MMTU: A Massive Multi-Task Table Understanding and Reasoning Benchmark

Junjie Xing[*]
University of Michigan

Yeye He[†]
Microsoft Corporation

Mengyu Zhou
Microsoft Corporation

Haoyu Dong
Microsoft Corporation

Shi Han
Microsoft Corporation

Lingjiao Chen
Microsoft Corporation

Dongmei Zhang
Microsoft Corporation

Surajit Chaudhuri
Microsoft Corporation

H. V. Jagadish
University of Michigan

Abstract

Tables and table-based use cases play a crucial role in many important real-world applications, such as spreadsheets, databases, and computational notebooks, which traditionally require expert-level users like data engineers, data analysts, and database administrators to operate. Although LLMs have shown remarkable progress in working with tables (e.g., in spreadsheet and database copilot scenarios), comprehensive benchmarking of such capabilities remains limited. In contrast to an extensive and growing list of NLP benchmarks, evaluations of table-related tasks are scarce, and narrowly focus on tasks like NL-to-SQL and Table-QA, overlooking the broader spectrum of real-world tasks that professional users face. This gap limits our understanding and model progress in this important area.

In this work, we introduce MMTU, a large-scale benchmark with around 28K questions across 25 real-world table tasks, designed to comprehensively evaluate models ability to understand, reason, and manipulate real tables at the expert-level. These tasks are drawn from decades' worth of computer science research on tabular data, with a focus on complex table tasks faced by professional users. We show that MMTU require a combination of skills – including table understanding, reasoning, and coding – that remain challenging for today's frontier models, where even frontier reasoning models like OpenAI GPT-5 and DeepSeek R1 score only around 69% and 57% respectively, suggesting significant room for improvement. We highlight key findings in our evaluation using MMTU and hope that this benchmark drives further advances in understanding and developing foundation models for structured data processing and analysis. Our code and data are available at `https://github.com/MMTU-Benchmark/MMTU` and `https://huggingface.co/datasets/MMTU-benchmark/MMTU`.

1 Introduction

Remarkable progress has been made in foundation models [43, 44, 72, 121, 37], partly thanks to an expanding array of large-scale benchmarking efforts. Prominent examples include benchmarks for general language understanding (e.g., GLUE [123], Super-GLUE [124], BIG-Bench [117], MMLU [78], MMLU-pro [127]), as well as benchmarks focused on coding and STEM reasoning (e.g., SWE-bench [13], GPQA-diamond [109], AIME [5], LiveCodeBench [11]). These efforts have

[1]Correspondence: jjxing@umich.edu
[2]Correspondence: yeyehe@microsoft.com

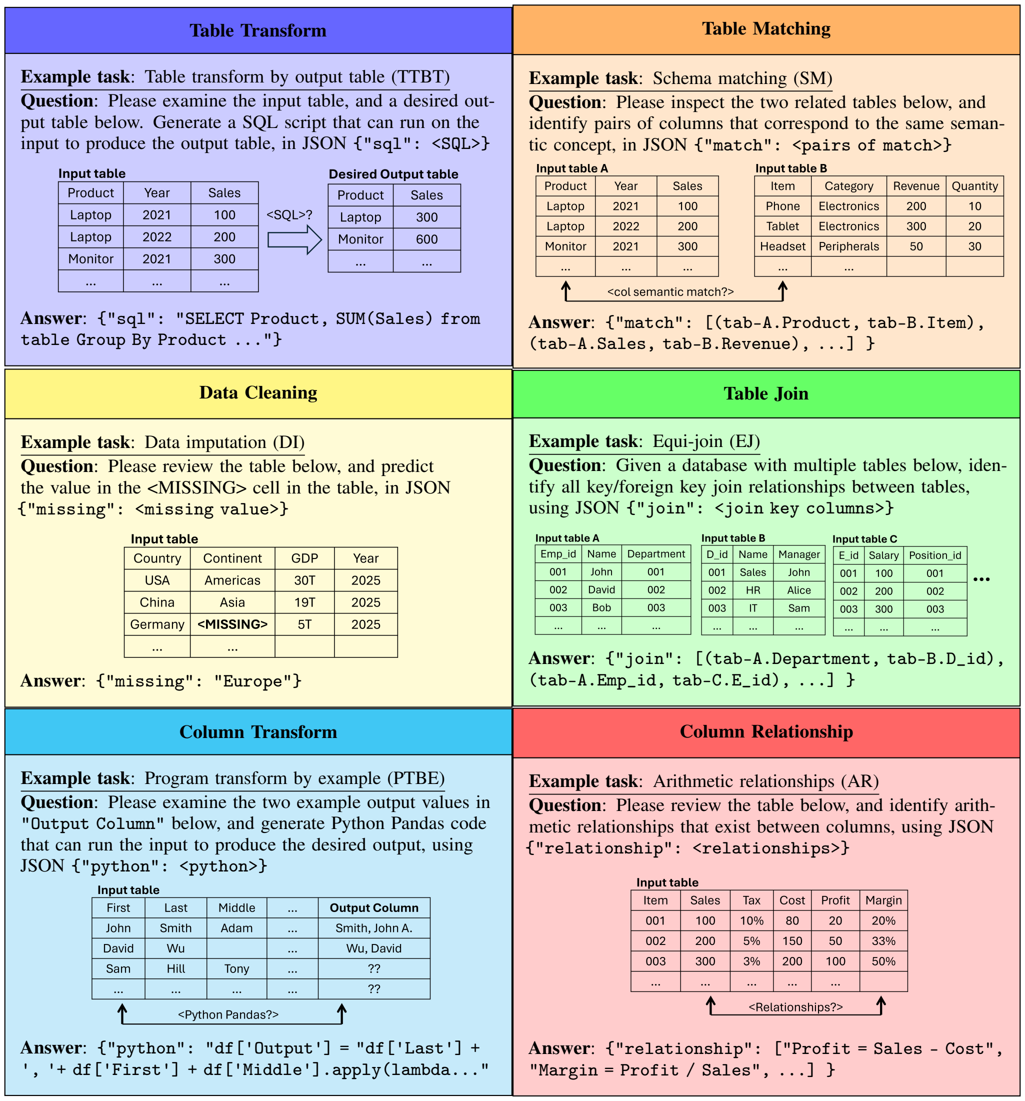


Table Transform

Example task: Table transform by output table (TTBT)

Question: Please examine the input table, and a desired output table below. Generate a SQL script that can run on the input to produce the output table, in JSON `{"sql": <SQL>}`

Input table

Product	Year	Sales
Laptop	2021	100
Laptop	2022	200
Monitor	2021	300
...	...	...

Desired Output table

Product	Sales
Laptop	300
Monitor	600
...	...

Answer: `{"sql": "SELECT Product, SUM(Sales) from table Group By Product ..."}`

Table Matching

Example task: Schema matching (SM)

Question: Please inspect the two related tables below, and identify pairs of columns that correspond to the same semantic concept, in JSON `{"match": <pairs of match>}`

Input table A

Product	Year	Sales
Laptop	2021	100
Laptop	2022	200
Monitor	2021	300
...	...	...

Input table B

Item	Category	Revenue	Quantity
Phone	Electronics	200	10
Tablet	Electronics	300	20
Headset	Peripherals	50	30
...	...		

Answer: `{"match": [(tab-A.Product, tab-B.Item), (tab-A.Sales, tab-B.Revenue), ...] }`

Data Cleaning

Example task: Data imputation (DI)

Question: Please review the table below, and predict the value in the <MISSING> cell in the table, in JSON `{"missing": <missing value>}`

Input table

Country	Continent	GDP	Year
USA	Americas	30T	2025
China	Asia	19T	2025
Germany	**<MISSING>**	5T	2025
...	...		

Answer: `{"missing": "Europe"}`

Table Join

Example task: Equi-join (EJ)

Question: Given a database with multiple tables below, identify all key/foreign key join relationships between tables, using JSON `{"join": <join key columns>}`

Input table A

Emp_id	Name	Department
001	John	001
002	David	002
003	Bob	003
...	...	...

Input table B

D_id	Name	Manager
001	Sales	John
002	HR	Alice
003	IT	Sam
...	...	...

Input table C

E_id	Salary	Position_id
001	100	001
002	200	002
003	300	003
...	...	...

...

Answer: `{"join": [(tab-A.Department, tab-B.D_id), (tab-A.Emp_id, tab-C.E_id), ...] }`

Column Transform

Example task: Program transform by example (PTBE)

Question: Please examine the two example output values in `"Output Column"` below, and generate Python Pandas code that can run the input to produce the desired output, using JSON `{"python": <python>}`

Input table

First	Last	Middle	...	**Output Column**
John	Smith	Adam	...	Smith, John A.
David	Wu		...	Wu, David
Sam	Hill	Tony	...	??
...	...	...	...	??

Answer: `{"python": "df['Output'] = "df['Last'] + ', '+ df['First'] + df['Middle'].apply(lambda..."`

Column Relationship

Example task: Arithmetic relationships (AR)

Question: Please review the table below, and identify arithmetic relationships that exist between columns, using JSON `{"relationship": <relationships>}`

Input table

Item	Sales	Tax	Cost	Profit	Margin
001	100	10%	80	20	20%
002	200	5%	150	50	33%
003	300	3%	200	100	50%
...	...	...	...	...	

Answer: `{"relationship": ["Profit = Sales - Cost", "Margin = Profit / Sales", ...] }`

Figure 1: Example questions sampled from different task categories in MMTU for illustrative purposes. Note that questions in MMTU all follow a standardized triplet format of: `(Instruction, Input-table(s), Ground-truth answer)`. The list of all tasks can be seen in Table 1.

played a critical role in deepening our understanding and accelerating the progress of foundation models.

Tables and table-based use cases are central to many real-world applications, including spreadsheets [21, 2], databases [17, 18], and computational notebooks [25, 19], which often require expert-level users such as data engineers, analysts, and database administrators to operate. While LLMs have shown great promise in working with tables [22, 16, 24], existing evaluations of table-tasks remain narrow in scope – primarily focusing on NL-to-SQL [92, 147, 138] and Table-QA [137, 58, 53] – and fail to reflect the broader spectrum of real-world tasks that professional users face.

In this work, we introduce MMTU, a large-scale and challenging benchmark comprising 28,136 questions across 25 real-world table tasks. It is designed to comprehensively evaluate models' ability to understand, reason, and manipulate real tables at the expert-level. The tasks we collect in MMTU are drawn from decades of computer science research on tabular data, with a particular focus on complex tasks that professional users face, as illustrated in Figure 1.

Our evaluation shows that the complex and technical nature of MMTU demands a combination of capabilities – including table understanding, reasoning, and coding – that remain challenging for today's frontier models. Even the top performing models, such as OpenAI GPT-5 and DeepSeek-R1

Table 1: Tasks and datasets in the MMTU benchmark. Most of these tasks have not traditionally been used to evaluate foundation models (except NL-to-code, Table QA, and KB mapping).

Task Category	Task Name	Task Description	Metric	References and Datasets	# Questions
Table Transform	table-transform-by-relationalization (TTBR)	Relationalize a table using transformation	Acc	[93, 39, 83, 132, 87]	230
	table-transform-by-output-schema (TTBS)	Synthesize transformation by output schema	Acc	[135, 113, 114]	685
	table-transform-by-output-table (TTBT)	Synthesize transformation by input/output tables	Acc	[40, 125, 122, 142]	86
Table Matching	Entity matching (EM)	Match rows refer to the same semantic entity	Acc	[65, 103, 146, 95, 105]	4228
	Schema matching (SM)	Match columns refer to the same concept	F1	[86, 145, 41, 100]	669
	Head value matching (HVM)	Match column-headers with cell-values	Acc	[94, 108]	900
Data Cleaning	data-imputation (DI)	Predict missing values in tables	Acc	[45, 68, 94]	1971
	error-detection (ED)	Detect erroneous cells in tables	F1	[60, 126, 101, 50]	1891
	list-to-table (L2T)	Split lists of undelimited values into table	Acc	[59, 46, 69, 90]	957
Table Join	semantic-join (SJ)	Predict semantic join between two tables	Acc	[75, 35, 64, 128]	130
	equi-join-detect (EJ)	Predict equi-joins between a set of tables	F1	[96, 56, 111, 81]	494
Column Transform	program-transform-by-example (PTBE)	Program transformation by input/output examples	Acc	[76, 70, 67, 116, 77]	558
	formula-by-context (FBC)	Predict formula based on table context	Acc	[52, 54]	3513
	semantic-transform-by-example (STBE)	Predict semantic transformations by examples	Acc	[75, 35, 64]	131
Column Relationship	arithmetic-relationship (AR)	Predict arithmetic-relationship (AR) in tables	F1	[115, 1]	818
	functional-relationship (FR)	Predict functional-relationship (FR) in tables	F1	[1, 106, 129]	267
	string-relationship (SR)	Predict string-relationship (SR) in tables	F1	[1, 76, 71]	765
Table understanding	Needle-in-a-haystack-table (NIHT)	Retrieve cell content in a table	Acc	[68]	999
	Needle-in-a-haystack-index (NIHI)	Retrieve index based on cell value	Acc	[68]	1000
NL-2-code	NL-2-SQL (NS)	Translate natural-language into SQL	Acc	[147, 92, 89, 138, 148]	2489
Table QA	Table Question Answering (TQA)	Answer questions based on tables	Acc	[131, 137, 57, 58]	1793
	Fact Verification (FV)	Verify facts based on tables	Acc	[53, 136, 141]	916
KB Mapping	Column type annotation (CTA)	Predict KB types based on column content	Acc	[82, 134, 118, 140, 80]	881
	Column property annotation (CPA)	Predict KB property for a pair of columns	Acc	[82, 63, 104]	873
	Cell entity annotation (CEA)	Predict KB entity for a table cell	Acc	[82, 66, 42, 130, 98]	892
Total					28,136

, achieve only 69.6% and 57.9% on MMTU, suggesting substantial room for improvement, and highlighting MMTU as a strong testbed for models aspiring toward general human-level intelligence, e.g., to meet or surpass the top 10% of skilled adults in diverse technical tasks like those characterized in [102].

We perform extensive experiments benchmarking a large collection of models using MMTU, and performed extensive analysis. Some of the key findings from our evaluation include:

- LLMs demonstrate strong potential in understanding and manipulating tabular data. Newer and larger models substantially outperform older and smaller ones, indicating significant advancements in table-related capabilities as captured by MMTU.
- Reasoning-focused models, such as OpenAI GPT-5 and DeepSeek R1, show a clear advantage over general-purpose chat models like GPT-5-Chat and DeepSeek-V3. The top reasoning models outperform the top chat models by over 10 percentage points (Table 3), underscoring the complexity of the tasks (which often require coding in SQL/Pandas) and the importance of reasoning in MMTU.
- Unlike earlier models, today's frontier models are less sensitive to how tables are formatted and serialized (e.g., markdown/CSV/JSON/HTML), reflecting general progress in models' abilities in understanding diverse data formats (Figure 8).
- LLMs still struggle with long table context, or large tables with many rows and columns. Complex tasks requiring holistic reasoning across cell values, especially in the column direction, remains challenging when the table context is long (Figure 6 and Figure 11).
- LLM performance can degrade under table-level perturbations such as row or column shuffling, even when these changes are supposed to be semantically invariant in the context of tables (Figure 7). This sensitivity points to possible limitations in models' ability to understand tables in a robust manner.

We hope MMTU can serve as a valuable addition to the growing landscape of model benchmarks, helping to track progress, identify limitations, and ultimately drive further advancements in this important area of using LLMs for table tasks.

2 Related Work

Large-scale benchmarks for foundation models. The rapid advancement of foundation models has made benchmark evaluation ever more important. Prominent large-scale benchmarks, such as GLUE [123] (9 NLP tasks), Super-GLUE [124] (10 NLP tasks), Big-BENCH [117] (204 tasks), MMMU [78] (15,908 questions), MMMU-pro [127] (12,032 questions), MMLU [139] (11,550 multi-modal questions) etc., offer comprehensive evaluations of model capabilities. However, as models improve rapidly, benchmarks can become saturated quickly (e.g., in the matter of a few years), prompting newer and more challenging benchmarks [124, 127]. All of these benchmarking efforts have nevertheless played a crucial role in measuring and stimulating the development of foundation models.

To contribute to this growing landscape, our new MMTU benchmark comprises 28,136 challenging questions across diverse table tasks that expert users would face, which is comparable in scale with prior efforts such as MMMU and MMLU. It complements existing benchmarks, by enabling comprehensive evaluation of foundation models in the important yet underexplored area of table reasoning and understanding.

Benchmarks for reasoning. Reasoning has recently emerged as a key challenge for foundation models. Beyond general intelligence benchmarks (e.g., GPQA Diamond [109], AGIEval [149], HLE [8]), there are specialized benchmarks targeting mathematical reasoning (e.g., AIME [5], MathVista [12], IMO [9]), coding (e.g., SWE-bench [13], CodeForce [7], LiveCodeBench [11], IOI [10]), and multi-modal reasoning (e.g., MMMU [139], ARC [6]). These benchmarks have become important tools for evaluating models' ability to tackle complex reasoning tasks, but can also get saturated quickly (e.g., GSM8k [61], Math500 [79], HuamEval [49]), making it necessary to create new and more challenging benchmarks.

We show that our MMTU benchmark can serve to complement existing reasoning benchmarks, by enabling evaluation on complex table-based tasks that require two-dimensional table understanding, coding, and logical reasoning. MMTU extends current reasoning benchmarks into the important yet underexplored domain of tabular data, which underpins many real-world applications.

Existing benchmarks relating to tables. Given the importance of table data, benchmarks have been developed in the ML and NLP community to evaluate model ability on tables, which however usually focus on a small set of table tasks such as NL-2-SQL [147, 92, 89, 138, 148] and Table-QA [53, 131, 137, 57, 58]. In contrast, MMTU expands the scope of current evaluations by incorporating 19 additional table tasks drawn from decades of research in communities such as data management and programming languages, resulting in a more comprehensive evaluation framework for assessing LLM capabilities on tabular data.

More recently, spreadsheet-centric benchmarks have emerged, including Spreadsheet-Bench [99], SheetCopilotBench [91], and Sheet-RM [55]. While these are important in the spreadsheet domain, these efforts are typically limited in scale (containing a few hundred cases), and are closely tied to specific file formats (e.g., .xlsx) and software environments. In comparison, MMTU focuses on general-purpose tabular data that applies broadly across spreadsheet, database, and computational notebook settings, enabling more scalable and format-independent evaluation of foundation models.

3 MMTU Benchmark for Tables

3.1 Benchmark overview

We introduce our Massive Multi-task Table Understanding and Reasoning (MMTU) benchmark, designed to evaluate models table understanding and reasoning capabilities at the expert-level, across a wide range of real-world tasks that would typically be performed by professional data engineers, data scientists, and database administrators.

The benchmark comprises 28,136 complex table-centric questions over 61,763 real tables, in 25 distinct task categories. Each question has a standardized format of "`<Instruction, Input-Table(s), Ground-truth answer>`", like illustrated in example questions in Figure 1. Detailed statistics of the benchmark can be found in Table 2, which highlight the diversity and complexity of the questions in MMTU.

These questions are meticulously collected and curated based on decades of computer science research in areas beyond ML/NLP – such as data management and programming languages – drawing on expert-labeled datasets developed over many years by researchers in these communities, as we will detail below which we will describe below.

3.2 Data curation workflow

Figure 2 illustrates the key steps in the overall workflow of our data curation process for producing MMTU. We detail each step in turn below.

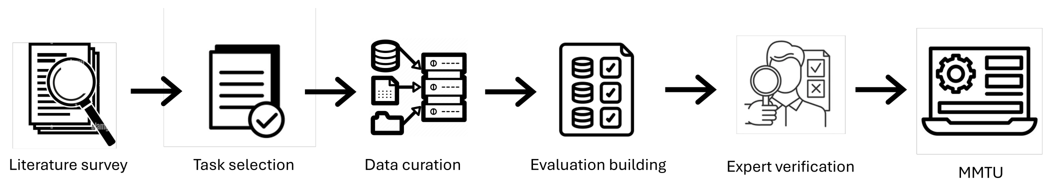


Figure 2: MMTU data curation workflow: we survey real-world table tasks from the literature, select 25 user-facing tasks with objective evaluation criteria, curate questions from 52 datasets, develop evaluation scripts and verify the results for these tasks, before arriving at the MMTU benchmark.

Statistics	Number
Total Questions	28,136
Total tasks / datasets	25/52
Total tables	61,763
Coding Questions	7,331 (26.1%)
- SQL questions	2,489 (18.8%)
- Python Pandas questions	1,329 (4.5%)
- Spreadsheet Formula questions	3,513 (11.9%)
Non-coding Questions	20,805 (73.9%)
Questions with tables	28,136
- Questions with 1 table	20,048 (71.3%)
- Questions with 2 tables	5,285 (18.8%)
- Questions with 3 or more tables	2,803 (10.0%)
Table characteristics	
- Average table row count	2,659
- Average table column count	11
- Average table cell count	33,251
Table sources	
- Web tables	46,264 (74.9%)
- Spreadsheet tables	4,540 (7.4%)
- Relational tables	10,959 (17.7%)

Table 2: Statistics of benchmark questions

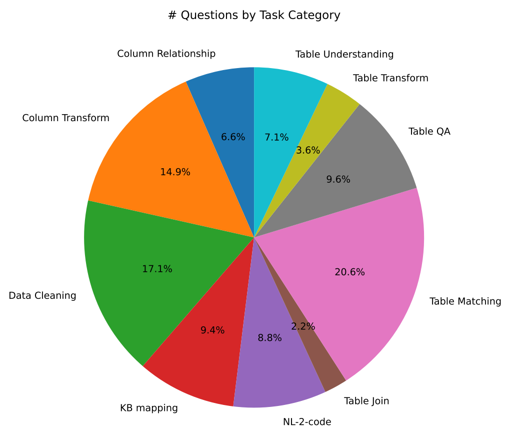


Figure 3: Question distribution by task category

Literature survey. To ensure our table tasks reflect real-world challenges, we draw on our experience working on related problems, that many challenging predictive table tasks have been studied in the decades worth of computer science research, particularly in data management (SIGMOD/VLDB), programming languages (PLDI/POPL), and web data (WWW/WSDM) communities. We conduct a systematic survey of publications from these venues over the past two decades, leveraging a combination of keyword searches of paper titles, and DeepResearch-like tools [20] (where we specify detailed requirements for papers in these venues), to arrive at a promising set of papers and possible candidate tasks.

Task selection. We manually examine candidate tasks described in the surveyed papers from the previous step, and select tasks that are:

(1) Real user-facing tasks, involving data tables that would otherwise require expert-level humans to perform. (We therefore exclude system-level predictive table tasks focused on performance improvements, such as query optimization [85, 88] and cardinality estimation [73, 84]);
(2) Objectively evaluable tasks, that come with unique manually-labeled ground truth. (We therefore exclude tasks such as table summarization [144, 38, 74, 107, 48] and table augmentation [143, 133], which lack unique ground-truth and may require subjective fuzzy LLM-based evaluations);
(3) Tasks based on real-world data tables, which can be real web tables, spreadsheet tables, or relational tables, etc. (We exclude tasks and datasets based on synthetic or perturbed data).
After the selection step, we arrive at 25 different tasks (listed in Table 1) from 52 diverse benchmark datasets (can be seen in Table 5 in the Appendix). We note that a majority of these real-world tasks (all except the last 3 categories in Table 1, NL-2-code, Table-QA and KB-mapping) have not been used to evaluate foundation models. A summary of these tasks is described in more detail in Appendix B.

Data standardization and curation. To accommodate the heterogeneity across the 52 benchmark datasets (which have diverse data formats, ground-truth labels, and task definitions), we next standardize the questions in each dataset into a consistent "`<Instruction, Table(s), Ground-truth>`"

format. Figure 1 shows examples of the triplet format for different tasks. This enables consistent representation across tasks, facilitating the integration of diverse table tasks within a single benchmark framework, and allowing different LLMs to be plugged in for easy model prediction and evaluation.

To ensure the quality of MMTU, we conduct an LLM-based quality check using o4-mini. The model is prompted to evaluate each question for two aspects: (1) whether the question is ambiguous (since tasks like NL-code can sometimes be inherently ambiguous [112]), and (2) whether the ground-truth answer is correct. Approximately 8% of the original questions were flagged as either ambiguous or incorrect and subsequently removed to enhance benchmark quality.

As an additional safeguard, we also use LLM to exclude any benchmark instances that might pose privacy or security risks. To maintain diversity across sources, we cap the number of questions drawn from any single dataset at 1000.

The final composition of the questions is reported in Table 1, with more statistics shown in Table 2 and Figure 3.

Evaluation framework. In contrast to benchmarks like MMLU [78] and MMMU [139], which primarily use multiple-choice formats (where evaluation involves comparing a predicted option such as “A/B/C/D” against a single-letter ground truth), real-world table tasks performed by professional experts are often more complex and nuanced. These tasks, such as code generation or structured reasoning, cannot be adequately evaluated using multiple-choice alone. In MMTU, we instead adopt a structured yet open-ended answer format (see Figure 1 for examples) for prediction and evaluation.

To support evaluations beyond simple string comparisons, we designed a lightweight evaluation framework that supports diverse evaluations in table tasks, including execution-based evaluation (for SQL and Python generation), and structured output evaluation (e.g., comparing an unordered JSON list against ground truth). Our framework is also extensible, making it easy to incorporate new task types and evaluation metrics. Details of our evaluation framework can be seen in [15].

Expert verification. As a final verification step, we sample 20 questions per task and employ domain experts with years of experience to manually review and verify that (1) raw data is integrated correctly, (2) the ground-truth aligns with human intuition and passes verification, (3) the reference instruction properly reflects the task to produce the desired output, and (4) the evaluation script is set up correctly to correctly evaluate model predictions against ground-truth.

After completing all data curation steps in the workflow illustrated in Figure 2, we obtain a total of 28,136 questions that form our MMTU benchmark, with main statistics summarized in Table 2.

3.3 Broader discussions

Limitations. A main limitations of our benchmark is how our tasks are sampled and selected. Like discussed in Section 3.2, for ease of evaluation, we include only tasks that can be objectively evaluated, and omit ones that are subjective or creative in nature (e.g., table summarization, generation, and enrichment) that are also important to users, but not included in the benchmark.

In addition, since we sampled table tasks from the existing research literature, which naturally introduces biases, as it omits tasks that are important in practice but not well studied in the literature, or tasks that lack good labeled data (e.g., multi-turn table manipulation).

Lastly, while human experts often read tables visually on two dimensional grid (which makes two dimensional spatial reasoning easy), our current evaluations only use text-based input, and do not consider multi-modal input. Extending the benchmark to multi-modal table input is an interesting direction for future work.

Broader impacts. Our benchmark is designed to evaluate LLMs performance on challenging expert-level table tasks, with the goal of identifying model shortcomings and stimulating model improvements. We hope this can lead to more capable models, to better assist human users in scenarios such as spreadsheet-copilot and database-copilot.

We make our best efforts to exclude any content that may raise privacy or security concerns, contain explicit material, depict violence, or be otherwise sensitive. For example, we manually review instructions and datasets at the task level, and employ GPT-4o at the data record level, to remove any that may have privacy concerns, in order to minimize potential negative effect of the data.

Model Type	Model	MMTU result	Cost per question (US$)
Reasoning	GPT-5	**0.696 ± 0.01**	0.01727
	o3	0.691 ± 0.01	0.01539
	GPT-5-mini	0.667 ± 0.01	0.00276
	Gemini-2.5-pro	0.665 ± 0.01	0.00790
	o4-mini (2024-07-18)	0.660 ± 0.01	0.00993
	Grok-3-mini	0.645 ± 0.01	0.00111
	Gemini 2.5 Flash	0.625 ± 0.01	0.00199
	Deepseek-R1	0.579 ± 0.01	0.00167
	gpt-oss-120b	0.543 ± 0.01	0.00050
	Qwen3-235B-A22B-Thinking-2507	0.529 ± 0.01	0.00068
	Qwen3-32B (thinking)	0.506 ± 0.01	0.00017
	gpt-oss-20b	0.478 ± 0.01	0.00040
	Qwen3-8B (thinking)	0.473 ± 0.01	0.00041
Chat	GPT-5-Chat	**0.577 ± 0.01**	0.00534
	Deepseek-V3	0.555 ± 0.01	0.00095
	Qwen3-235B-A22B-Instruct-2507	0.524 ± 0.01	0.00044
	GPT-4o (2024-11-20)	0.507 ± 0.01	0.01019
	Llama-4-Maverick-17B-128E-Instruct-FP8	0.490 ± 0.01	0.00066
	Llama-3.3-70B	0.454 ± 0.01	0.0015
	Mistral-Large-2411	0.446 ± 0.01	0.01066
	Mistral-Small-2503	0.417 ± 0.01	0.00273
	GPT-4o-mini (2024-07-18)	0.400 ± 0.01	0.00061
	Llama-4-Scout-17B-16E-Instruct	0.393 ± 0.01	0.00036
	Qwen3-32B (no thinking)	0.379 ± 0.01	0.00007
	Qwen3-8B (no thinking)	0.353 ± 0.01	0.00015
	Qwen2.5-7B-Instruct	0.310 ± 0.01	0.00010
	Llama-3.1-8B	0.268 ± 0.01	0.00003

Table 3: Overall performance and cost results of chat and reasoning models. Cost per question is calculated based on public pricing information on `https://openrouter.ai/` as of Aug 2025. Results for top-performing models, such as GPT-5 and GPT-5-Chat, are averaged over 3 runs.

4 Experiments

Experimental setup. In all our experiments reported below, we use publicly available model endpoints for inference [4, 23] with default parameter settings. All of our code and data are publicly available at [15] for future research.

4.1 Overall performance

We benchmark a range of frontier open-source and proprietary models using MMTU. Table 3 gives an overview of their performance, as well as their cost information. Notably, reasoning models such as GPT-5, o3 and Gemini-2.5-pro, significantly outperform chat-oriented models, with OpenAI GPT-5 achieving the best score at 69.6%, highlighting the challenging nature of MMTU. Among open-source reasoning models, DeepSeek R1 attains the best performance with a score of 57.9%, which while strong, reveals a potential gap between leading proprietary and open-source models.

Our analysis of traces generated by reasoning models reveals that they excel on MMTU over chat models, because of their strong coding skills, and abilities to break complex tasks on large tables, into a sequence of more manageable sub-tasks with smaller table context (i.e., subsets of rows and columns relevant to the task at hand).

From a cost-efficiency perspective[1], looking at the "cost per question" column in the table, several light-weight reasoning models (e.g., GPT-5-mini and Gemini-2.5-flash) are quite competitive in terms of both quality and cost (even after accounting for their intermediate thinking tokens), making them cost-effective choices for complex table tasks.

While we benchmark a broad range of models, our detailed analysis in the remainder of the paper will focus on four leading models to avoid clutter: two reasoning models (OpenAI GPT-5 and DeepSeek R1) and two chat-oriented models (OpenAI GPT-5-Chat and DeepSeek V3).

Figure 5 provides a detailed comparison across 10 task categories, between these four leading models. While reasoning models outperform chat-based models, the relative empty nature of the radar chart reveals substantial gaps in model performance, highlighting opportunities for improvement. For instance, we can see that models still face difficulties with table-centric coding tasks (e.g., Column

[1]Price-per-token figures are sourced from the respective API providers [27–34].

Transform, Table Transform, etc.). Tasks that require holistic reasoning across multiple tables and multiple columns (e.g., Table Join, Column Relationship, Data Cleaning, etc.) also remain challenging. We will present an error-analysis in Section 4.3.

A more granular breakdown across all 25 individual tasks is shown in Figure 4, where reasoning models (represented by shaded bars) noticeably outperform chat models on many complex tasks. Additional dataset-level performance results can be found in Table 5 in the appendix.

4.2 Detailed analysis and sensitivity

We highlight key analysis in this section, including long contexts, robustness to table perturbations, and sensitivity to format variations.

Long table context. Figure 6 shows the impact of long table context on model performance. In this analysis, we bucketize all questions within each task into four quartiles based on the token length of the associated tables. We then evaluate model accuracy within each quartile and aggregate the results across tasks, as shown in the figure. Across all four frontier models, performance consistently declines as the table context length increases (from left to right within each group).

These findings suggest that, despite recent advances in long-context LLMs [51, 36], long contexts remain a significant challenge for tables. In Appendix E, we use a detailed experimental comparison between the standard "needle-in-a-haystack (NIH)" [3, 110], and our table-based "needle-in-a-haystack in table (NIHT)" test – while frontier models perform nearly perfectly on NIH, their performance drops sharply on NIHT, revealing fundamental limitations in their ability to handle long table contexts.

Robustness to table permutation. Figure 7 illustrates model performance under different table permutations. Specifically, we randomly shuffle rows and/or columns in input tables[2], with the 4 bars in each group representing: (1) no shuffle, (2) row-only shuffle, (3) column-only shuffle, and (4) both row and column shuffle. Recall that unlike natural-language text, two dimension relational tables are permutation-invariant [94, 62], meaning that shuffling rows and columns should generally not change their semantic meanings and the associated tasks (e.g., the tasks in Figure 1 will remain the same, even if the rows/columns in the associated tables are shuffled).

However, the results show a consistent decline in model performance as we move from no permutation to row, column, and full (row + column) shuffling. Notably, column shuffling leads to a steeper decline than row shuffling. This suggests that despite their capabilities, language models that are pretrained primarily on linear, left-to-right text can remain sensitive to the structural ordering of tables, indicating a lack of robustness to alternate but semantically equivalent table layouts.

Table format variations. Figure 8 shows different models' performance using common table input formats: Markdown, CSV, JSON, and HTML. We observe that unlike prior studies that showed notable sensitivity of LLMs [119] to table formats, our results indicate that today's frontier models, especially reasoning ones (GPT-5 and R1), are becoming less sensitive to these format variations (except HTML that still lags behind other formats). This reduces the need for format-specific optimization as models continue to improve. For chat models (GPT-5-Chat and DeepSeek-V3), we note that using JSON has an advantage on MMTU, mainly because it is easier for models to identify value/column correspondence in the JSON format, especially in long table context settings (e.g., in needle-in-a-haystack tasks like NIHT and NIHI shown in Table 1).

4.3 Error analysis

We sampled 10 questions from each task, to manually analyze the underlying reason of these errors. We categorize all model errors into 4 main categories below:

Table understanding (38%). Table understanding is the largest category of errors in our analysis. Figure 9 shows a simple example from the Data Imputation task that is intuitive to understand. While the model correctly identifies the person's name for the missing cell, it filled in the value "`D. H. McFadden`" that is in an abbreviated form, inconsistent with other values in the same column that use the full-name format (the correct answer should be "`David Henry McFadden`").

[2]In shuffling columns, we keep the 3 left-most columns in all tables intact, as these are likely the "key columns" or "entity columns" in tables [47, 45] in tables, to best preserve the meaning of these tables.

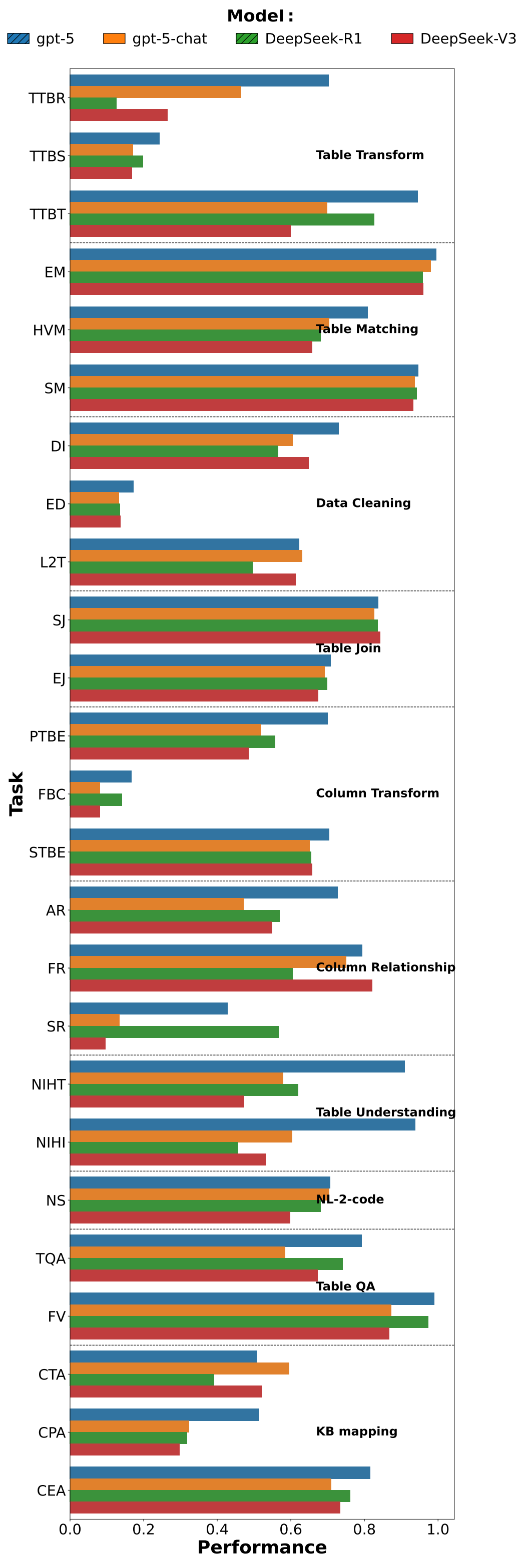


Figure 4: Performance comparison in all 25 tasks, for chat-models GPT-5-Chat and DeepSeek-V3 (solid bars), as well as reasoning models DeepSeek-R1 and OpenAI GPT-5 (shaded bars). On average, reasoning models are noticeably better.

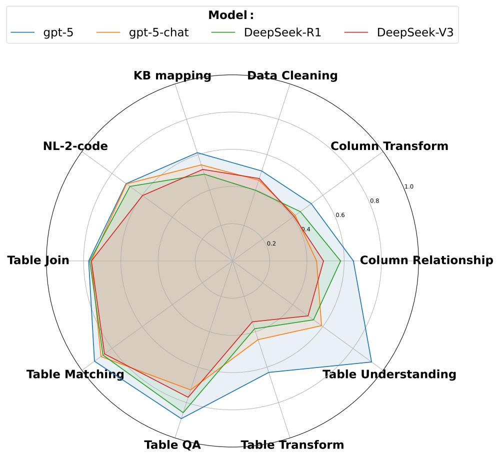


Figure 5: Performance comparison of frontier models in all 10 task-categories.

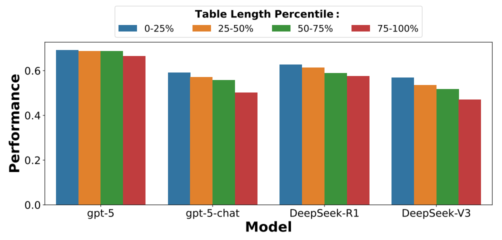


Figure 6: Impact of long table context on model performance.

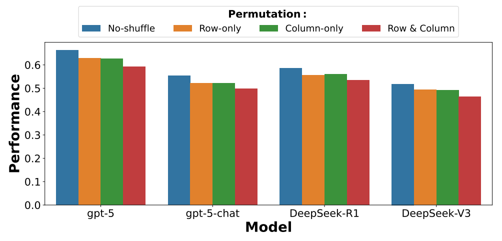


Figure 7: Impact of table-level permutation on model performance.

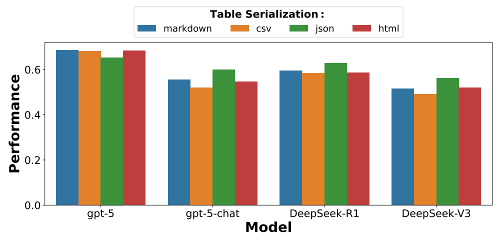
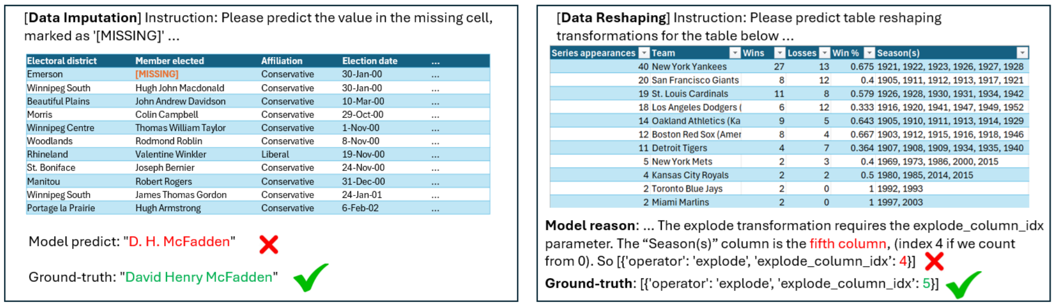


Figure 8: Impact of table format (markdown/csv/json/html) on model performance.


[**Data Imputation**] Instruction: Please predict the value in the missing cell, marked as '[MISSING]' ...

Electoral district	Member elected	Affiliation	Election date	...
Emerson	[MISSING]	Conservative	30-Jan-00	...
Winnipeg South	Hugh John Macdonald	Conservative	30-Jan-00	...
Beautiful Plains	John Andrew Davidson	Conservative	10-Mar-00	...
Morris	Colin Campbell	Conservative	29-Oct-00	...
Winnipeg Centre	Thomas William Taylor	Conservative	1-Nov-00	...
Woodlands	Rodmond Roblin	Conservative	8-Nov-00	...
Rhineland	Valentine Winkler	Liberal	19-Nov-00	...
St. Boniface	Joseph Bernier	Conservative	24-Nov-00	...
Manitou	Robert Rogers	Conservative	31-Dec-00	...
Winnipeg South	James Thomas Gordon	Conservative	24-Jan-01	...
Portage la Prairie	Hugh Armstrong	Conservative	6-Feb-02	...

Model predict: "D. H. McFadden" ✗

Ground-truth: "David Henry McFadden" ✓

[**Data Reshaping**] Instruction: Please predict table reshaping transformations for the table below ...

Series appearances	Team	Wins	Losses	Win %	Season(s)
40	New York Yankees	27	13	0.675	1921, 1922, 1923, 1926, 1927, 1928
20	San Francisco Giants	8	12	0.4	1905, 1911, 1912, 1913, 1917, 1921
19	St. Louis Cardinals	11	8	0.579	1926, 1928, 1930, 1931, 1934, 1942
18	Los Angeles Dodgers (	6	12	0.333	1916, 1920, 1941, 1947, 1949, 1952
14	Oakland Athletics (Ka	9	5	0.643	1905, 1910, 1911, 1913, 1914, 1929
12	Boston Red Sox (Amer	8	4	0.667	1903, 1912, 1915, 1916, 1918, 1946
11	Detroit Tigers	4	7	0.364	1907, 1908, 1909, 1934, 1935, 1940
5	New York Mets	2	3	0.4	1969, 1973, 1986, 2000, 2015
4	Kansas City Royals	2	2	0.5	1980, 1985, 2014, 2015
2	Toronto Blue Jays	2	0	1	1992, 1993
2	Miami Marlins	2	0	1	1997, 2003

Model reason: ... The explode transformation requires the explode_column_idx parameter. The "Season(s)" column is the fifth column, (index 4 if we count from 0). So [{'operator': 'explode', 'explode_column_idx': 4}] ✗

Ground-truth: [{'operator': 'explode', 'explode_column_idx': 5}] ✓

Figure 9: (Left): An example "table understanding" error in Data Imputation: while the model knows the person's name for the missing cell, it misses the table context and uses an abbreviated form of the name, instead of the full-name in the ground-truth. (Right): Another example "table understanding" error in Table Reshaping: while the model knows the correct transforms to perform ("explode" in Pandas), it miscounts the column index for the target column "Season(s)", which should be the sixth column instead of the fifth based on the reasoning trace, which leads to an incorrect transformation.

We observe that models are prone to errors when handling long table contexts, such as multiple tables or large tables with many rows and columns, as illustrated in Figure 6. Figure 9 (Right) shows a concrete example of the issue in the task of table-reshaping (TTBR in Table 1), where the reasoning trace shows that the model miscalculates the column index of the target column. This issue of long-context table is also notable in tasks like List-to-Table (L2T), Data Imputation (DI), Equi-Join (EJ), and NL-to-SQL (NS), etc.

A more detailed analysis of challenges in long-context tables is provided in Appendix E, where we examine a table-specific variant of the "needle-in-a-haystack" task, to understand why long-context tables remain challenging for models.

Reasoning and coding (28%). We find models can often make mistakes on tasks that require coding and reasoning on top of tables. For instance, for tasks in Table Transform, Column Transform, and NL-2-Code categories, models can generate code (SQL or Pandas) that is "close" to the correct answer, but misses important details that require reasoning over table context holistically (e.g., data format across all cells in the same column), which leads to incorrect results.

Knowledge (18%). For tasks in the categories of KB mapping (CEA and CTA), Semantic Join (SJ), and Data Imputation (DI), we find models can sometimes hallucinate facts (e.g., in the case of CEA, creating knowledge base references that do not exist), or recite inaccurate facts (e.g., in the case of Semantic Join and Data Imputation).

Other (15%). The remaining issues, such as result extraction, timeout and context limitation in response generation, and possible ambiguity in questions/ground-truth, fall in this category.

LLM-based error analysis. In addition, we complement the manual error analysis, by instructing LLM to inspect the question and ground-truth, to classify model prediction errors into these categories. We present the corresponding results in Appendix F in the interest of space.

5 Conclusion

In this work, we present MMTU, a comprehensive benchmark designed to assess foundation models on a broad range of real-world table tasks. By focusing on real-world tasks that professional data engineers, data analysts, and database administrators often have to face, MMTU poses expert-level challenges for foundation models in table understanding and reasoning.

Future work includes broadening the scope of table tasks to cover those that are important in practice but underrepresented in the existing research literature, as well as incorporating more subjective or creative tasks such as data generation, summarization, and enrichment. Another promising direction is the integration of multi-modal models for evaluation, which may have an advantage over text-only models on tasks that require two-dimensional table understanding.

References

[1] Auto-relate: A unified approach to discovering reliable functional relationships leveraging statistical tests. https://drive.google.com/drive/folders/1opO-tglvyJ6_yG37ZbMob57dLRjBAuMI.

[2] Google sheets. https://workspace.google.com/products/sheets/.

[3] Llm test: Needle in a haystack. https://github.com/gkamradt/LLMTest_NeedleInAHaystack.

[4] Azure OpenAI inference. https://learn.microsoft.com/en-us/azure/ai-services/openai/reference.

[5] Aime: American invitational mathematics examination. https://maa.org/math-competitions/american-invitational-mathematics-examination-aime, .

[6] Arc challenge. https://arcprize.org/, .

[7] codeforce competition. https://codeforces.com/contests, .

[8] Humanity's last exam. https://agi.safe.ai/, .

[9] Imo: International mathematics olympiad questions for model testing. https://openai.com/index/introducing-openai-o1-preview/, .

[10] Ioi: International olympiad in informatics questions for model testing. https://openai.com/index/learning-to-reason-with-llms/, .

[11] Livecodebench. https://livecodebench.github.io/, .

[12] Mathvista: Evaluating math reasoning in visual contexts. https://mathvista.github.io/, .

[13] Swe-bench. https://www.swebench.com/, .

[14] Claude 3 release annoucement. https://www.anthropic.com/news/claude-3-family?utm_source=chatgpt.com.

[15] MMTU: Code and data. Benchmark data: https://huggingface.co/MMTU-benchmark, Evaluation code: https://github.com/MMTU-Benchmark/MMTU.

[16] Database copilot. https://learn.microsoft.com/en-us/azure/azure-sql/copilot/copilot-azure-sql-overview?view=azuresql, .

[17] Oracle. https://www.oracle.com/, .

[18] Sql server. https://www.microsoft.com/en-us/sql-server, .

[19] Databricks notebooks. https://www.databricks.com/product/collaborative-notebooks, .

[20] Arc challenge. https://openai.com/index/introducing-deep-research/.

[21] Microsoft excel. https://www.microsoft.com/en-us/microsoft-365/excel, .

[22] Excel copilot. https://support.microsoft.com/en-us/office/get-started-with-copilot-in-excel-d7110502-0334-4b4f-a175-a73abdfc118a, .

[23] Models endpoints in Azure AI Foundry. https://learn.microsoft.com/en-us/azure/ai-foundry/concepts/models-featured#cohere.

[24] Gemini in bigquery. https://cloud.google.com/gemini/docs/bigquery/overview.

[25] Jupyter notebooks. https://jupyter.org/.

[26] The llama 4 herd: The beginning of a new era of natively multimodal ai innovation. `https://ai.meta.com/blog/llama-4-multimodal-intelligence/`.

[27] DeepSeek r1: price info (Retrieved in 05/2025). `https://api-docs.deepseek.com/quick_start/pricing`, .

[28] GPT-4o: price info (Retrieved in 05/2025). `https://platform.openai.com/docs/pricing`, .

[29] GPT-4o-mini: price info (Retrieved in 05/2025). `https://platform.openai.com/docs/pricing`, .

[30] Llama 3: price info (Retrieved in 05/2025). `https://azuremarketplace.microsoft.com/en-us/marketplace/apps/metagenai.meta-llama-3-1-8b-instruct-offer?tab=PlansAndPrice`, .

[31] Llama 3: price info (Retrieved in 05/2025). `https://azuremarketplace.microsoft.com/en-us/marketplace/apps/metagenai.llama-3-3-70b-instruct-offer?tab=PlansAndPrice`, .

[32] Mistral Large 2411: price info (Retrieved in 05/2025). `https://azuremarketplace.microsoft.com/en-us/marketplace/apps/000-000.mistral-large-2411?tab=PlansAndPrice`, .

[33] Mistral Small 2503: price info (Retrieved in 05/2025). `https://azuremarketplace.microsoft.com/en-us/marketplace/apps/000-000.mistral-small-2503?tab=PlansAndPrice`, .

[34] o4-mini: price info (Retrieved in 05/2025). `https://platform.openai.com/docs/pricing`, .

[35] Ziawasch Abedjan, John Morcos, Ihab F Ilyas, Mourad Ouzzani, Paolo Papotti, and Michael Stonebraker. Dataxformer: A robust transformation discovery system. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 1134–1145. IEEE, 2016.

[36] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[37] Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

[38] Junwei Bao, Duyu Tang, Nan Duan, Zhao Yan, Yuanhua Lv, Ming Zhou, and Tiejun Zhao. Table-to-text: Describing table region with natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[39] Daniel W Barowy, Sumit Gulwani, Ted Hart, and Benjamin Zorn. Flashrelate: extracting relational data from semi-structured spreadsheets using examples. *ACM SIGPLAN Notices*, 50 (6):218–228, 2015.

[40] Rohan Bavishi, Caroline Lemieux, Roy Fox, Koushik Sen, and Ion Stoica. Autopandas: neural-backed generators for program synthesis. *Proceedings of the ACM on Programming Languages*, 3(OOPSLA):1–27, 2019.

[41] Zohra Bellahsene, Angela Bonifati, Fabien Duchateau, and Yannis Velegrakis. *On evaluating schema matching and mapping*. Springer, 2011.

[42] Chandra Sekhar Bhagavatula, Thanapon Noraset, and Doug Downey. Tabel: Entity linking in web tables. In *International Semantic Web Conference*, pages 425–441. Springer, 2015.

[43] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[44] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[45] Michael J Cafarella, Alon Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. Webtables: exploring the power of tables on the web. *Proceedings of the VLDB Endowment*, 1(1):538–549, 2008.

[46] Michael J Cafarella, Alon Y Halevy, Yang Zhang, Daisy Zhe Wang, and Eugene Wu. Uncovering the relational web. In *WebDB*, pages 1–6. Citeseer, 2008.

[47] Kaushik Chakrabarti, Surajit Chaudhuri, Zhimin Chen, Kris Ganjam, Yeye He, and W Redmond. Data services leveraging bing's data assets. *IEEE Data Eng. Bull.*, 39(3):15–28, 2016.

[48] Jieying Chen, Jia-Yu Pan, Christos Faloutsos, and Spiros Papadimitriou. Tsum: fast, principled table summarization. In *Proceedings of the Seventh International Workshop on Data Mining for Online Advertising*, pages 1–9, 2013.

[49] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[50] Qixu Chen, Yeye He, Raymond Chi-Wing Wong, Weiwei Cui, Song Ge, Haidong Zhang, Dongmei Zhang, and Surajit Chaudhuri. Auto-test: Learning semantic-domain constraints for unsupervised error detection in tables. *arXiv preprint arXiv:2504.10762*, 2025.

[51] Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation. *arXiv preprint arXiv:2306.15595*, 2023.

[52] Sibei Chen, Yeye He, Weiwei Cui, Ju Fan, Song Ge, Haidong Zhang, Dongmei Zhang, and Surajit Chaudhuri. Auto-formula: Recommend formulas in spreadsheets using contrastive learning for table representations. *Proceedings of the ACM on Management of Data*, 2(3): 1–27, 2024.

[53] Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. Tabfact: A large-scale dataset for table-based fact verification. *arXiv preprint arXiv:1909.02164*, 2019.

[54] Xinyun Chen, Petros Maniatis, Rishabh Singh, Charles Sutton, Hanjun Dai, Max Lin, and Denny Zhou. Spreadsheetcoder: Formula prediction from semi-structured context. In *International Conference on Machine Learning*, pages 1661–1672. PMLR, 2021.

[55] Yibin Chen, Yifu Yuan, Zeyu Zhang, Yan Zheng, Jinyi Liu, Fei Ni, and Jianye Hao. Sheetagent: A generalist agent for spreadsheet reasoning and manipulation via large language models. In *ICML 2024 Workshop on LLMs and Cognition*, 2024.

[56] Zhimin Chen, Vivek Narasayya, and Surajit Chaudhuri. Fast foreign-key detection in microsoft sql server powerpivot for excel. *Proceedings of the VLDB Endowment*, 7(13):1417–1428, 2014.

[57] Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, et al. Finqa: A dataset of numerical reasoning over financial data. *arXiv preprint arXiv:2109.00122*, 2021.

[58] Zhoujun Cheng, Haoyu Dong, Zhiruo Wang, Ran Jia, Jiaqi Guo, Yan Gao, Shi Han, Jian-Guang Lou, and Dongmei Zhang. Hitab: A hierarchical table dataset for question answering and natural language generation. *arXiv preprint arXiv:2108.06712*, 2021.

[59] Xu Chu, Yeye He, Kaushik Chakrabarti, and Kris Ganjam. Tegra: Table extraction by global record alignment. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data*, pages 1713–1728, 2015.

[60] Xu Chu, Ihab F Ilyas, Sanjay Krishnan, and Jiannan Wang. Data cleaning: Overview and emerging challenges. In *Proceedings of the 2016 international conference on management of data*, pages 2201–2206, 2016.

[61] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[62] Tianji Cong, Madelon Hulsebos, Zhenjie Sun, Paul Groth, and HV Jagadish. Observatory: Characterizing embeddings of relational tables. *arXiv preprint arXiv:2310.07736*, 2023.

[63] Vincenzo Cutrona, Jiaoyan Chen, Vasilis Efthymiou, Oktie Hassanzadeh, Ernesto Jiménez-Ruiz, Juan Sequeda, Kavitha Srinivas, Nora Abdelmageed, Madelon Hulsebos, Daniela Oliveira, et al. Results of semtab 2021. *Proceedings of the semantic web challenge on tabular data to knowledge graph matching*, 3103:1–12, 2022.

[64] Arash Dargahi Nobari and Davood Rafiei. Dtt: An example-driven tabular transformer for joinability by leveraging large language models. *Proceedings of the ACM on Management of Data*, 2(1):1–24, 2024.

[65] Sanjib Das, AnHai Doan, Paul Suganthan G. C., Chaitanya Gokhale, Pradap Konda, Yash Govind, and Derek Paulsen. The magellan data repository. `https://sites.google.com/site/anhaidgroup/projects/data`.

[66] Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. Turl: Table understanding through representation learning. *ACM SIGMOD Record*, 51(1):33–40, 2022.

[67] Jacob Devlin, Jonathan Uesato, Surya Bhupatiraju, Rishabh Singh, Abdel-rahman Mohamed, and Pushmeet Kohli. Robustfill: Neural program learning under noisy i/o. In *International conference on machine learning*, pages 990–998. PMLR, 2017.

[68] Gus Eggert, Kevin Huo, Mike Biven, and Justin Waugh. Tablib: A dataset of 627m tables with context. *arXiv preprint arXiv:2310.07875*, 2023.

[69] Hazem Elmeleegy, Jayant Madhavan, and Alon Halevy. Harvesting relational tables from lists on the web. *Proceedings of the VLDB Endowment*, 2(1):1078–1089, 2009.

[70] Sumit Gulwani. Automating string processing in spreadsheets using input-output examples. *ACM Sigplan Notices*, 46(1):317–330, 2011.

[71] Sumit Gulwani, William R Harris, and Rishabh Singh. Spreadsheet data manipulation using examples. *Communications of the ACM*, 55(8):97–105, 2012.

[72] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[73] Yuxing Han, Ziniu Wu, Peizhi Wu, Rong Zhu, Jingyi Yang, Liang Wei Tan, Kai Zeng, Gao Cong, Yanzhao Qin, Andreas Pfadler, et al. Cardinality estimation in dbms: A comprehensive benchmark evaluation. *arXiv preprint arXiv:2109.05877*, 2021.

[74] Braden Hancock, Hongrae Lee, and Cong Yu. Generating titles for web tables. In *The World Wide Web Conference*, pages 638–647, 2019.

[75] Yeye He, Kris Ganjam, and Xu Chu. Sema-join: joining semantically-related tables using big table corpora. *Proceedings of the VLDB Endowment*, 8(12):1358–1369, 2015.

[76] Yeye He, Xu Chu, Kris Ganjam, Yudian Zheng, Vivek Narasayya, and Surajit Chaudhuri. Transform-data-by-example (tde) an extensible search engine for data transformations. *Proceedings of the VLDB Endowment*, 11(10):1165–1177, 2018.

[77] Yeye He, Kris Ganjam, Kukjin Lee, Yue Wang, Vivek Narasayya, Surajit Chaudhuri, Xu Chu, and Yudian Zheng. Transform-data-by-example (tde) extensible data transformation in excel. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1785–1788, 2018.

[78] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

[79] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

[80] Madelon Hulsebos, Kevin Hu, Michiel Bakker, Emanuel Zgraggen, Arvind Satyanarayan, Tim Kraska, Çagatay Demiralp, and César Hidalgo. Sherlock: A deep learning approach to semantic data type detection. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1500–1508, 2019.

[81] Lan Jiang and Felix Naumann. Holistic primary key and foreign key detection. *Journal of Intelligent Information Systems*, 54(3):439–461, 2020.

[82] Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen, and Kavitha Srinivas. Semtab 2019: Resources to benchmark tabular data to knowledge graph matching systems. In *The Semantic Web: 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31–June 4, 2020, Proceedings 17*, pages 514–530. Springer, 2020.

[83] Zhongjun Jin, Michael R Anderson, Michael Cafarella, and HV Jagadish. Foofah: Transforming data by example. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 683–698, 2017.

[84] Kyoungmin Kim, Jisung Jung, In Seo, Wook-Shin Han, Kangwoo Choi, and Jaehyok Chong. Learned cardinality estimation: An in-depth study. In *Proceedings of the 2022 international conference on management of data*, pages 1214–1227, 2022.

[85] Jan Kossmann, Thorsten Papenbrock, and Felix Naumann. Data dependencies for query optimization: a survey. *The VLDB Journal*, 31(1):1–22, 2022.

[86] Christos Koutras, George Siachamis, Andra Ionescu, Kyriakos Psarakis, Jerry Brons, Marios Fragkoulis, Christoph Lofi, Angela Bonifati, and Asterios Katsifodimos. Valentine: Evaluating matching techniques for dataset discovery. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 468–479. IEEE, 2021.

[87] Eugenie Y Lai, Yeye He, and Surajit Chaudhuri. Auto-prep: Holistic prediction of data preparation steps for self-service business intelligence. *arXiv preprint arXiv:2504.11627*, 2025.

[88] Hai Lan, Zhifeng Bao, and Yuwei Peng. A survey on advancing the dbms query optimizer: Cardinality estimation, cost model, and plan enumeration. *Data Science and Engineering*, 6: 86–101, 2021.

[89] Chia-Hsuan Lee, Oleksandr Polozov, and Matthew Richardson. Kaggledbqa: Realistic evaluation of text-to-sql parsers. *arXiv preprint arXiv:2106.11455*, 2021.

[90] Kristina Lerman, Craig Knoblock, and Steven Minton. Automatic data extraction from lists and tables in web sources. In *IJCAI-2001 Workshop on Adaptive Text Extraction and Mining*, volume 98, 2001.

[91] Hongxin Li, Jingran Su, Yuntao Chen, Qing Li, and ZHAO-XIANG ZHANG. Sheetcopilot: Bringing software productivity to the next level through large language models. *Advances in Neural Information Processing Systems*, 36:4952–4984, 2023.

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

[93] Peng Li, Yeye He, Cong Yan, Yue Wang, and Surajit Chaudhuri. Auto-tables: Synthesizing multi-step transformations to relationalize tables without using examples. *arXiv preprint arXiv:2307.14565*, 2023.

[94] Peng Li, Yeye He, Dror Yashar, Weiwei Cui, Song Ge, Haidong Zhang, Danielle Rifinski Fainman, Dongmei Zhang, and Surajit Chaudhuri. Table-gpt: Table-tuned gpt for diverse table tasks. *arXiv preprint arXiv:2310.09263*, 2023.

[95] Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. Deep entity matching with pre-trained language models. *arXiv preprint arXiv:2004.00584*, 2020.

[96] Yiming Lin, Yeye He, and Surajit Chaudhuri. Auto-bi: Automatically build bi-models leveraging local join prediction and global schema graph. *arXiv preprint arXiv:2306.12515*, 2023.

[97] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[98] Jixiong Liu, Yoan Chabot, Raphaël Troncy, Viet-Phi Huynh, Thomas Labbé, and Pierre Monnin. From tabular data to knowledge graphs: A survey of semantic table interpretation tasks and methods. *Journal of Web Semantics*, 76:100761, 2023.

[99] Zeyao Ma, Bohan Zhang, Jing Zhang, Jifan Yu, Xiaokang Zhang, Xiaohan Zhang, Sijia Luo, Xi Wang, and Jie Tang. Spreadsheetbench: towards challenging real world spreadsheet manipulation. *arXiv preprint arXiv:2406.14991*, 2024.

[100] Jayant Madhavan, Philip A Bernstein, and Erhard Rahm. Generic schema matching with cupid. In *vldb*, volume 1, pages 49–58, 2001.

[101] Mohammad Mahdavi and Ziawasch Abedjan. Baran: Effective error correction via a unified context representation and transfer learning. *Proceedings of the VLDB Endowment*, 13(12): 1948–1961, 2020.

[102] Meredith Ringel Morris, Jascha Sohl-Dickstein, Noah Fiedel, Tris Warkentin, Allan Dafoe, Aleksandra Faust, Clement Farabet, and Shane Legg. Levels of agi for operationalizing progress on the path to agi. *arXiv preprint arXiv:2311.02462*, 2023.

[103] Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. Deep learning for entity matching: A design space exploration. In *Proceedings of the 2018 international conference on management of data*, pages 19–34, 2018.

[104] Phuc Nguyen, Ikuya Yamada, Natthawut Kertkeidkachorn, Ryutaro Ichise, and Hideaki Takeda. Semtab 2021: Tabular data annotation with mtab tool. In *SemTab@ ISWC*, pages 92–101, 2021.

[105] George Papadakis, Ekaterini Ioannou, Emanouil Thanos, and Themis Palpanas. *The four generations of entity resolution*. Springer, 2021.

[106] Marcel Parciak, Sebastiaan Weytjens, Niel Hens, Frank Neven, Liesbet M Peeters, and Stijn Vansummeren. Measuring approximate functional dependencies: A comparative study. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 3505–3518. IEEE, 2024.

[107] Ankur P Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. Totto: A controlled table-to-text generation dataset. *arXiv preprint arXiv:2004.14373*, 2020.

[108] Erhard Rahm and Philip A Bernstein. A survey of approaches to automatic schema matching. *the VLDB Journal*, 10:334–350, 2001.

[109] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

[110] Jonathan Roberts, Kai Han, and Samuel Albanie. Needle threading: Can llms follow threads through near-million-scale haystacks? *arXiv preprint arXiv:2411.05000*, 2024.

[111] Alexandra Rostin, Oliver Albrecht, Jana Bauckmann, Felix Naumann, and Ulf Leser. A machine learning approach to foreign key discovery. In *WebDB*, 2009.

[112] Irina Saparina and Mirella Lapata. Ambrosia: A benchmark for parsing ambiguous questions into database queries. *Advances in Neural Information Processing Systems*, 37:90600–90628, 2024.

[113] Ankita Sharma, Xuanmao Li, Hong Guan, Guoxin Sun, Liang Zhang, Lanjun Wang, Kesheng Wu, Lei Cao, Erkang Zhu, Alexander Sim, et al. Automatic data transformation using large language model-an experimental study on building energy data. In *2023 IEEE International Conference on Big Data (BigData)*, pages 1824–1834. IEEE, 2023.

[114] Ankita Sharma, Jaykumar Tandel, Xuanmao Li, Lanjun Wang, Anna Fariha, Liang Zhang, Syed Arsalan Ahmed Naqvi, Irbaz Bin Riaz, Lei Cao, and Jia Zou. Datamorpher: Automatic data transformation using llm-based zero-shot code generation. In *2025 IEEE 41st International Conference on Data Engineering (ICDE)*, pages 4536–4539. IEEE, 2025.

[115] Roee Shraga and Renée J Miller. Explaining dataset changes for semantic data versioning with explain-da-v. *Proceedings of the VLDB Endowment*, 16(6), 2023.

[116] Rishabh Singh. Blinkfill: Semi-supervised programming by example for syntactic string transformations. *Proceedings of the VLDB Endowment*, 9(10):816–827, 2016.

[117] 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. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

[118] Yoshihiko Suhara, Jinfeng Li, Yuliang Li, Dan Zhang, Çağatay Demiralp, Chen Chen, and Wang-Chiew Tan. Annotating columns with pre-trained language models. In *Proceedings of the 2022 International Conference on Management of Data*, pages 1493–1503, 2022.

[119] Yuan Sui, Mengyu Zhou, Mingjie Zhou, Shi Han, and Dongmei Zhang. Table meets llm: Can large language models understand structured table data? a benchmark and empirical study. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 645–654, 2024.

[120] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[121] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[122] Quoc Trung Tran, Chee-Yong Chan, and Srinivasan Parthasarathy. Query by output. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pages 535–548, 2009.

[123] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

[124] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32, 2019.

[125] Chenglong Wang, Alvin Cheung, and Rastislav Bodik. Synthesizing highly expressive sql queries from input-output examples. In *Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 452–466, 2017.

[126] Pei Wang and Yeye He. Uni-detect: A unified approach to automated error detection in tables. In *Proceedings of the 2019 International Conference on Management of Data*, pages 811–828, 2019.

[127] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

[128] Yue Wang and Yeye He. Synthesizing mapping relationships using table corpus. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1117–1132, 2017.

[129] Ziheng Wei and Sebastian Link. Discovery and ranking of functional dependencies. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 1526–1537. IEEE, 2019.

[130] Tianxing Wu, Shengjia Yan, Zhixin Piao, Liang Xu, Ruiming Wang, and Guilin Qi. Entity linking in web tables with multiple linked knowledge bases. In *Semantic Technology: 6th Joint International Conference, JIST 2016, Singapore, Singapore, November 2-4, 2016, Revised Selected Papers 6*, pages 239–253. Springer, 2016.

[131] Xianjie Wu, Jian Yang, Linzheng Chai, Ge Zhang, Jiaheng Liu, Xeron Du, Di Liang, Daixin Shu, Xianfu Cheng, Tianzhen Sun, et al. Tablebench: A comprehensive and complex benchmark for table question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25497–25506, 2025.

[132] Kai Xiong, Cynthia A Huang, Michael Wybrow, and Yingcai Wu. Tablecanoniser: Interactive grammar-powered transformation of messy, non-relational tables to canonical tables. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pages 1–20, 2025.

[133] Mohamed Yakout, Kris Ganjam, Kaushik Chakrabarti, and Surajit Chaudhuri. Infogather: entity augmentation and attribute discovery by holistic matching with web tables. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 97–108, 2012.

[134] Cong Yan and Yeye He. Synthesizing type-detection logic for rich semantic data types using open-source code. In *Proceedings of the 2018 International Conference on Management of Data*, pages 35–50, 2018.

[135] Junwen Yang, Yeye He, and Surajit Chaudhuri. Auto-pipeline: synthesizing complex data pipelines by-target using reinforcement learning and search. *arXiv preprint arXiv:2106.13861*, 2021.

[136] Xiaoyu Yang and Xiaodan Zhu. Exploring decomposition for table-based fact verification. *arXiv preprint arXiv:2109.11020*, 2021.

[137] Yi Yang, Wen-tau Yih, and Christopher Meek. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 2013–2018, 2015.

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

[139] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9556–9567, 2024.

[140] Dan Zhang, Yoshihiko Suhara, Jinfeng Li, Madelon Hulsebos, Çağatay Demiralp, and Wang-Chiew Tan. Sato: Contextual semantic type detection in tables. *arXiv preprint arXiv:1911.06311*, 2019.

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

[142] Sai Zhang and Yuyin Sun. Automatically synthesizing sql queries from input-output examples. In *2013 28th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 224–234. IEEE, 2013.

[143] Shuo Zhang and Krisztian Balog. Entitables: Smart assistance for entity-focused tables. In *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, pages 255–264, 2017.

[144] Shuo Zhang, Zhuyun Dai, Krisztian Balog, and Jamie Callan. Summarizing and exploring tabular data in conversational search. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1537–1540, 2020.

[145] Yu Zhang, Mei Di, Haozheng Luo, Chenwei Xu, and Richard Tzong-Han Tsai. Smutf: Schema matching using generative tags and hybrid features. *arXiv preprint arXiv:2402.01685*, 2024.

[146] Chen Zhao and Yeye He. Auto-em: End-to-end fuzzy entity-matching using pre-trained deep models and transfer learning. In *The World Wide Web Conference*, pages 2413–2424, 2019.

[147] Danna Zheng, Mirella Lapata, and Jeff Z Pan. Archer: A human-labeled text-to-sql dataset with arithmetic, commonsense and hypothetical reasoning. *arXiv preprint arXiv:2402.12554*, 2024.

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

[149] Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. Agieval: A human-centric benchmark for evaluating foundation models. *arXiv preprint arXiv:2304.06364*, 2023.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: Please see Section 1.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Please see Section 3.3.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [NA]

Justification: [NA]

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Please see Section 4.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: Please see Appendix A.1.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Please see Section 4.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Please see Table 3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Please see Section 4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We have reviewed the Code of Ethics and believe the research conducted in the paper conforms to the Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: Please see Section 3.3.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Please see Appendix A.2.

    Guidelines:

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

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [NA]

    Justification: [NA]

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: no crowd-sourcing or research with human subjects was conducted.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: [NA]

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [NA]

Justification: [NA]

Guidelines:

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

A Broader Discussions

A.1 Open access to data and code

Our benchmark data is available at `https://huggingface.co/MMTU-benchmark`, and our evaluation code is available at `https://github.com/MMTU-Benchmark`.

A.2 License for existing assets

We provide comprehensive references to the sources of our data assets throughout this paper, with more details in [15]. We respect and conform to the licensing terms of existing datasets, whenever we can find such terms when we create our benchmark.

B Task overview

B.1 Column transformation

Column transformation refers to the category of tasks, where new columns in a table are derived using existing columns. We select three key tasks in this category:

Program Transform by Example (PTBE) [76, 70, 67, 116, 77]. This is a popular task studied in the data management and programming language literature. In this task, we are given a few example output values in a new output column (usually provided by users as demonstrations), and the task is to synthesize a transformation program using input tables, such that the produced output can match the user-given output examples. Figure 1 shows an example of this task. We use the benchmark datasets from [76, 70] for this task in MMTU.

Semantic Transform by example (STBE) [75, 35, 64]. This task is similar to PFBE above, in that users also provide a few example output to demonstrate the intended transformations, but the target transformation requires "semantic" transformations (e.g., country to capital-city, or company to stock-ticker), that cannot be programmed using a syntactic program like in PFBE. We use the datasets from [75, 35]for our benchmark.

Formula by Context (FBC) [52, 54]. This task is studied in the context of spreadsheets, where given a target spreadsheet cell in which users want to enter a spreadsheet formula, we are asked to predict the intended formula in the target cell. We use the 4 benchmark datasets in [52].

B.2 Table Transformation

In the table transformation task category, we also need to perform transformations, but unlike "Column transformations" above where only new columns are derived (without changing the shape of the input table), in "Table transformation", a broader class of transformations can be invoked (e.g., table reshaping, restructuring, group-by, aggregation, etc.), which can change the form and shape of the original input table.

Table Transform by Output Table (TTBT) [40, 125, 122, 142]. In this task, we are given a pair of input/output tables, and the task is to infer a transformation program (in SQL or Pandas), which when executed on the input table, can generate the desired output table. Figure 1 shows an example of this task. We use the datasets from [40, 125].

Table Transform by Output Schema (TTBS) [135, 113, 114]. This task is similar to TTBT above, except that the output table is a schematic depiction of how the desired output table should look like, without being the exact output that the desired transformation program would generate on the given input (as it is sometimes hard to generate such output table without first producing the desired program). We use the dataset in [135]for this task.

Table Transform by Relationalization (TTBR) [93, 39, 83, 132, 87]. Because relational data analysis often requires input tables to be in a relational form, this task transforms data in non-relational semi-structured forms, into the standard relational form. The task is to predict such a relationalization transformation program, based on the characteristics of the input table. We use the dataset in [93].

B.3 Table matching

In this task category, we try to match relevant rows and columns between multiple tables.

Entity Matching (EM) [65, 103, 146, 95, 105]. Entity matching is the task of determining whether rows or entities from two tables correspond to the same real-world entity. We use the datasets in [65, 103]for benchmarking.

Schema Matching (SM) [86, 145, 41, 100]. Schema matching tries to match relevant columns from two tables that map to the same semantic concept. Figure 1 shows an example of this task. We use datasets in [86, 145]for MMTU.

Header Value Matching (HVM) [94, 108]. In HVM, we are given a table without headers, and a shuffled list of the original headers in the table, where the task is to match column headers with column values in tables. We use the dataset in [94]for benchmarking.

B.4 Data cleaning

Tasks in the data cleaning category tries to improve the quality of input tables, which are popular tasks studied in the data management literature.

Data Imputation (DI) [45, 68, 94]. Data imputation is the intuitive task of predicting missing values in a relational table, based on the surrounding context of the table. Figure 1 shows an example of this task. We use the dataset from [45, 68].

Error Detection (ED) [60, 126, 101, 50]. The task of Error detection aims to identify erroneous values in a table that are semantically inconsistent or anomalous with the rest of the column. We use the dataset in [50].

List to Tables (L2T) [59, 46, 69, 90]. In List-to-table, the task is to segment records of data without clear separators, into columns, so that values in the same column are homogeneous, and the resulting table becomes a natural relational table, with values consistently aligned in homogenous columns. We use the dataset from [59].

B.5 Table join

Join is an important operation that connects multiple related tables together.

Equal Join (EJ) [96, 56, 111, 81]. In Equal-join, we are given a collection of related tables, and the task is to identify all join relationships between the tables. Figure 1 shows an example of this task. We use the dataset in [96].

Semantic Join (SJ) [75, 35, 64, 128]. In Semantic Join, we are also tasked to join related tables together, but instead of the common equi-join, which is based on string-equality comparisons, the join relationship is based on semantic relatedness (e.g., country and capital city, or company and stock ticket). We use the same semantic-transformation datasets from [75, 35], but converting the input/output transformation columns, as join keys from two separate tables.

B.6 Column relationships

In this category of tasks, the goal is to identify implicit but semantically meaningful relationships from an input table.

Arithmetic Relationship (AR) [115, 1]. This task focuses on identifying arithmetic relationships from an underlying table. Figure 1 shows an example of this task.

String Relationship (SR) [1, 76, 71]. This task focuses on identifying string transformation relationships from a given table.

Functional Relationship (FR) [1, 106, 129]. This task focuses on identifying functional-relationships from input tables. We use the datasets from [1] our as benchmarks for all three tasks.

B.7 Table understanding

Tasks in this category are intended to test models ability to understand tables, and retrieve relevant facts from tables.

Needle-in-a-haystack-table (NIHT). In this task, we create a variant of the popular "Needle-in-a-haystack" task in the context of tables, like described in detail in Appendix E. We use the dataset from [68]to construct tests in this task.

Needle-in-a-haystack-index (NIHI). In this task, we reverse the NIHT task above, and ask models to identify index positions corresponding to a given value in a table. We use the same dataset from [68]to construct tests for this task.

B.8 NL-2-Code

NL-2-SQL (NS) [147, 92, 89, 138, 148]. NL-2-SQL is a popular task to translate natural-language questions into SQL statements that can execute given an input table. We use the benchmarks from [147, 92, 89, 138, 148]in MMTU.

B.9 Table Question Answering (Table QA)

Table-QA (TQA) [131, 137, 57, 58]. Table QA is another popular task, where models are used to directly answer questions on a given input table, without code execution. We use benchmarks from [53, 131, 137, 57]for this task.

Fact verification (FV) [53, 136, 141]. Table fact verification is a variant of TQA, in which models are asked if a statement is refuted or supported by facts presented in an input table. We use data from [53]for this task.

B.10 Knowledge-base mapping (KB Mapping)

KB mapping is the task where we map facts and relationships of a table, to known KB ontologies.

Column type annotation (CTA) [82, 134, 118, 140, 80]. This is a popular task where columns in a table is mapped to known entity types in a knowledge-base. We use the CTA dataset from [82].

Column property annotation (CPA) [82, 63, 104]. [82]. The CPA task is to predict the relationship of a given pair of columns, based on known properties in a knowledge-base. We use the CPA dataset from [82].

Cell entity annotation (CEA) [82, 66, 42, 130, 98]. The CEA task predicts the knowledge-base entity id of a given cell in a table. We use the CEA dataset from [82].

C Additional Statistics

Table 4 lists the exact ranges of token count calculated for MMTU questions, in each quartile and for each task.

D Detailed model performance at the dataset level

Table 5 shows a detailed breakdown of performance results on MMTU, at the dataset level. Reasoning models indeed outperform chat models in many cases, though chat models are better on knowledge-centric tasks like CTA, where we find reasoning models have a tendency to hallucinate (e.g., type names that do not exist in knowledge bases).

Task	0–25% range	25–50% range	50–75% range	75–100% range
Arithmetic-Relationship	1150.0–3547.0	3547.0–6834.0	6834.0–15601.5	15601.5–126654.0
Cell-entity-annotation	263.0–2138.5	2138.5–3706.0	3706.0–4693.5	4693.5–7526.0
Column-type-annotation	274.0–576.8	576.8–1346.5	1346.5–3094.8	3094.8–6744.0
Columns-property-anotation	261.0–612.8	612.8–1416.0	1416.0–3004.2	3004.2–6754.0
Data-Imputation	214.0–628.2	628.2–1320.0	1320.0–1971.0	1971.0–102929.0
Data-transform-pbe	262.0–309.8	309.8–333.5	333.5–412.2	412.2–21846.0
Data-transform-reshape	534.0–916.5	916.5–1692.0	1692.0–3621.0	3621.0–76096.0
Entity-Matching	175.0–207.0	207.0–232.0	232.0–248.5	248.5–425.0
Error-Detect	211.0–235.0	235.0–293.0	293.0–638.5	638.5–21040.0
Formula-prediction-context	269.0–1096.2	1096.2–1315.0	1315.0–1564.0	1564.0–10332.0
Functional-Dependency	1341.0–3641.0	3641.0–5157.0	5157.0–7573.0	7573.0–76083.0
List-to-table	184.0–283.5	283.5–352.0	352.0–514.5	514.5–2955.0
NL2SQL	418.0–872.0	872.0–1245.0	1245.0–3293.0	3293.0–35250.0
Schema-Matching	547.0–1803.5	1803.5–2378.0	2378.0–3737.5	3737.5–10562.0
String-Relationship	2961.0–7596.2	7596.2–14127.5	14127.5–26032.5	26032.5–92595.0
Table-Fact-Verification	269.0–451.0	451.0–584.0	584.0–836.5	836.5–2959.0
Table-Locate-by-Row-Col	11190.0–21158.0	21158.0–22433.0	22433.0–23731.0	23731.0–72074.0
Table-QA	229.0–381.0	381.0–537.0	537.0–847.2	847.2–13752.0
Table-needle-in-a-haystack	11123.0–21286.0	21286.0–22737.0	22737.0–23664.0	23664.0–72009.0
Transform-by-input-output-table	200.0–259.0	259.0–297.0	297.0–366.0	366.0–728.0
Transform-by-output-target-schema	360.0–1095.5	1095.5–1827.5	1827.5–2845.8	2845.8–113819.0
equi-join-detect	477.0–2471.0	2471.0–3972.0	3972.0–6741.0	6741.0–107690.0
header-value-matching	186.0–260.0	260.0–338.5	338.5–489.2	489.2–1102.0
semantic-join	243.0–367.5	367.5–742.0	742.0–1860.5	1860.5–19184.0
semantic-transform	209.0–247.0	247.0–265.0	265.0–285.0	285.0–454.0

Table 4: Exact ranges of token count calculated for MMTU questions, in each quartile and for each task, as used in long-table-context analysis (Figure 6). The number of tokens is calculated using GPT-4o tokenizer.

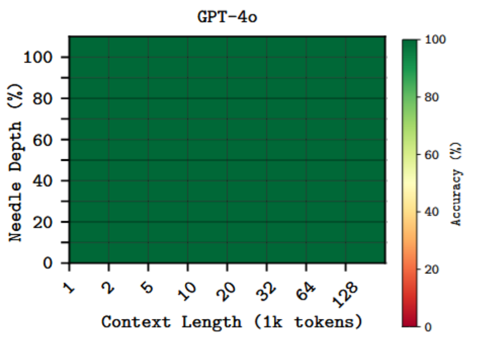


Figure 10: The needle-in-a-haystack test used in NLP context. Each cell in the heatmap corresponds to a test to retrieve a simple fact planted in a document, for a given document length and retrieval depth. Frontier models like GPT-4o and Gemini-2.5 can now typically achieve perfect accuracy, as shown by the all-green heatmap (results from [110]).

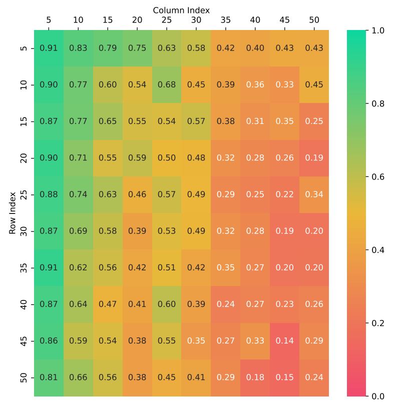


Figure 11: Our table-based needle-in-a-haystack test, where frontier models like GPT-4o continue to struggle to retrieve simple facts from large tables, especially with more columns (notice the asymmetry in the heatmap, where an increased column-index makes the task significantly harder).

E Long-context table understanding: A case study of "needle in a haystack in tables"

In this section, we take a closer look at one simple task in MMTU that we refer to as "needle in a haystack in tables", to more deeply analyze the problem of table understanding with long table context (large tables with many rows and columns).

Recall that "Needle-in-a-haystack (NIH)" [3, 110], is a popular test traditionally used to evaluate long-context LLM's ability to retrieve a simple fact (a needle) from a long document context (e.g., 100K or 1M tokens). Recent advances in frontier long-context LLMs have made this task almost irrelevant, as most frontier models such as GPT-4o, Llama-4, and Gemini-2.5 can now score perfectly on NIH [97, 120, 26, 14], like the heatmap in Figure 10 would show. Here the all-green heatmap

Task Name	Dataset	Chat Model		Reasoning Model	
		GPT-5-Chat	DeepSeek-V3	GPT-5	Deepseek-R1
Data-transform-reshape	Auto-Tables	0.500	0.282	**0.710**	0.126
Transform-by-output-target-schema	commercial-pipelines	0.231	0.154	**0.385**	0.154
	github-pipelines	0.253	0.224	**0.269**	0.213
	Average	0.242	0.189	**0.327**	0.183
Transform-by-input-output-table	AutoPandas	0.593	0.481	**0.889**	0.852
	Scythe	0.667	0.600	**0.950**	0.883
	Average	0.630	0.541	**0.919**	0.868
Entity-Matching	Amazon-Google	0.872	0.856	**0.938**	0.868
	BeerAdvo-RateBeer	0.966	0.955	**0.989**	0.943
	DBLP-ACM	0.983	0.985	**0.997**	0.973
	DBLP-Scholar	0.962	0.962	**0.976**	0.950
	Fodors-Zagats	**1.000**	0.995	**1.000**	0.989
	Walmart-Amazon	0.948	0.940	**0.983**	0.940
	iTunes-Amazon	0.953	0.943	**0.991**	0.915
	Average	0.955	0.948	**0.982**	0.940
header-value-matching	TableGPT	0.699	0.633	**0.818**	0.678
Schema-Matching	DeepMDatasets	**1.000**	**1.000**	**1.000**	**1.000**
	HXD	0.928	**0.945**	0.937	0.937
	Wikidata	0.897	0.882	0.912	**0.931**
	assays	0.843	0.913	0.887	**0.924**
	miller2	0.916	**0.946**	0.925	0.929
	prospect	0.976	0.927	**0.996**	0.958
	Average	0.927	0.935	0.943	**0.947**
Data-Imputation	WebTable	0.493	0.483	**0.642**	0.499
	tablib	0.704	0.627	**0.843**	0.664
	Average	0.599	0.555	**0.742**	0.582
Error-Detect	Relational-Tables	0.136	**0.214**	0.211	0.153
	Spreadsheet-Tables	0.103	0.123	**0.128**	0.113
	Average	0.119	0.168	**0.170**	0.133
List-to-table	TEGRA	**0.604**	0.600	0.598	0.567
semantic-join	DataXFormer	0.797	0.759	0.834	**0.840**
	SEMA-join	0.899	0.889	**0.947**	0.928
	Average	0.848	0.824	**0.890**	0.884
equi-join-detect	Auto-BI	0.665	0.655	**0.692**	0.688
Data-transform-pbe	TDE	0.487	0.419	**0.708**	0.614
	Transformation-text	0.515	0.479	**0.681**	0.636
	Average	0.501	0.449	**0.694**	0.625
Formula-prediction-context	cisco-random	0.124	0.101	**0.224**	0.166
	enron-random	0.051	0.043	**0.163**	0.105
	pge-random	0.071	0.080	**0.116**	0.125
	ti-random	0.076	0.051	**0.179**	0.140
	Average	0.081	0.069	**0.170**	0.134
semantic-transform	DataXFormer	0.423	0.419	**0.507**	0.443
	SEMA-join	0.877	0.865	**0.915**	0.883
	Average	0.650	0.642	**0.711**	0.663
Arithmetic-Relationship	Auto-Relate	0.476	0.506	**0.735**	0.693
Functional-Dependency	Auto-Relate	0.739	0.730	**0.799**	0.762
String-Relationship	Auto-Relate	0.148	0.128	0.388	**0.551**
Table-needle-in-a-haystack	MMTU	0.546	0.446	**0.920**	0.703
Table-Locate-by-Row-Col	MMTU	0.570	0.517	**0.941**	0.710
NL2SQL	Archer	0.327	0.183	0.394	**0.404**
	KaggleDBQA	**0.535**	0.465	0.524	0.481
	Spider	**0.785**	0.776	0.739	0.707
	WikiSQL	**0.745**	0.733	0.732	0.704
	bird	0.463	0.429	**0.478**	0.459
	Average	0.571	0.517	**0.574**	0.551
Table-QA	FinQA	0.211	0.229	**0.497**	0.403
	TableBench	0.441	0.432	**0.580**	0.566
	WikiQA	0.728	0.685	**0.827**	0.797
	Average	0.460	0.449	**0.635**	0.589
Table-Fact-Verification	TabFact	0.852	0.838	**0.940**	0.927
Column-type-annotation	SemTab2019	**0.554**	0.409	0.462	0.371
Columns-property-anotation	SemTab2019	0.296	0.258	**0.492**	0.280
Cell-entity-annotation	SemTab2019	0.687	0.676	**0.803**	0.689

Table 5: Dataset-level Performance of Frontier Chat & Reasoning Models

indicates that this particular LLM under test, GPT-4o, can perfectly retrieve a needle planted at varying depth (y-axis), within documents of varying lengths (x-axis), with 100% accuracy.

Super-Categories	Percentage
Reasoning & Coding Errors	40.3%
Table-Understanding Errors	25.6%
Knowledge Errors	16.4%
Other Errors	11.7%

Table 6: Distribution of Super-category Errors On Incorrect Questions by o4-mini

Super Categories	Sub Categories	Percentage
Table-Understanding Errors	Row/Column index misalignment	0.283122
Reasoning & Coding Errors	Incorrect reasoning	0.223264
Reasoning & Coding Errors	Semantically incorrect code	0.138167
Knowledge Errors	Incorrect knowledge	0.105523
Knowledge Errors	Hallucinated facts	0.0334465
Other Errors	Format error / Result-extraction failure	0.0323357
Other Errors	Ground-truth ambiguity / quality	0.0323357
Table-Understanding Errors	Complex table	0.0156742
Table-Understanding Errors	Multi-table understanding	0.0153039
Knowledge Errors	Numerical Errors	0.014008

Table 7: Distribution of Top-10 Sub-category Errors On Incorrect Questions by o4-mini.

We adapt NIH to the table setting, and study the table-version of the "needle in a haystack", which we call "needle-in-a-haystack-in-table" (NIHT), that tests LLM's ability to retrieve a simple fact (needle) within a cell of a table. Specifically, in NIHT, we randomly sample 100 real tables with at least 50 rows and columns, and ask LLMs to retrieve a simple fact randomly planted at row-i and column-j of each table, repeated over all (i, j) positions. LLM responses at different (i, j) positions can then be compared with the ground-truth (the cell value at those positions) to measure accuracy.

We would like to highlight that NIHT is the simplest possible task, for table understanding in long table context, as it performs a simple retrieval with no additional steps. Any real table tasks with long table context (e.g., Table QA on large tables) will necessarily be at least as hard as NIHT, as those tasks will need to first retrieve/identify relevant cells in long table context, before further processing (reasoning, calculation or coding) can be performed, making NIHT a good case study for long-context table understanding.

The results of NIHT are illustrated also using a heatmap in Figure 11. As we can see, compared to Figure 10, today's frontier LLMs still face substantial challenges in correctly identifying a cell value within a two-dimensional table, like indicated by the red cells in the heatmap. This is especially true when we increase the number of columns (e.g., with more than 25 columns, LLM accuracy quickly falls below 0.5). Furthermore, we observe a *strong asymmetry* in the heatmap – e.g., with 50 rows and 5 columns we still have a reasonable accuracy of 0.81 (at the lower-left corner of the heatmap), but with 5 rows and 50 columns the accuracy drops to a lowly 0.43 (at the top-right corner), underscoring the challenge LLMs face when reading "vertically" in the column direction, which are consistent with our intuition that LLMs pre-trained on one-dimensional natural-language texts are less effective in reading two-dimensional tables "vertically", like was also observed in [94, 119].

F LLM-based Error analysis

Setup. We conducted a LLM-based error analysis to understand the underlying reasons of these errors on MMTU. We built a detailed categorization of typical errors that we observe in models' answers, using the following 4 super-categories, and 14 sub-categories:

- **Super-category: Table-Understanding Errors**
  - Row/Column index misalignment (e.g., the row/column index is not correctly recognized in a table, leading to incorrect model output)
  - Long context (e.g., long table causes model to produce incorrect results)

  - Multi-table understanding (e.g., join relationships between multiple tables are identified incorrectly, leading to incorrect answers)
  - Complex table (e.g., merged cells spanning multi-row/cols, hierarchical headers, etc., causing models to produce incorrect answers)
- **Super-category: Reasoning & Coding Errors**
  - Syntax or execution error (if the task asks for SQL/Python code, and the code should encounter an execution error)
  - Semantically incorrect code (if the task asks for SQL/Python code, and the code is executable but the result is incorrect)
  - Incorrect reasoning (pick this category only if the model's reasoning shows obvious flaws that directly lead to incorrect answers)
- **Super-category: Knowledge Errors**
  - Hallucinated facts (e.g., in CEA, output knowledge base references that do not exist)
  - Incorrect knowledge (e.g., in data imputation, filled in a related but incorrect value)
  - Numerical Errors (e.g., in data imputation, filled a numerical value that is close but not precise)
- **Super-category: Other Errors**
  - Format error / Result-extraction failure
  - Timeout
  - Ground-truth ambiguity / quality
  - None of the above: create a new category, and give a short description

Result. We used o4-mini to categorize all 11K questions for which o4-mini were incorrect on MMTU into these categories. The distribution of super-categories is shown in Table 6, note that the percentage does not sum to one due to format issue and LLM curated super-categories. The distribution of sub-categories is shown in Table 7. We observe that reasoning & coding, and table understanding remain the two largest categories of errors.