral	22.59 (-2.6)	33.42 (-0.1)	26.79 (-1.6)
openchat	28.68 (+23.2)	32.74 (+21.8)	30.23 (+22.9)
zephyr	10.22 (+6.4)	15.83 (+3.5)	12.19 (+5.4)
Average	28.39 (+7.4)	34.23 (+6.6)	30.63 (+7.2)

Table 4: **Experiment 2**: Detail of few-shot accuracy per Open Wikitables question type, few shot with 10 examples. Accuracy gains *vs.* baseline in parentheses.

ion for a suite of language models. In this section, we further validate our method in two ways: (1) we study how robust our automatic evaluation is with respect to standard human-created datasets (section 6.1), and (2) we perform an ablation study where synthetic questions using our methodology in the few-shot settings are replaced for generic questions (section 6.2).

6.1. Comparison with manually-crafted table QA datasets

While we claim that our synthetically-generated question and answers can be used for online LLM evaluation, it is important that we validate our approach against existing human-crafted benchmarks. As we can see in Figure 3, the hierarchy of LLMs with respect to the evaluation on the synthetic datasets we generated in Table 3 is roughly preserved. The best performing models on the synthetic data are also the best performing ones in the human-created datasets. This is especially the case in Open Wikitables for `openchat` and `zephyr` that have a poor performance in our synthetic test set and have a similarly poor zero-shot performance in the human-crafted benchmarks. In DataBench the trends are also maintained, although with smaller differences among them.The *Pearson* correlation score of the model performance between the real (zero-shot - baseline) and synthetic sets yields $r = 0.93$ in *Open Wikitables* and $r = 0.81$ in *DataBench*, taking into account all of the runs performed for the experiments, including those that are averaged to create Figure 3. These results show that the performance on synthetic data could therefore serve as a proxy for performance on real data (*online evaluation*) on its own, as question and answers can be generated ad hoc for any given tabular dataset.

6.2. Ablation study

When information about a dataset is injected into the prompt through few-shot examples, two tasks occur simultaneously. First, the model receives general guidance on how to handle datasets (few-shot learning), and second, it gains dataset-specific details through synthetic QA pair generation. The system generates variables based on column names and sampled values from the target dataset, automatically retrieving column information. However, because this combines sampled data and general instructions, the goal was to isolate the impact of sampling specific column values and make it depend only on the dataset extracting generic information about it. To achieve this, a simpler set of generic questions, such as "What is the number of rows in this dataset?" or "What is the number of columns in this dataset?" was developed. These 30 fixed questions, which do not rely on sampling or dataset structure, enable comparison of model performance to determine how much learning comes from the few-shot mechanism versus the proposed approach (section 3). Results showing the improvements from these proposed questions across both data collections are presented in Table 6. The values marked with "+" indicate the performance gains from SYNTABQA compared to a more generic few-shot approach. Overall, adding sampled table values using our proposed approach consistently yields positive effects across all models.

Paraphrase and Variations. To assess the influence of specific question wording, we tested a simple paraphrasing procedure by rephrasing the questions generated in Experiment 1 using claude-4.5-sonnet. As shown in Table 7, the results suggest that paraphrasing has little effect on question difficulty, compared to the results of Table 3. Consequently, retaining the original on-the-fly generation method allows us to avoid reliance on external models for paraphrasing without sacrificing performance, albeit at the cost of a necessarily less complex question wording.

7. Conclusions

In this paper, we presented SYNTABQA, a method to generate questions and answers automatically given any table as input. Our method relies on an organised and diverse set of heuristics that leverages structures found in all tables regardless of storage type. This method can then be used for multiple purposes, from which we focus on two aspects: (1) developing an automatically-constructed benchmark to assess the performance of existing LLMs for different types of questions, and (2) using these questions to guide an LLM to better answer questions from a given dataset through means such as in-context learning. One of the key aspects of this method is that it can be applied *on-the-fly*, i.e., for tables in any data format for which

	boolean	category	number	list[num]	list[cat]	all
codellama	45.80 (+32.8)	63.85 (+22.3)	76.25 (+14.9)	38.26 (+25.0)	38.61 (+23.2)	52.53 (+23.7)
deepseek	55.34 (+31.7)	73.46 (+4.2)	81.61 (+4.6)	60.61 (+29.6)	56.76 (+46.7)	65.54 (+23.4)
mistral	41.60 (+15.7)	64.62 (+24.2)	70.50 (+8.4)	28.79 (+12.9)	45.56 (+37.1)	50.15 (+19.6)
openchat	51.91 (+37.4)	58.46 (+13.9)	72.80 (+0.8)	32.20 (+23.9)	18.53 (+11.2)	46.78 (+17.5)
zephyr	52.67 (+12.2)	36.15 (+3.5)	67.43 (+6.9)	34.85 (+28.8)	32.05 (+20.5)	44.64 (+14.4)
average	49.46 (+26.0)	59.31 (+13.6)	73.72 (+7.1)	38.94 (+24.0)	38.30 (+27.7)	51.93 (+19.7)

Table 5: **Experiment 2:** Few-shot (n=10) accuracy per DataBench question response type. Accuracy gains *vs.* baseline in parentheses.

Model	n=5	10	15	20	25	30
Open Wikitables						
codellama	+1.2	+4.2	+5.9	+6.1	+5.2	+9.9
deepseek	+2.2	+4.2	+4.5	+3.5	+3.8	+4.1
mistral	-4.4	-2.7	-1.3	-1.7	+0.3	+0.6
openchat	+2.9	+8.9	+9.0	+7.1	+5.9	+12.3
zephyr	+1.7	+1.2	+0.1	-1.7	+0.1	-0.8
DataBench						
codellama	+9.7	+8.9	+13.2	+8.2	+14.0	+6.5
deepseek	+9.6	+7.4	+9.5	+5.1	+9.7	+7.2
mistral	+3.7	+5.4	+5.1	+6.0	+5.5	+5.8
openchat	-0.7	+2.0	-0.8	+0.3	-0.5	+3.4
zephyr	+2.8	+2.1	+2.9	+2.4	+6.7	+5.3

Table 6: **Experiment 2:** Accuracy improvements attributable to SYNTABQA versus a generic few-shot approach in DataBench and Open Wikitables.

Model	Synth. QA Pairs O. WikiTables	Synth. QA Pairs DataBench
codellama	80.60 ± 3.12	47.17 ± 1.84
deepseek	83.43 ± 2.91	63.59 ± 3.27
mistral	82.20 ± 3.73	28.65 ± 1.98
openchat	25.07 ± 2.36	36.26 ± 0.45
zephyr	28.13 ± 0.67	31.67 ± 2.21

Table 7: **Experiment 1 with paraphrases**: Average accuracy with paraphrases ($\pm$ maximum deviation over 5 runs with paraphrased questions) for each model on the synthetically generated Question-Answer pairs using SYNTABQA on Open Wikitables and Databench.

we need to answer questions quickly. The source code is available in the GitHub repository [4].

Limitations

One of the main limitations of our experimentation is the lack of experiments with LLM of more than 7 billion parameters. This is partly due to computational resources, but also due to a conscious decision to test open and relatively small models which can be potentially benefited from the approach to a larger extent. This does not mean that larger models cannot benefit from our approach, but this would need to be empirically tested. Moreover, the automatic evaluation suite proposed can be used by any type of model, which can also be viewed as a kind of unit test for different types of questions.

Moreover, all the experiments are done on English datasets, mainly due to the scarcity of data in other languages. We are planning to extend our setting to other languages — our approach is in theory language independent and therefore, low resource languages for which specific datasets do not exist could be benefitted from it.

The evaluation was based on fixed prompts, which can be viewed as a limitation due to our limited computing resources to complete all the experiments. A multi-prompt evaluation can provide more robust evidence in future evaluation.

Finally, all the evaluation is based on a fixed set of answer types. This is a limited set of all possible settings of the QA over tabular data task. We focus on these types of questions as they represent a wide variety of the type of concrete questions that can be asked from a table, and also make for a solid evaluation setting with existing benchmarks such as Open Wikitables and Databench, which we use as the basis for our evaluation. Nonetheless, future research could explore more open QA settings, as well as investigating robust evaluation protocols.

Ethical Statement

In this work, we use two existing and publicly available benchmarks and reference them accordingly. From our work, there is no particular risk, but models deployed to answer questions from tables need to be further investigated – in this work, we focused on the effectiveness of synthetic data but did not fully investigate the type of errors made by the system. This can be problematic depending on the domain, and further testing would be required, for which our automatically-generated tests can serve as the basis for unit testing on unseen datasets. During the work we focus on open weights model due to privacy concerns, as usually datasets are

[4] https://github.com/jorses/syngen-tables

proprietary owned, and external APIs may not be suitable, in addition to remain unreliable and the inability of performing further testing.

Acknowledgments

This work is funded by the Ministerio para la Transformación Digital y de la Función Pública and Plan de Recuperación, Transformación y Resiliencia - Funded by EU – NextGenerationEU within the framework of the project Desarrollo Modelos ALIA. This work has also been partially supported by Project CONSENSO (PID2021-122263OB-C21) and Project SocialTox (PDC2022-133146-C21) funded by MCIN/AEI/10.13039/501100011033 and by the European Union NextGenerationEU/PRTR, Project ROMANET (CERV-2024-CHAR-LITI-101215052), funded by the European Union under the Citizens, Equality, Rights and Values programme, Project HEART-NLP-UJA (PID2024-156263OB-C21) and project VERITAS-H (AIA2025-163322-C64) funded by MICIU/AEI/10.13039/501100011033 and by ERDF/EU, Project GALENO-IA (DGP_PIDI_2024_00852) funded by Junta de Andalucía. Jose Camacho-Collados was supported by a UKRI Future Leaders Fellowship.

8. Bibliographical References

Apache parquet. https://arrow.apache.org/. Accessed: March 9, 2026.

Kaggle dataset: Venta de coches. https://www.kaggle.com/datasets/datamarket/venta-de-coches. Accessed on 2023-10-20.

2016. County-to-county migration flows 2012-2016. https://www.census.gov/topics/population/migration/guidance/county-to-county-migration-flows.html. Accessed on 2023-10-20.

2018. Police department incident reports 2018 to present. data.gov, catalog.data.gov/dataset/police-department-incident-reports-2018-to-present. Data.gov dataset. URL: catalog.data.gov/dataset/police-department-incident-reports-2018-to-present.

2021. Kaggle dataset: Emoji diet nutritional data sr28. https://www.kaggle.com/datasets/ofrancisco/emoji-diet-nutritional-data-sr28. Accessed on 2023-10-20.

2022. New york city weather 1869-2022 dataset. Kaggle, www.kaggle.com/datasets/danbraswell/new-york-city-weather-18692022. Kaggle dataset. URL: www.kaggle.com/datasets/danbraswell/new-york-city-weather-18692022.

2023. Airbnb listings in new york city dataset. Kaggle, www.kaggle.com/datasets/labdmitriy/airbnb. Kaggle dataset. URL: www.kaggle.com/datasets/labdmitriy/airbnb.

2023. Data scraped from idealista. Graphext.

2023. Fifa 21 complete player dataset. Kaggle, www.kaggle.com/datasets/stefanoleone992/fifa-21-complete-player-dataset. Kaggle dataset. URL: www.kaggle.com/datasets/stefanoleone992/fifa-21-complete-player-dataset.

2023. Us tornado dataset 1950-2021. Kaggle, www.kaggle.com/datasets/danbraswell/us-tornado-dataset-1950-2021. Kaggle dataset. URL: www.kaggle.com/datasets/danbraswell/us-tornado-dataset-1950-2021.

Harold Abelson, Gerald Jay Sussman, and Julie Sussman. 1985. *Structure and Interpretation of Computer Programs*. MIT Press, Cambridge, Massachusetts.

Susant Achary. 2021. Holiday package purchase prediction dataset. Kaggle Dataset.

US Small Business Administration. 2021. Should this loan be approved or denied? Accessed on 2023-10-20.

AEMET. 2020. Últimos datos de observación de temperatura en madrid. Accessed on: 2020.

Abien Fred Agarap. 2018. Statistical analysis on e-commerce reviews, with sentiment classification using bidirectional recurrent neural network (rnn). *arXiv preprint arXiv:1805.03687*.

Alfred V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling*, volume 1. Prentice-Hall, Englewood Cliffs, NJ.

Inside Airbnb. Dec, 2022. Airbnb listings in madrid, spain (december, 2022). Dataset, Inside Airbnb. URL: http://insideairbnb.com/get-the-data/.

Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic QA corpora generation with roundtrip consistency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6168–6173, Florence, Italy. Association for Computational Linguistics.

Alexandra. 2018. Generic food database. data.world, data.world/alexandra/generic-food-database. Data.world dataset. URL: data.world/alexandra/generic-food-database.

Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. The fact extraction and VERification over unstructured and structured information (FEVEROUS) shared task. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*, pages 1–13, Dominican Republic. Association for Computational Linguistics.

American Psychological Association. 1983. *Publications Manual*. American Psychological Association, Washington, DC.

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

Galen Andrew and Jianfeng Gao. 2007. Scalable training of L1-regularized log-linear models. In *Proceedings of the 24th International Conference on Machine Learning*, pages 33–40.

Shir Ashury-Tahan, Yifan Mai, Rajmohan C, Ariel Gera, Yotam Perlitz, Asaf Yehudai, Elron Bandel, Leshem Choshen, Eyal Shnarch, Percy Liang, and Michal Shmueli-Scheuer. 2025. The mighty torr: A benchmark for table reasoning and robustness.

Rounak Banik. 2023. Pokemon feature correlation dataset. Kaggle.

Robert Baumgartner, Georg Gottlob, and Sergio Flesca. 2001. Visual information extraction with Lixto. In *Proceedings of the 27th International Conference on Very Large Databases*, pages 119–128, Rome, Italy. Morgan Kaufmann.

Ronald J. Brachman and James G. Schmolze. 1985. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171–216.

Brandwatch. 2021. Us election raw polling data. https://www.brandwatch.com/p/us-election-raw-polling-data/. Accessed on 2023-10-20.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

BSI. 1973a. *Natural Fibre Twines*, 3rd edition. British Standards Institution, London. BS 2570.

BSI. 1973b. Natural fibre twines. BS 2570, British Standards Institution, London. 3rd. edn.

Thomas Buhrmann. 2023. Lector.

A. Castor and L. E. Pollux. 1992. The use of user modelling to guide inference and learning. *Applied Intelligence*, 2(1):37–53.

Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. 1981. Alternation. *Journal of the Association for Computing Machinery*, 28(1):114–133.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Kaijie Zhu, Hao Chen, Linyi Yang, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. 2023. A survey on evaluation of large language models. *arXiv preprint arXiv:2307.03109*.

Harrison Chase. 2022. Langchain. https://github.com/langchain-ai/langchain.

Saneem Chemmengath, Vishwajeet Kumar, Samarth Bharadwaj, Jaydeep Sen, Mustafa Canim, Soumen Chakrabarti, Alfio Gliozzo, and Karthik Sankaranarayanan. 2021. Topic transferable table question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4159–4172, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023. How is chatgpt's behavior changing over time?

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski

Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021a. Evaluating large language models trained on code.

Wenhu Chen. 2023. Large language models are few(1)-shot table reasoners. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1120–1130, Dubrovnik, Croatia. Association for Computational Linguistics.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Yang Wang, and William W Cohen. 2020. Open question answering over tables and text. In *International Conference on Learning Representations*.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. 2021b. FinQA: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3697–3711, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

J.L. Chercheur. 1994. *Case-Based Reasoning*, 2nd edition. Morgan Kaufman Publishers, San Mateo, CA.

Arush Chillar. 2023. Disneyland customer reviews dataset. Kaggle.

N. Chomsky. 1973. Conditions on transformations. In *A festschrift for Morris Halle*, New York. Holt, Rinehart & Winston.

CIS. 2015. 2015 spain political polls cis. https://public.graphext.com/90ca7539b160fdfa/index.html?section=data. Accessed on 2023-10-20.

James W. Cooley and John W. Tukey. 1965. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90):297–301.

Cerdeira A. Almeida F. Matos T. Cortez, Paulo and J. Reis. 2009. Wine Quality. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C56S3T.

Jirka Daberger. 2023. Clustering zoo animals. Kaggle.

Xiang Deng, Vasilisa Bashlovkina, Feng Han, Simon Baumgartner, and Michael Bendersky. 2023. Llms to the moon? reddit market sentiment analysis with large language models. In *Companion Proceedings of the ACM Web Conference 2023*, WWW '23 Companion, page 1014–1019, New York, NY, USA. Association for Computing Machinery.

Xinya Du and Claire Cardie. 2018. Harvesting paragraph-level question-answer pairs from Wikipedia. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1907–1917, Melbourne, Australia. Association for Computational Linguistics.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada. Association for Computational Linguistics.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874, Copenhagen, Denmark. Association for Computational Linguistics.

Umberto Eco. 1990. *The Limits of Interpretation*. Indian University Press.

Julian Eisenschlos, Maharshi Gor, Thomas Müller, and William Cohen. 2021. MATE: Multi-view attention for table transformer efficiency. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7606–7619, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xi Fang, Weijie Xu, Fiona Anting Tan, Ziqing Hu, Jiani Zhang, Yanjun Qi, Srinivasan H. Sengamedu, and Christos Faloutsos. 2024. Large language models (LLMs) on tabular data: Prediction, generation, and understanding - a survey. *Transactions on Machine Learning Research*.

Forbes. 2022. Forbes billionaires. https://www.forbes.com/billionaires/. Accessed on 2023-10-20.

Xavier Vivancos García. 2020. Bakery market basket analysis dataset. Kaggle.

G. Gerganov. 2023. llama.cpp: Low-latency audio streaming library for c++. https://github.com/ggerganov/llama.cpp. Accessed: Sep 20, 2023.

Siavash Golkar, Mariel Pettee, Michael Eickenberg, Alberto Bietti, Miles Cranmer, Geraud Krawezik, Francois Lanusse, Michael McCabe, Ruben Ohana, Liam Parker, Bruno Régaldo-Saint Blancard, Tiberiu Tesileanu, Kyunghyun Cho, and Shirley Ho. 2023. xval: A continuous number encoding for large language models.

Google. 2021. Bigquery dataset: Patents. https://www.kaggle.com/datasets/bigquery/patents/data. Accessed on 2023-10-20.

Georg Gottlob. 1992. Complexity results for nonmonotonic logics. *Journal of Logic and Computation*, 2(3):397–425.

Georg Gottlob, Nicola Leone, and Francesco Scarcello. 2002. Hypertree decompositions and tractable queries. *Journal of Computer and System Sciences*, 64(3):579–627.

Graphext. Trustpilot: Wise vs n26 reviews - scraped by graphext. https://public.graphext.com/367e29432331fbfd/index.html?section=data. Accessed on 2023-10-20.

Graphext. 2019. 2019 ing twitter mentions - scraped by graphext. https://public.graphext.com/075030310aa702c6/index.html. Accessed on 2023-10-20.

Graphext. 2020a. Trump tweets - scraped by graphext. https://public.graphext.com/be903c098a90e46f/index.html?section=data. Accessed on 2023-10-20.

Graphext. 2020b. X influencer analysis - scraped by graphext. https://public.graphext.com/e097f1ea03d761a9/index.html. Accessed on 2023-10-20.

Graphext. 2021a. Airline mentions on x - scraped by graphext. https://public.graphext.com/7e6999327d1f83fd/index.html. Accessed on 2023-10-20.

Graphext. 2021b. Data-driven seo: A keyword optimization guide using web scraping, co-occurrence analysis (graphext, deepnote, adwords). https://www.graphext.com/post/data-driven-seo-a-keyword-optimization-guide-using-web-scraping-co-occurrence-analysis-graphext-deepnote-adwords. Accessed on 2023-10-20.

Graphext. 2022a. Boris johnson tweets as pm - scraped by graphext. https://public.graphext.com/f6623a1ca0f41c8e/index.html. Accessed on 2023-10-20.

Graphext. 2022b. Joe biden tweets (scraped from x). https://public.graphext.com/339cee259f0a9b32/index.html?section=data. Accessed on 2023-10-20.

Graphext. 2023a. Love survey. Accessed on 2023-10-20.

Graphext. 2023b. Professional map. Graphext.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y. Wu, Y. K. Li, Fuli Luo, Yingfei Xiong, and Wenfeng Liang. 2024a. Deepseek-coder: When the large language model meets programming – the rise of code intelligence.

Shash Guo, Lizi Liao, Cuiping Li, and Tat-Seng Chua. 2024b. A survey on neural question generation: methods, applications, and prospects. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, IJCAI '24.

Vivek Gupta, Pranshu Kandoi, Mahek Bhavesh Vora, Shuo Zhang, Yujie He, Ridho Reinanda, and Vivek Srikumar. 2023. Temptabqa: Temporal question answering for semi-structured tables.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Dan Gusfield. 1997. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, Cambridge, UK.

Heesoo37. 2023. 120 years of olympic history: Athletes and results dataset. Kaggle.

Michael Heilman and Noah A. Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617, Los Angeles, California. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask

language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Jonathan Herzig, Thomas Müller, Syrine Krichene, and Julian Eisenschlos. 2021. Open domain question answering over tables via dense retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 512–519, Online. Association for Computational Linguistics.

Jonathan Herzig, Paweł Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. 2020. Tapas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Seattle, Washington, United States.

Paul Gerhard Hoel. 1971a. *Elementary Statistics*, 3rd edition. Wiley series in probability and mathematical statistics. Wiley, New York, Chichester. ISBN 0 471 40300.

Paul Gerhard Hoel. 1971b. *Elementary Statistics*, 3rd edition, Wiley series in probability and mathematical statistics, pages 19–33. Wiley, New York, Chichester. ISBN 0 471 40300.

IJCAI Proceedings. IJCAI camera ready submission. https://proceedings.ijcai.org/info.

Impapan. 2023. Predict student performance dataset. Kaggle.

INE. 2018. Encuesta cuatrienal de estructura salarial. INE Website.

Todor Ivanov and Valeri Penchev. 2024. Ai benchmarks and datasets for llm evaluation. *arXiv preprint arXiv:2412.01020*.

Ashish Jangra. 2021. Ted talks - scraped from ted website. Kaggle.

Ram Jas. 2022. Telco customer churn dataset. Kaggle, www.kaggle.com/datasets/blastchar/telco-customer-churn. Kaggle dataset. URL: www.kaggle.com/datasets/blastchar/telco-customer-churn.

Otto Jespersen. 1922. *Language: Its Nature, Development, and Origin*. Allen and Unwin.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023a. Mistral 7b.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023b. Mistral 7b.

Wenxiang Jiao, Wenxuan Wang, Jen tse Huang, Xing Wang, and Zhaopeng Tu. 2023. Is chatgpt a good translator? a preliminary study. In *ArXiv*.

Nengzheng Jin, Joanna Siebert, Dongfang Li, and Qingcai Chen. 2022. A survey on table question answering: Recent advances. In *Knowledge Graph and Semantic Computing: Knowledge Graph Empowers the Digital Economy*, pages 174–186, Singapore. Springer Nature Singapore.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. 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.

Kaggle. 2021. Titanic dataset. Kaggle, www.kaggle.com/c/titanic. Dataset available on Kaggle. URL: www.kaggle.com/c/titanic.

Kaggle. 2023a. Bank customer churn. Kaggle.

Kaggle. 2023b. Employee satisfaction index dataset. Kaggle Dataset.

Kaggle. 2023c. Imdb movies. Kaggle.

Kaggle. 2023d. Professionals kaggle survey. Kaggle.

Kaggle. 2023e. Supermarket sales. Kaggle.

Minki Kang, Seanie Lee, Jinheon Baek, Kenji Kawaguchi, and Sung Ju Hwang. 2023. Knowledge-augmented reasoning distillation for small language models in knowledge-intensive tasks. In *Advances in Neural Information Processing Systems*, volume 36, pages 48573–48602. Curran Associates, Inc.

Margaret L. Kern, Paul X. McCarthy, Deepanjan Chakrabarty, and Marian-Andrei Rizoiu. 2019. Social media-predicted personality traits and values can help match people to their ideal jobs. *Proceedings of the National Academy of Sciences*, 116(52):26459–64.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Oleksandr Kolomiyets and Marie-Francine Moens. 2011. A survey on question answering technology from an information retrieval perspective. *Information Sciences*, 181(24):5412–5434.

Sunjun Kweon, Yeonsu Kwon, Seonhee Cho, Yohan Jo, and Edward Choi. 2023. Open-WikiTable : Dataset for open domain question answering with complex reasoning over table. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8285–8297, Toronto, Canada. Association for Computational Linguistics.

Megagon Labs. 2017. Happy moments dataset. Kaggle.

Igor Labutov, Sumit Basu, and Lucy Vanderwende. 2015. Deep questions without deep understanding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 889–898, Beijing, China. Association for Computational Linguistics.

Dong Bok Lee, Seanie Lee, Woo Tae Jeong, Donghwan Kim, and Sung Ju Hwang. 2020. Generating diverse and consistent QA pairs from contexts with information-maximizing hierarchical conditional VAEs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 208–224, Online. Association for Computational Linguistics.

Hector J. Levesque. 1984a. Foundations of a functional approach to knowledge representation. *Artificial Intelligence*, 23(2):155–212.

Hector J. Levesque. 1984b. A logic of implicit and explicit belief. In *Proceedings of the Fourth National Conference on Artificial Intelligence*, pages 198–202, Austin, Texas. American Association for Artificial Intelligence.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, et al. 2024. 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.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods.

David Lindberg, Fred Popowich, John Nesbit, and Phil Winne. 2013. Generating natural language questions to support learning on-line. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 105–114, Sofia, Bulgaria. Association for Computational Linguistics.

Yuan Ling, Yuan An, and Sadid Hasan. 2017. Improving clinical diagnosis inference through integration of structured and unstructured knowledge. In *Proceedings of the 1st Workshop on Sense, Concept and Entity Representations and their Applications*, pages 31–36, Valencia, Spain. Association for Computational Linguistics.

Qian Liu, Bei Chen, Jiaqi Guo, Morteza Ziyadi, Zeqi Lin, Weizhu Chen, and Jian-Guang Lou. 2022. TAPEX: Table pre-training via learning a neural SQL executor. In *International Conference on Learning Representations*.

Weizheng Lu, Jing Zhang, Ju Fan, Zihao Fu, Yueguo Chen, and Xiaoyong Du. 2025. Large language model for table processing: a survey. *Frontiers of Computer Science*, 19(2):192350.

Kaixin Ma, Hao Cheng, Xiaodong Liu, Eric Nyberg, and Jianfeng Gao. 2022. Open-domain question answering via chain of reasoning over heterogeneous knowledge. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5360–5374, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Dehai Min, Nan Hu, Rihui Jin, Nuo Lin, Jiaoyan Chen, Yongrui Chen, Yu Li, Guilin Qi, Yun Li, Nijun Li, and Qianren Wang. 2024. Exploring the impact of table-to-text methods on augmenting LLM-based question answering with domain hybrid data. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, pages 464–482, Mexico City, Mexico. Association for Computational Linguistics.

UCI ML. 2015. Online Retail. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5BW33.

UCI ML. 2016. German credit. https://www.kaggle.com/datasets/uciml/german-credit/data. Accessed on 2023-10-20.

UCI ML. 2021. Heart failure prediction dataset. Kaggle, www.kaggle.com/datasets/fedesoriano/heart-failure-prediction. Originally published by UCI ML at https://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease.

Raphaël Mouravieff, Benjamin Piwowarski, and Sylvain Lamprier. 2024. Learning relational decomposition of queries for question answering from tables. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10471–10485, Bangkok, Thailand. Association for Computational Linguistics.

Rob Mulla. 2021. Roller coaster scraped from wikipedia. Kaggle, www.kaggle.com/datasets/robikscube/rollercoaster-database. Kaggle dataset. URL: www.kaggle.com/datasets/robikscube/rollercoaster-database.

Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryściński, Hailey Schoelkopf, Riley Kong, Xiangru Tang, Mutethia Mutuma, Ben Rosand, Isabel Trindade, Renusree Bandaru, Jacob Cunningham, Caiming Xiong, Dragomir Radev, and Dragomir Radev. 2022a. FeTaQA: Free-form table question answering. *Transactions of the Association for Computational Linguistics*, 10:35–49.

Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryściński, Hailey Schoelkopf, Riley Kong, Xiangru Tang, Mutethia Mutuma, Ben Rosand, Isabel Trindade, Renusree Bandaru, Jacob Cunningham, Caiming Xiong, Dragomir Radev, and Dragomir Radev. 2022b. FeTaQA: Free-form table question answering. *Transactions of the Association for Computational Linguistics*, 10:35–49.

Bernhard Nebel. 2000. On the compilability and expressive power of propositional planning formalisms. *Journal of Artificial Intelligence Research*, 12:271–315.

Hacker News. 2017. Hacker news dataset. Kaggle, www.kaggle.com/datasets/hacker-news/hacker-news. Kaggle dataset. URL: www.kaggle.com/datasets/hacker-news/hacker-news.

University of Brown. 2017. Billboard lyrics. https://www.kaggle.com/code/djohnbar/text-mining-of-billboard-lyrics-1964-2015.

University of Columbia. 2009. Speed dating.

University of Vanderbilt. 2019. Predict diabetes. Kaggle.

OpenAI. 2023. Gpt-4 technical report.

Jorge Osés-Grijalba, Luis Alfonso Ureña López, Jose Camacho-Collados, and Eugenio Martínez Cámara. 2024a. Towards quality benchmarking in question answering over tabular data in Spanish. *Procesamiento del Lenguaje Natural*, 73:283–296.

Jorge Osés-Grijalba, Luis Alfonso Ureña López, Eugenio Martínez Cámara, and Jose Camacho-Collados. 2024b. Question answering over tabular data with databench: A large-scale empirical evaluation of llms. In *Proceedings of LREC-COLING 2024*, Turin, Italy.

Vaishali Pal, Andrew Yates, Evangelos Kanoulas, and Maarten de Rijke. 2023a. MultiTabQA: Generating tabular answers for multi-table question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6322–6334, Toronto, Canada. Association for Computational Linguistics.

Vaishali Pal, Andrew Yates, Evangelos Kanoulas, and Maarten de Rijke. 2023b. MultiTabQA: Generating tabular answers for multi-table question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6322–6334, Toronto, Canada. Association for Computational Linguistics.

Panupong Pasupat and Percy Liang. 2015a. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480, Beijing, China. Association for Computational Linguistics.

Panupong Pasupat and Percy Liang. 2015b. Compositional semantic parsing on semi-structured tables. *CoRR*, abs/1508.00305.

Panupong Pasupat and Percy Liang. 2015c. Compositional semantic parsing on semi-structured tables.

PavanKalyan. 2021. Predicting employee attrition. https://www.kaggle.com/datasets/pavan9065/

predicting-employee-attrition. Accessed on 2023-10-20.

Kamil Pytlak. 2023. Stroke likelihood dataset. Kaggle.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Ramjas Maurya. 2022. Aviation accidents history (1919 – april 2022).

Mohammad Sadegh Rasooli and Joel R. Tetreault. 2015. Yara parser: A fast and accurate dependency parser. *Computing Research Repository*, arXiv:1503.06733. Version 2.

Meg Risdal. 2017. New york city taxi trip duration.

Álvaro Rodrigo, Anselmo Peñas, Yusuke Miyao, and Noriko Kando. 2018. Do systems pass university entrance exams? *Inf. Process. Manag.*, 54(4):564–575.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2023. Code llama: Open foundation models for code.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2024. Code llama: Open foundation models for code.

Ruchi798. 2023. Data science job salaries dataset. Kaggle.

Rodolfo Saldanha. 2020. Marketing campaign. https://www.kaggle.com/datasets/rodsaldanha/arketing-campaign. Accessed on 2023-10-20.

Maximilian Schmidt, Andrea Bartezzaghi, and Ngoc Thang Vu. 2024. Prompting-based synthetic data generation for few-shot question answering. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 13168–13178, Torino, Italia. ELRA and ICCL.

Philippa Shoemark, Farhana Ferdousi Liza, Dong Nguyen, Scott Hale, and Barbara McGillivray. 2019. Room to Glo: A systematic comparison of semantic change detection approaches with word embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 66–76, Hong Kong, China. Association for Computational Linguistics.

Charles Joseph Singer, E. J. Holmyard, and A. R. Hall, editors. 1954–58. *A history of technology*. Oxford University Press, London. 5 vol.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S. Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, Perry Payne, Martin Seneviratne, Paul Gamble, Chris Kelly, Nathaneal Scharli, Aakanksha Chowdhery, Philip Mansfield, Blaise Aguera y Arcas, Dale Webster, Greg S. Corrado, Yossi Matias, Katherine Chou, Juraj Gottweis, Nenad Tomasev, Yun Liu, Alvin Rajkomar, Joelle Barral, Christopher Semturs, Alan Karthikesalingam, and Vivek Natarajan. 2022. Large language models encode clinical knowledge.

Charlotte Siska, Katerina Marazopoulou, Melissa Ailem, and James Bono. 2024. Examining the robustness of LLM evaluation to the distributional assumptions of benchmarks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10406–10421, Bangkok, Thailand. Association for Computational Linguistics.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018. Leveraging context information for natural question generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 569–574, New Orleans, Louisiana. Association for Computational Linguistics.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya

Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Jannik Strötgen and Michael Gertz. 2012. Temporal tagging on different domains: Challenges, strategies, and gold standards. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, pages 3746–3753, Istanbul, Turkey. European Language Resource Association (ELRA).

Aofeng Su, Aowen Wang, Chao Ye, Chen Zhou, Ga Zhang, Gang Chen, Guangcheng Zhu, Haobo Wang, Haokai Xu, Hao Chen, Haoze Li, Haoxuan Lan, Jiaming Tian, Jing Yuan, Junbo Zhao, Junlin Zhou, Kaizhe Shou, Liangyu Zha, Lin Long, Liyao Li, Pengzuo Wu, Qi Zhang, Qingyi Huang, Saisai Yang, Tao Zhang, Wentao Ye, Wufang Zhu, Xiaomeng Hu, Xijun Gu, Xinjie Sun, Xiang Li, Yuhang Yang, and Zhiqing Xiao. 2024. Tablegpt2: A large multimodal model with tabular data integration.

S. Superman, B. Batman, C. Catwoman, and S. Spiderman. 2000. *Superheroes experiences with books*, 20th edition. The Phantom Editors Associates, Gotham City.

New York Times. 2021. Redivis dataset: Nyt world 2021. https://redivis.com/datasets/prrj-e3mazx6p3. Accessed on 2023-10-20. Sampled for 2021.

Tomigelo. 2023. Spotify audio features dataset. Kaggle.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. 2023. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*.

Omar Olivares Urrutia. 2018. Emoji diet. Kaggle.

Ellen M Voorhees. 2001. The trec question answering track. *Natural Language Engineering*, 7(4):361–378.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Superglue: a stickier benchmark for general-purpose language understanding systems. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 3266–3280.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Guan Wang, Sijie Cheng, Qiying Yu, and Changling Liu. 2023. OpenLLMs: Less is More for Open-source Models.

Zilong Wang, Hao Zhang, Chun-Liang Li, Julian Martin Eisenschlos, Vincent Perot, Zifeng Wang, Lesly Miculicich, Yasuhisa Fujii, Jingbo Shang, Chen-Yu Lee, et al. 2024. Chain-of-table: Evolving tables in the reasoning chain for table understanding. In *The Twelfth International Conference on Learning Representations*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models. *Transactions on Machine Learning Research*. Survey Certification.

WH. 2020. World happiness report 2020 data. https://worldhappiness.report/data. Accessed on 2023-10-20.

Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. 2023. Bloomberggpt: A large language model for finance.

Xianjie Wu, Jian Yang, Linzheng Chai, Ge Zhang, Jiaheng Liu, Xeron Du, Di Liang, Daixin Shu, Xianfu Cheng, Tianzhen Sun, Tongliang Li, Zhoujun Li, and Guanglin Niu. 2025. Tablebench: A comprehensive and complex benchmark for table question answering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(24):25497–25506.

Bohao Yang, Yingji Zhang, Dong Liu, André Freitas, and Chenghua Lin. 2025. Does table source matter? benchmarking and improving multimodal scientific table understanding and reasoning.

Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Shaochen Zhong, Bing Yin, and Xia Hu. 2024a. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *ACM Trans. Knowl. Discov. Data*, 18(6).

Yazheng Yang, Yuqi Wang, Guang Liu, Ledell Wu, and Qi Liu. 2024b. Unitabe: A universal pretraining protocol for tabular foundation model in data science. In *The Twelfth International Conference on Learning Representations*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

City Of New York. 2022. Nyc 311 data. https://data.cityofnewyork.us/Social-Services/NYC-311-Data/jrb2-thup. Accessed on 2023-10-20.

Qin Zhang, Shangsi Chen, Dongkuan Xu, Qingqing Cao, Xiaojun Chen, Trevor Cohn, and Meng Fang. 2023a. A survey for efficient open domain question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14447–14465, Toronto, Canada. Association for Computational Linguistics.

Qin Zhang, Shangsi Chen, Dongkuan Xu, Qingqing Cao, Xiaojun Chen, Trevor Cohn, and Meng Fang. 2023b. A survey for efficient open domain question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14447–14465, Toronto, Canada. Association for Computational Linguistics.

Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B. Hashimoto. 2024a. Benchmarking large language models for news summarization. *Transactions of the Association for Computational Linguistics*, 12:39–57.

Wenxuan Zhang, Yue Deng, Bing Liu, Sinno Pan, and Lidong Bing. 2024b. Sentiment analysis in the era of large language models: A reality check. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3881–3906, Mexico City, Mexico. Association for Computational Linguistics.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. 2024. Expel: Llm agents are experiential learners. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19632–19642.

Wenting Zhao, Ye Liu, Yao Wan, Yibo Wang, Zhongfen Deng, and Philip S. Yu. 2023. Localize, retrieve and fuse: A generalized framework for free-form question answering over tables. In *Findings of the Association for Computational Linguistics: IJCNLP-AACL 2023 (Findings)*, pages 1–12, Nusa Dua, Bali. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017a. Seq2sql: Generating structured queries from natural language using reinforcement learning.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017b. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and M. Zhou. 2017. Neural question generation from text: A preliminary study. In *NLPCC*.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3277–3287, Online. Association for Computational Linguistics.

Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Neil Zhenqiang Gong, Yue Zhang, et al. 2023. Promptbench: Towards evaluating the robustness of large language models on adversarial prompts. *arXiv preprint arXiv:2306.04528*.

Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. 2023. Toolqa: A dataset for llm question answering with external tools.

A. Example Prompts for Tabular QA

We provide some example prompts provided to LLMs in our experiments. Table information showcased within the prompt changes depending on the table needed to answer the question.

A.1. SQL

Listing 1: SQL Prompt Example for Open Wikitables. Inspired by the one provided by LangChain(**?**)

```
You are a SQLite expert. Given an input question,
 first create a syntactically correct SQLite
query to run, then look at the results of the
query and return the answer to the input
question.

Unless the user specifies in the question a
specific number of examples to obtain, query for
 at most 1 result using the LIMIT clause as per
SQLite. You can order the results to return the
most informative data in the database.

Never query for all columns from a table. You
must query only the columns that are needed to
answer the question. Wrap each column name in
double quotes (\") to denote them as delimited
identifiers. Pay attention to use only the
column names you can see in the tables below. Be
 careful to not query for columns that do not
exist. Also, pay attention to which column is in
 which table.

Use the following format:

Question: <Your question here>
SQLQuery: <SQL Query to run>

Only use the following table:

CREATE TABLE table_1_1013129_8 (
    "Pick" TEXT,
    "Player" TEXT,
    "Position_" TEXT,
)

/*
Example rows from table_1_1013129_8:
Pick       Player           Position_
183     Jason Boudrias      Forward
184     Brad Englehart      Centre
185     Rob Guinn           Defence
*/

Question:
```

A.2. Python

Listing 2: Python Prompt Template for DataBench

```
You are a Pandas expert. You are given a
function called answer, which
must answer a given question.

Below are a number of examples of questions and
their corresponding Pandas queries.

# Dataframe representation:
# Left  Satisfaction Level  Avg Monthly Hours
#  0         0.94                173
#  1         0.21                273
#  0         0.63                293

col_names= ['Left', 'Satisfaction Level', 'Avg
Monthly Hours']

def answer(df: pd.DataFrame):
    """Question: question to answer will be here
"""
    df.columns = col_names
    return # answer code here
```

B. Additional Results

B.1. Synthetic Dataset Descriptions

In Table 8 we see a detailed composition of the synthetic datasets generated that end up being used in our main few-shot experiment. The procedure followed is that described in section 3.

B.2. Detailed Averages

In Table 9 and Table 10 we display the same data as in Figure 3 but in tabular format.

B.3. Generic Question Experiments

In Table 11 and Table 12 we can see accuracies for the repetition of the main experiment but using generic questions. The result of subtracting the values from the generic questions in Table 12 to the averages displayed in Table 10 yields Table 6.

C. Template Examples

In Table 15, Table 16, and Table 17, we present representative samples of the templates generated for the 4-column toy dataset described in Table 14.

Q. Type	Wiki A	Wiki B	Wiki C	Bench A	Bench B	Bench C
category	50813	50867	19026	220	221	220
number	17047	16993	48834	800	800	803
list[category]	-	-	-	773	749	771
list[number]	-	-	-	157	180	156
Total	67860	67860	67860	1950	1950	1950

Table 8: Type composition and number of QA pairs generated per data type in Open Wikitables (Wiki) and Databench (Bench).

	baseline (0)	5	10	15	20	25	30
codellama	33.07	37.00	39.40	40.60	41.10	41.20	41.90
deepseek	40.83	46.90	48.30	48.50	47.30	46.60	46.70
mistral	27.93	25.30	26.30	27.20	27.90	29.20	29.80
openchat	12.37	22.00	29.10	32.10	30.30	27.30	26.10
zephyr	6.77	8.80	10.50	11.20	8.80	7.60	6.20

Table 9: Results of *Open Wikitables* in Figure 3 in tabular format.

	baseline (0)	5	10	15	20	25	30
codellama	29.07	51.30	52.40	51.90	50.80	51.10	51.10
deepseek	42.13	63.60	65.40	65.40	64.30	64.60	64.90
mistral	30.77	49.90	51.00	52.00	51.30	51.80	52.20
openchat	29.37	44.90	46.70	47.60	46.10	45.90	44.80
zephyr	30.20	45.10	44.10	44.90	43.90	44.70	43.30

Table 10: Results of DataBench in Figure 3 in tabular format.

	baseline (0)	5	10	15	20	25	30
codellama	33.07	35.78	35.14	34.69	35.00	35.96	31.98
deepseek	40.83	44.73	44.08	44.03	43.77	42.77	42.60
mistral	27.93	29.78	29.00	28.55	29.60	28.85	29.20
openchat	12.37	19.04	20.21	23.10	23.20	21.37	13.78
zephyr	6.77	7.12	9.30	11.09	10.50	7.44	7.02

Table 11: Accuracies on *Open WikiTables* with generic questions at varying few-shot counts.

	baseline (0)	5	10	15	20	25	30
codellama	29.07	41.58	43.50	38.74	42.60	37.14	44.60
deepseek	42.13	53.98	58.00	47.93	59.20	43.95	57.70
mistral	30.77	46.17	45.60	46.86	45.32	46.32	46.40
openchat	29.37	45.64	44.70	48.39	45.80	46.40	41.40
zephyr	30.20	42.34	42.00	41.96	41.50	37.98	38.00

Table 12: Accuracies on *DataBench* with only generic questions at varying few-shot counts.

The number of QA pairs can be computed directly from the dataset in Table 14, without requiring any assumptions. The dataset contains $N = 4$ columns: `Team`, `Location`, `Points`, and `Age`, each with 2 distinct non-null values. For every ordered pair of distinct columns (A, B)—where A is the target and B is the conditioning column—we generate one QA pair for each unique value in B. Since there are 12 such ordered pairs and each conditioning column has 2 unique values, the base number of QA pairs is:

$$12 \times 2 = 24\ N_{pairs}.$$

By applying 3 selectors and 3 filters to each base pair—yielding 9 combinations—the total number of QA pairs becomes:

$$9 \times 24 = 216.$$

Type	Numerical Columns	Categorical Columns
Selectors	average, maximum, minimum	unique, first by row order, last by row order
Filters	bigger than, smaller than, equal to	*is*, *is not*

Table 13: Selectors and filters used for numerical and categorical columns.

Team	Location	Points	Age
Red Hawks	East Coast	34	21
Blue Whales	West Coast	28	25

Table 14: Extended toy dataset with two categorical and two numerical columns.

This example is meant to illustrate that even an extremely small table like Table 14 can produce as many as 216 synthetic QA pairs. The various selectors and filters applicable to each type of column in the dataset are listed in Table 13. Despite the limited data variety, even this small toy dataset enables the generation of a substantial number of synthetic questions.

Index	Template	Value	Columns	Filled Template	Answer
1	What is the average of A where B is equal to c?	Red Hawks	points, team	What is the average of points where team is equal to Red Hawks?	34
2	What is the average of A where B is equal to c?	Blue Whales	points, team	What is the average of points where team is equal to Blue Whales?	28
3	What is the average of A where B is not equal to c?	Red Hawks	points, team	What is the average of points where team is not equal to Red Hawks?	28
4	What is the average of A where B is not equal to c?	Blue Whales	points, team	What is the average of points where team is not equal to Blue Whales?	34
5	What is the max of A where B is equal to c?	Red Hawks	points, team	What is the max of points where team is equal to Red Hawks?	34
6	What is the max of A where B is equal to c?	Blue Whales	points, team	What is the max of points where team is equal to Blue Whales?	28
7	What is the max of A where B is not equal to c?	Red Hawks	points, team	What is the max of points where team is not equal to Red Hawks?	28
8	What is the max of A where B is not equal to c?	Blue Whales	points, team	What is the max of points where team is not equal to Blue Whales?	34
9	What is the min of A where B is equal to c?	Red Hawks	points, team	What is the min of points where team is equal to Red Hawks?	34
10	What is the min of A where B is equal to c?	Blue Whales	points, team	What is the min of points where team is equal to Blue Whales?	28
11	What is the min of A where B is not equal to c?	Red Hawks	points, team	What is the min of points where team is not equal to Red Hawks?	28
12	What is the min of A where B is not equal to c?	Blue Whales	points, team	What is the min of points where team is not equal to Blue Whales?	34

Table 15: Templates 1–12 generated for Table 14 (part 1/3).

Index	Template	Value	Columns	Filled Template	Answer
13	What is the first value of A where B is equal to c?	34	location, points	What is the first value of location where points is equal to 34?	East Coast
14	What is the first value of A where B is equal to c?	28	location, points	What is the first value of location where points is equal to 28?	West Coast
15	What is the first value of A where B is less than c?	34	location, points	What is the first value of location where points is less than 34?	West Coast
16	What is the first value of A where B is less than c?	28	location, points	What is the first value of location where points is less than 28?	
17	What is the first value of A where B is more than c?	28	location, points	What is the first value of location where points is more than 28?	East Coast
18	What is the first value of A where B is more than c?	34	location, points	What is the first value of location where points is more than 34?	
19	What is the last value of A where B is equal to c?	34	location, points	What is the last value of location where points is equal to 34?	East Coast
20	What is the last value of A where B is equal to c?	28	location, points	What is the last value of location where points is equal to 28?	West Coast
21	What is the last value of A where B is less than c?	34	location, points	What is the last value of location where points is less than 34?	West Coast
22	What is the last value of A where B is less than c?	28	location, points	What is the last value of location where points is less than 28?	
23	What is the last value of A where B is more than c?	28	location, points	What is the last value of location where points is more than 28?	East Coast
24	What is the last value of A where B is more than c?	34	location, points	What is the last value of location where points is more than 34?	

Table 16: Templates 13–24 generated for Table 14 (part 2/3).

Index	Template	Value	Columns	Filled Template	Answer
25	What are the unique values of A where B is equal to c?	34	location, points	What are the unique values of location where points is equal to 34?	East Coast
26	What are the unique values of A where B is equal to c?	28	location, points	What are the unique values of location where points is equal to 28?	West Coast
27	What are the unique values of A where B is less than c?	34	location, points	What are the unique values of location where points is less than 34?	West Coast
28	What are the unique values of A where B is more than c?	28	location, points	What are the unique values of location where points is more than 28?	East Coast
29	What is the average of A where B is equal to c?	34	age, points	What is the average of age where points is equal to 34?	21
30	What is the average of A where B is less than c?	34	age, points	What is the average of age where points is less than 34?	25
31	What is the max of A where B is equal to c?	28	age, points	What is the max of age where points is equal to 28?	25
32	What is the min of A where B is equal to c?	34	age, points	What is the min of age where points is equal to 34?	21
33	What is the first value of A where B is equal to c?	21	team, age	What is the first value of team where age is equal to 21?	Red Hawks
34	What is the last value of A where B is equal to c?	25	team, age	What is the last value of team where age is equal to 25?	Blue Whales
35	What are the unique values of A where B is less than c?	25	team, age	What are the unique values of team where age is less than 25?	Red Hawks
36	What is the average of A where B is equal to c?	21	points, age	What is the average of points where age is equal to 21?	34

Table 17: Templates 25–36 generated for Table 14 (part 3/3).