arXiv:2505.19563v3 [cs.AI] 8 Jan 2026

TabularMath: Understanding Math Reasoning over Tables with Large Language Models

Shi-Yu Tian[1,2*], **Zhi Zhou**[1*], **Wei Dong**[2*], **Kun-Yang Yu**[1,2], **Ming Yang**[1,2], **Zi-Jian Cheng**[3], **Lan-Zhe Guo**[1,3†], **Yu-Feng Li**[1,2†]
[1]National Key Laboratory for Novel Software Technology, Nanjing University
[2]School of Artificial Intelligence, Nanjing University
[3]School of Intelligence Science and Technology, Nanjing University
{tiansy,zhouz,guolz,liyf}@lamda.nju.edu.cn

Abstract

Mathematical reasoning has long been a key benchmark for evaluating large language models. Although substantial progress has been made on math word problems, the need for reasoning over tabular data in real-world applications has been overlooked. For instance, applications such as business intelligence demand not only multi-step numerical reasoning with tables but also robustness to incomplete or inconsistent information. However, comprehensive evaluation in this area is severely limited, constrained by the reliance on manually collected tables that are difficult to scale and the lack of coverage for potential traps encountered in real-world scenarios. To address this problem, we propose AUTOT2T, a neuro-symbolic framework that controllably transforms math word problems into scalable and verified tabular reasoning tasks. Building on this pipeline, we develop TabularMath, a benchmark comprising four subsets that include both text-based and image-based tables, covering table complexity, table quality, and table representation dimensions. Our study reveals three key observations: (1) Table complexity and reasoning difficulty impact reasoning performance jointly; (2) Low-quality tables pose severe risks to reliable reasoning in current LLMs; (3) Different table modalities show similar trends, with text-based tables typically being easier for models to reason over. In-depth analyses are conducted for each observation to guide future research.

1 Introduction

Mathematical reasoning has long been a critical benchmark for evaluating the capabilities of large language models (LLMs). The field has advanced remarkably in recent years (OpenAI, 2023; Guo et al., 2025), with many single-scenario benchmarks now considered largely solved (Hosseini et al., 2014; Patel et al., 2021; Cobbe et al., 2021).

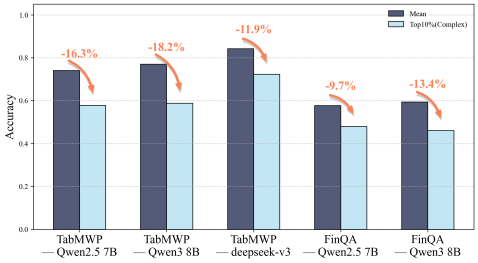


Figure 1: Model performance comparasion between average questions and top10% complex questions.

This progress has prompted a shift in research focus toward real-world applications, particularly reasoning over semi-structured data like tables (Lu et al., 2023). Unlike plain text, tables present information in a highly structured and organized format, making them indispensable in domains such as business intelligence (Zhang et al., 2024) and financial forecasting (Zhu et al., 2021).

Nevertheless, real-world table reasoning scenarios present significant challenges for LLMs. For example, in the financial sector, the need to process large-scale tables continues to grow with the increasing volume and complexity of data, alongside stricter requirements for reliability and security (Bradley et al., 2024; Zavitsanos et al., 2024). In quarterly financial reports, models are expected not only to perform cross-column computations on numerous metrics like revenue, profit, and liabilities but also to verify numerical consistency (e.g., ensuring total assets equal the sum of liabilities and equity). Failure to properly interpret the data or detect inconsistencies can lead to severe consequences in downstream applications like investment decisions and risk assessment (Cerchiello and Giudici, 2016).

Despite prior works (Zhu et al., 2021; Chen et al., 2021; Lu et al., 2023) addressing some aspects of tabular mathematical reasoning, these efforts have been limited in table scale and primarily focused on the accuracy of perfectly crafted, text-only problems. Specifically, existing tabular benchmarks

*Equal contribution.
†Corresponding author.

largely rely on manual annotation and collection, making it difficult to scale the datasets effectively. As a result, these benchmarks fail to explore the limits of LLMs' reasoning capabilities on more complex tables, where the models perform worse (as shown in Fig 1). Then, current benchmarks have not adequately assessed tabular mathematical reasoning in a comprehensive manner, with limited evaluation of both robustness to incomplete or inconsistent data and reasoning over image-based tables. Therefore, to systematically assess model capabilities across multiple dimensions, a comprehensive benchmark is crucial. In this context, there is an urgent need for a more complete and systematic evaluation framework to thoroughly explore and challenge the boundaries of existing models.

To address the above limitations, we propose an **Auto**matic **T**ext-**t**o-**T**able generation framework, AUTOT2T. It is a neuro-symbolic pipeline that converts math word problems into scalable and verified tabular reasoning tasks without human annotation, enabling the evaluation of both accuracy and robustness. To facilitate standardized evaluation and fair comparison, we construct a comprehensive tabular mathematical reasoning benchmark, **TabularMath**, based on AUTOT2T. It comprises three subsets of increasing difficulty (*Easy, Medium, Hard*) and an *Imperfect* subset that includes both text-based and image-based tables, covering table complexity, table quality and table representation dimensions. Based on this, we conduct systematic experiments and analyses on 18 open-source and proprietary models. The results are organized around the following three research questions and lead to several key observations.

1. ***How does table complexity affect mathematical reasoning?*** **Table complexity and reasoning difficulty impact reasoning performance jointly.** Nearly all models suffer significant performance drops when transitioning from pure text to tabular modalities, with degradation increasing as table complexity grows. The coupling between retrieval and reasoning forms a core bottleneck, with pure retrieval being substantially easier than joint reasoning and retrieval (performance gap exceeding 20% on average).

2. ***How does table quality affect mathematical reasoning?*** **Low-quality tables pose severe risks to reliable reasoning in current LLMs.**. When tables contain missing or contradictory information, most models fail to identify these flaws and produce misleading answers, with error rates exceeding 50% in some cases. Moreover, when models are informed that inputs may contain imperfect expression, they experience performance degradation on well-defined problems, demonstrating a trade-off between solvency and discriminative ability.

3. ***How does table representation affect mathematical reasoning?*** **Different table modalities show similar trends, with text-based tables typically being easier for models to reason over.** Across models and difficulty levels, image- and text-based tables show similar trends, while even multimodal models achieve comparable or higher accuracy on text-based tables. Among textual formats, key-value structured formats such as JSON and serialization perform better.

Overall, we conduct a systematic and in-depth analysis of tabular mathematical reasoning from the perspectives of table complexity, table quality, and table representation, complemented by additional discussions. This work represents an exploratory step toward multimodal reasoning over structured data, laying the groundwork for addressing these challenges in future research.

2 Related work

Math Reasoning. Mathematical reasoning serves as a key benchmark for evaluating the capabilities of large language models (LLMs) due to its verifiable nature. Early progress was made on elementary-level math problems using datasets such as GSM8K (Cobbe et al., 2021), MultiArith (Koncel-Kedziorski et al., 2016), and SVAMP (Patel et al., 2021), where methods like in-context learning (Wei et al., 2022; Gao et al., 2023), supervised fine-tuning (Li et al., 2024b), and reinforcement learning (Guo et al., 2025) demonstrated strong performance. Since then, researchers have questioned the accuracy of current assessments of large models' mathematical reasoning, exploring approaches such as neural-symbolic methods (Mirzadeh et al., 2024). A growing area of interest is the robustness of mathematical reasoning (Zhou et al., 2024; Shi et al., 2023), specifically, whether models can refrain from generating hallucinations when faced with incomplete or logically

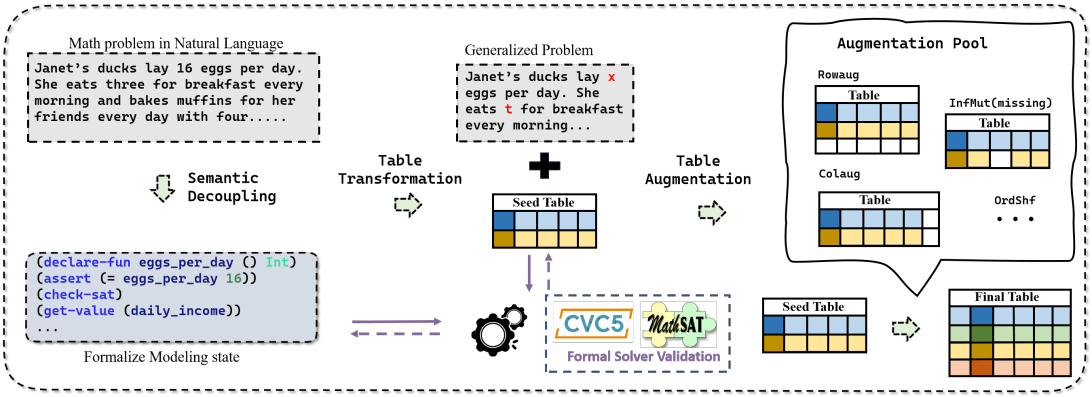


Figure 2: An overview of AUTOT2T pipeline.

deceptive prompts (Tian et al., 2025; Zhao et al., 2024).

Table Question Answering. Table Question Answering(Table QA) has significant practical applications across various domains, including financial statement analysis (Chen et al., 2021) and medical diagnosis (Hasny et al., 2025). The field has advanced considerably with the development of high-quality datasets, beginning with the pioneering work of Pasupat et al. (Pasupat and Liang, 2015), who constructed the WikiTableQuestions (WTQ) dataset using Wikipedia tables. Subsequent research shifted to more complex QA tasks requiring reasoning capabilities, exemplified by datasets such as ToTTo (Parikh et al., 2020)(focused on answer generation) and OTTQA (Chen et al., 2020) (emphasizing cross-table reasoning). More recently, FinQA (Chen et al., 2021) and AiTQA (Katsis et al., 2021) have explored numerical reasoning in tables, while TableBench (Wu et al., 2025) and Text2Analysis (He et al., 2024) introduced multimodal approaches incorporating visual elements. However, most existing datasets rely on manual annotation, lacking an automated pipeline for scalable data processing.

3 Automated Text to Table

The AUTOT2T pipeline converts math word problems into tabular problems through the following three stages (Fig 2).

3.1 Semantic Decoupling

Firstly, our objective is to semantically decouple the text of the math word problems and extract key elements that can be structurally represented. We decompose math problems using formal language modeling(Such as SMT-Lib (Barrett et al., 2010; Li et al., 2024a)), structuring problems as:

$$\begin{aligned} \textit{Goal} \quad & g := \text{solve } f(v) \\ \textit{Constraints} \quad & c := e_1(v) \bowtie e_2(v), \\ & \quad \bowtie \in \{\geq, \leq, >, <, =, \neq\} \\ \textit{Expressions} \quad & e := h \mid e_1 \oplus e_2, \\ & \quad \oplus \in \{+, -, \times, \div\} \\ \textit{Domains} \quad & \mathcal{D} := \mathbb{N} \mid \mathbb{N}^+ \mid \mathbb{R} \end{aligned}$$

where v is a variable, c is a constraint, e is an expression, h is a constant, and f is the objective function. For a problem p, we define the modeling state as $\mathcal{S} = (\mathcal{V}, \mathcal{C})$, where $\mathcal{V}$ and $\mathcal{C}$ denote the variable and constraint sets, respectively. The LLM constructs $\mathcal{S}$ by extracting candidate components from the problem description, while a formal solver Φ (e.g., Z3 (de Moura and Bjørner, 2008), CVC5 (Barbosa et al., 2022)) verifies satisfiability and consistency, providing feedback for refinement and identifying ill-defined formulations.

3.2 Table Transformation

After obtaining the formal modeling state, the next step is to transform the semantically decoupled components into a structured tabular representation. Specifically, we convert the formal state into a table by introducing a `name` field as the primary key and mapping variables $\mathcal{V}$ and active constraints $\mathcal{C}_a$ to table columns. Given a problem p, the LLM produces a blurred textual description $\hat{p}$ together with a two-row seed table t_{seed}:

$$(\hat{p}, t_{seed}) = LLM_{tt}(p, \mathcal{V}, \mathcal{C}_a) \tag{1}$$

To ensure the correctness of entity extraction, we validate the generated table using a formal solver with updated constraints $\mathcal{C}_{\hat{a}}$:

$$\hat{R}_{valid} = \Phi\left(\mathcal{V}, (\mathcal{C} \setminus \mathcal{C}_a) \cup \mathcal{C}_{\hat{a}}\right) \tag{2}$$

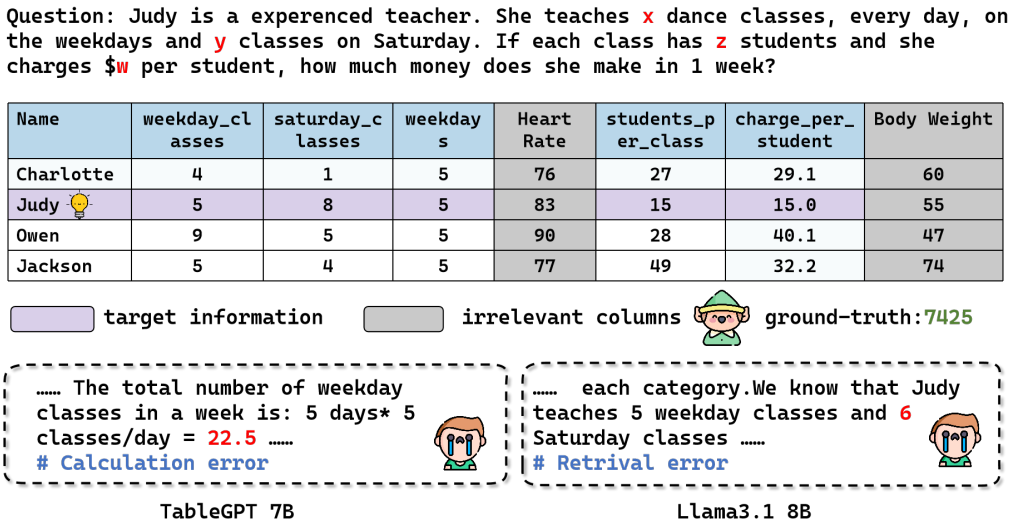


Figure 3: Illustrative cases in TabularMath and corresponding model responses

3.3 Table Augmentation

Starting from the initial seed table t_{seed}, we construct an augmentation pool $\mathcal{A}$ that applies randomized operations to expand and perturb table structures. The pool consists of four strategies: *Row Augmentation (RowAug)*, *Column Augmentation (ColAug)*, *Order Shuffling (OrdShf)*, and *Information Modification (InfMut)*. Based on user-specified selections, these strategies are iteratively applied to generate diverse table variants.

$$t_i = \begin{cases} t_{seed} & \text{if } i = 0 \\ Aug_j(t_{i-1}), \quad Aug_j(\cdot) \in \mathcal{A} & \text{if } i > 0 \end{cases} \tag{3}$$

The AUTOT2T enables automated generation of diverse tabular problem variants without manual annotation. Details on augmentation strategies are in Appendix A.2.1.

4 TabularMath Benchmark

4.1 Benckmark Details

To enable standardized evaluation and fair comparison, we construct a predefined benchmark dataset, TabularMath, through the AUTOT2T pipeline. We have converted a benchmark, consisting of four subsets: *Easy, Medium, Hard, Imperfect*, covering a total of 3,391 unique tables. For each table, we further augment the data by rendering it into an image format using the matplotlib library in Python and storing the corresponding visual representation. As a result, the final benchmark contains 6,782 samples. Table 2 summarizes the key statistics of each subset as well as the augmentation strategies employed, while additional examples from TabularMath and visual comparisons with GSM8K are provided in Appendix A.2. We will describe our benchmark subsets separately.

***Easy, Medium, Hard* Subset** For these three subsets, we employ the AUTOT2T to systematically reorganize and embed the information originally implicit in the textual descriptions into tabular representations, thereby transforming explicit textual semantics into structured tables. Each subset contains 797 examples (with the remaining samples excluded due to failures in table conversion or consistency validation), and all examples are derived from the same set of seed questions to ensure semantic consistency and comparability across subsets. The progression from easy to hard reflects an increasing level of table complexity, as measured by the number of rows or columns (induced via *RowAug, ColAug and OrdShf*). This growth in structural complexity substantially increases the difficulty of information localization and retrieval within the tables.

***Imperfect* Subset** Real-world tables are not always perfect and may contain errors or omissions. In the *Imperfect* subset, we simulated such a real-world open environment, which falls into two categories: **missing-type** (removing essential information from the target line of the table) and **contra-type** (injecting intermediate variables required for the question but designed to conflict with existing information). This subset contains 1,000 instances in total, evenly split between the two types (500 per type). These traps are introduced via *Information Modification* during table augmentation. During evaluation, we inform the model that the ta-

Table 1: Comparison between TabularMath and existing datasets

Dataset	Key Statistic		Modality		Content Coverage			Construction Process	
	Test size	Table cells	Text	Image	Information Retrieval	Math Reasoning	Robustness	Automated Verification	Symbolizable
AddSub (Hosseini et al., 2014)	600	NA	✓	✗	✗	✓	✗	✗	✗
SVAMP (Patel et al., 2021)	1000	NA	✓	✗	✗	✓	✗	✗	✗
GSM8k (Cobbe et al., 2021)	1438	NA	✓	✗	✗	✓	✗	✗	✗
PMC (Tian et al., 2025)	5374	NA	✓	✗	✗	✓	✓	✓	✗
Tabfact (Chen et al., 2019)	1695	15.1	✓	✗	✓	✗	✗	✗	✗
FinQA (Chen et al., 2021)	1147	24.5	✓	✗	✓	✓	✗	✗	✗
TaT-QA (Zhu et al., 2021)	669	37.6	✓	✗	✓	✓	✗	✗	✗
TabMWP (Lu et al., 2023)	7686	11.8	✓	✓	✓	✓	✗	✗	✗
TabularGSM	6782	93.5	✓	✓	✓	✓	✓	✓	✓

Table 2: Key statistics in TabularMath

Statistic	Easy	Medium	Hard	Imperfect
Total questions	797	797	797	1000
Table cells	41	82	162	90
Table Rows	4.1	4.1	8.1	4.5
Table Columns	10	20	20	20
Question Length	232.2	232.2	232.2	237
RowAug	✓	✓	✓	✓
OrdShf	✗	✓	✓	✓
ColAug	✗	✗	✓	✗
InfMut	✗	✗	✗	✓

ble might contain omissions, and observe whether models can accurately identify these traps and abstain from answering, which serves as the core metric. We measure performance by reporting the proportion of ill-defined questions that a model successfully rejects.

4.2 Comparison with Existing Benchmarks

As shown in Table 1, TabularMath differs from existing ones in three key aspects: (1) Tables in TabularMath are more complex, containing more cells, which makes it harder to retrieve useful information; (2) Compared to prior mathematical reasoning datasets, our benchmark jointly evaluates reasoning and retrieval abilities, and includes an imperfect subset with incomplete information. Unlike existing tabular QA datasets, we emphasize mathematical reasoning and assess models' ability to detect traps (e.g., flawed or contradictory conditions), promoting robust and safe reasoning. (3) In terms of construction, unlike prior work that relies heavily on manual annotation, we adopt a neuro-symbolic approach AUTOT2T, which rewrites textual problems into tabular form. This allows us to generate multiple table variants for the same seed problem, achieving efficient and controllable data creation.

Table 3: Main results on TabularMath benchmark

Dataset	GSM8k	Easy	Medium	Hard	Avg
Open-source General Models					
Qwen3 14B	**94.54**	77.87	70.21	61.59	**69.89**
Qwen3 8B	**93.30**	73.18	55.63	47.30	**58.70**
Qwen3 4B	**91.79**	71.57	52.68	41.73	**55.32**
Qwen3 1.7B	**81.25**	50.76	30.62	18.84	**33.40**
Qwen2.5 14B	**93.40**	79.21	64.10	49.09	**64.13**
Qwen2.5 14B coder	**90.68**	72.63	57.74	45.61	**58.66**
Qwen2.5 7B	**82.86**	53.92	34.45	20.64	**36.33**
Qwen2.5 7B coder	**84.71**	64.78	42.13	23.52	**43.47**
Qwen2.5 3B	**80.28**	39.74	23.96	15.68	**26.46**
LLaMA3.1 8B	**83.69**	48.61	33.37	32.15	**38.04**
LLaMA3 8B	**55.34**	36.30	21.12	20.68	**26.03**
Open-source Math Models					
Qwen-Math 7B	**95.45**	53.69	30.37	14.59	**32.88**
DeepSeek-Math 7B	**80.13**	12.81	6.60	2.04	**7.15**
Open-source Tabular Models					
TableGPT 7B	**24.33**	30.60	16.44	17.64	**21.56**
StructLM 7B	**32.97**	14.78	8.28	4.44	**9.17**
Closed-source APIs					
DeepSeek v3	**96.36**	88.63	87.63	85.83	**87.37**
GLM-4-Plus	**95.07**	84.52	81.03	78.27	**81.27**
GPT-4	**94.46**	85.54	78.42	75.23	**79.73**

5 Experiments and Results

In this section, we conduct a series of experiments on 18 models spanning four categories to analyze their performance on mathematical reasoning over structured data systematically. We primarily investigate the following three research questions: (1)How does table complexity affect mathematical reasoning? (2)How does table quality affect mathematical reasoning? (3)How does table representation affect mathematical reasoning?

5.1 RQ1. How does table complexity affect mathematical reasoning?

Performance drops when moving from pure text to tabular modalities, and increases in ta-

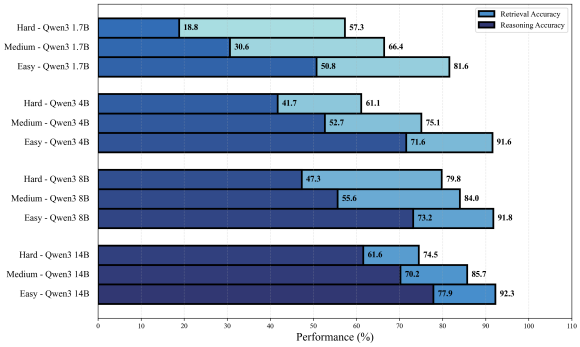


Figure 4: Performance comparison of table reasoning and single-step table retrieval

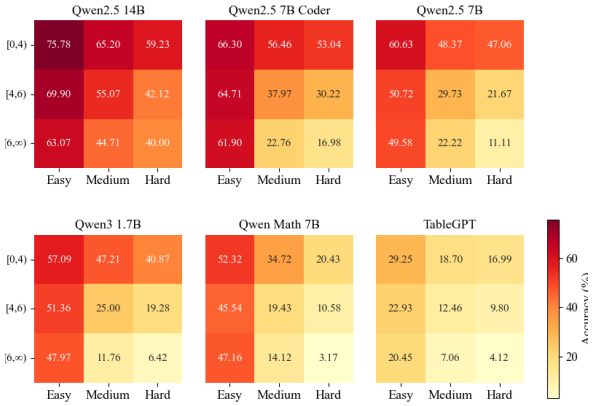


Figure 5: Accuracy heatmap of table complexity and reasoning difficulty in tabular math reasoning

ble complexity further exacerbate this degradation. As shown in Table 3, when transitioning from the original GSM8K dataset to our constructed TabularMath, all models exhibit consistent and significant performance degradation. Moreover, this degradation becomes more pronounced as table complexity increases. Smaller models, whose reasoning capabilities are inherently limited, suffer more severe performance drops, while domain-specific models (including math-specialized and table-specialized models) also struggle to generalize under this setting. We further conduct manual analysis on a subset of model outputs and identify four primary error types: retrieval omission (failing to recognize the need for retrieval), retrieval mismatch (retrieving incorrect table values), expression errors (incorrect formulation of the target equation), and numerical calculation errors (correct formulas but incorrect computations). Among these, retrieval mismatch is the dominant source of errors. Detailed case studies are provided in Appendix A.4.

Is the degradation in model inference performance due to the inability to find information? **Pure retrieval is substantially easier than joint reasoning and retrieval.** Pure retrieval refers to replacing end-to-end reasoning with a simplified single-step retrieval task. Specifically, given a question such as `"Janet's ducks lay x eggs per day ..."`, we directly ask `"How many eggs do Janet's ducks lay per day?"`, which corresponds to the `"eggs-per-day"` key in the table.As shown in Fig. 4, model performance on full reasoning tasks is significantly worse than on pure retrieval tasks. This indicates that when the retrieval target is explicitly specified, models can reliably perform retrieval. However, once retrieval is embedded within multi-step reasoning, performance drops by an average of 20%. This reveals a substantial gap between retrieval and reasoning over tables.

Moreover, we categorize all test samples in TabularMath for each model along two dimensions: retrieval difficulty (i.e., table complexity, divided into Easy, Medium, and Hard subsets) and reasoning difficulty (measured by the number of variables). This results in nine categories, whose accuracies are visualized as a heatmap in Fig. 5. The results indicate that model performance is strongly correlated with both factors, and the impacts of retrieval difficulty and reasoning difficulty are nearly comparable.

5.2 RQ2. How does table quality affect mathematical reasoning?

The quality of tables is not always guaranteed to be intact; information may be missing or contradictory. This research question aims to investigate whether, under such circumstances, models can timely detect these issues and inform users that the question is ill-posed or problematic, rather than providing a misleading answer.

Flawed table quality could pose significant risks for current LLMs. As shown in 4, when tables contain missing or contradictory information, most LLMs fail to properly identify these flaws and instead produce misleading answers. The high Missing and Contradictory scores indicate that models frequently generate responses even when critical information is absent or conflicting, rather than alerting users to the problematic nature of the questions. This behavior is particularly dangerous in real-world applications where users may rely on these answers without recognizing the underlying data quality issues. Contradictory cases prove more challenging than missing-information cases, suggesting that models struggle more when informa-

Table 4: Performance on Imperfect subset

Dataset	Missing	Contradictory	Average
Open-source General Models			
Qwen3 14B	67.03	28.21	**47.62**
Qwen3 8B	68.57	34.54	**51.56**
Qwen3 4B	69.75	32.12	**50.94**
Qwen3 1.7B	37.33	14.40	**29.56**
Qwen2.5 14B	20.00	6.80	**13.40**
Qwen2.5 14B Coder	51.60	23.60	**37.60**
Qwen2.5 7B	34.80	16.00	**25.40**
Qwen2.5 7B Coder	34.00	20.80	**27.40**
Qwen2.5 3B	79.20	69.60	**74.40**
LLaMA3.1 8B	9.60	10.40	**10.00**
LLaMA3 8B	35.20	19.20	**21.90**
Open-source Math Models			
Qwen-Math 7B	48.93	20.80	**34.87**
DeepSeek-Math 7B	53.60	60.40	**57.00**
Open-source Tabular Models			
TableGPT 7B	46.40	23.20	**34.80**
StructLM 7B	0.00	0.00	**0.00**
Closed-source APIs			
DeepSeek v3	82.80	68.00	**75.40**
GLM-4-plus	65.60	27.60	**46.60**
GPT-4	80.20	21.11	**50.66**

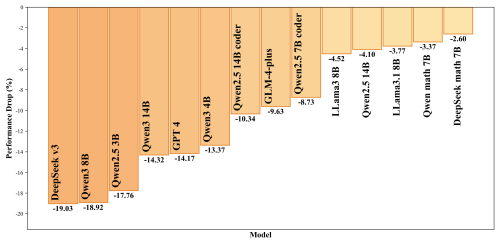


Figure 6: Performance degradation on well-defined table problems caused by table-checking instructions

tion conflicts rather than when it is simply absent. Even advanced proprietary APIs exhibit significant vulnerabilities, with average error rates exceeding 50% in some cases. Notably, the seemingly higher robustness of some smaller models mainly stems from their coarse-grained refusal behavior, as they tend to reject most queries rather than genuinely reasoning about flawed tables.

What impact does informing the model of potential issues with table formatting have on solving normal well-defined problems? **All models experienced varying degrees of performance degradation, demonstrating the trade-off between solvency and discriminative ability.** When models are explicitly informed that inputs may contain trap problems, they must first assess problem solvability before solution planning. This discriminative requirement forces models to allocate resources to verification, altering the reasoning process. As shown in Table 6, this mixed setting leads to performance degradation on well-defined problems across all model types, including advanced proprietary APIs. The trade-off between solvency and discriminative ability indicates that current models struggle to simultaneously maintain high accuracy on solvable problems while effectively detecting problematic inputs.

Table 5: Performance comparison with and without visual table representations on TabularMath

Model	with image				w/o
	Easy	Medium	Hard	Avg	Avg
Qwen2.5-VL-7B	57.26	49.36	42.70	**49.44**	45.51
Qwen2.5-VL-3B	29.65	23.36	18.26	23.76	**26.47**
Qwen3-VL-8B	66.25	64.14	59.83	63.41	**66.40**
Qwen3-VL-4B	66.17	60.22	51.22	59.20	**60.09**
InternVL3-8B	15.69	14.20	10.02	**13.30**	10.82
InternVL3-14B	52.60	34.03	20.13	35.59	**38.66**

5.3 RQ3. How does table representation affect mathematical reasoning?

Given that real-world scenarios frequently involve diverse forms of image-based tables, a natural and important question arises as to how reasoning over image-based tables compares to reasoning over text-based tables across different models

Image-based and text-based tables exhibit similar performance trends, but image-based tables are more challenging to reason over. As shown in Table 5, across different models and difficulty levels, image-based tables follow performance trends largely consistent with those of text-based tables, with accuracy decreasing as problem difficulty increases. However, even for multimodal models, accuracy on text-based tables is comparable to or higher than that on image-based tables, with gains of 0.89–3.07 percentage points. These results suggest that image-based tables do not necessarily provide more usable information for mathematical reasoning, likely due to OCR noise and layout ambiguity, whereas text-based representations offer more explicit structure and numerical precision.

For text-based tables, does the choice of table format affect model performance across different difficulty levels? **JSON and Serialized formats consistently outperform Markdown, with the gap widening as table complexity increases.** We evaluate three widely used text-based table formats representations: serialized, Markdown, and JSON,

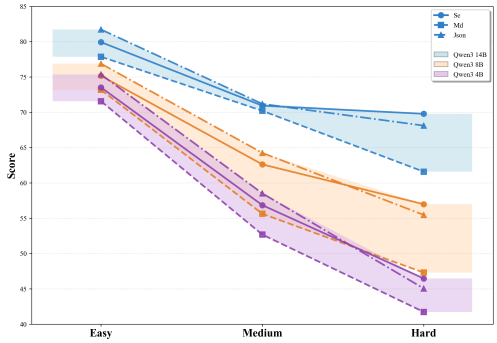


Figure 7: Performance comparison across different table formats (Se, Md, Json) and difficulty levels

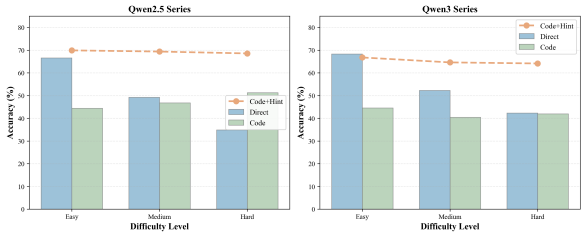


Figure 8: Performance comparison between code-based reasoning and direct reasoning.

and conduct a detailed comparison of their performance (Explanation in Appendix). As shown in Fig. 7, JSON and serialized representations achieve comparable performance and consistently outperform Markdown across all difficulty levels(Full results in table 14). Moreover, the performance gap becomes more pronounced as problem complexity increases. We attribute this trend to the explicit key–value structure in JSON and serialized formats, where each row is associated with clear keys, facilitating more reliable retrieval and reasoning, whereas Markdown introduces additional parsing ambiguity that degrades performance on complex tables.

5.4 Further Discussion

Beyond the above questions, we further explore several complementary aspects, including code-based solving (Fig 8), fine-tuning performance (Table 6), row/column ablations (Fig 9), and trap-type ablations (Table 13). Due to space limitations, some analyses are deferred to the appendix.

Can using code enhance the model's inference ability? **Models often struggle to accurately infer what information should be retrieved or executed during the reasoning process.** This limitation constrains the potential benefits of code-based reasoning. We conduct systematic experiments on the Qwen2.5 and Qwen3 series of models, and visualize in Fig 8 (Full Results in Table 15). The results show that incorporating code leads to more stable performance overall; however, in many cases, it does not outperform direct natural language reasoning. In contrast, when the prompt explicitly specifies which pieces of information should be retrieved and used for reasoning, model performance improves substantially. These findings suggest that the performance gains stem not from the mere inclusion of code, but from providing the model with a clear and actionable information retrieval and reasoning trajectory.

Table 6: Performance changes on TabularMath of the basic model after AUTOT2T training (In-Domain)

Settings	Easy (Md)	Medium (Md)	Hard (Md)
Qwen2.5-3B-Instrct	39.74	23.96	15.68
Qwen2.5-3B-Instrct + finetune	53.58	41.78	36.51
Δ	↑ 13.84	↑ 17.82	↑ 20.83
Qwen2.5-7B-Instrct	53.92	34.45	20.64
Qwen2.5-7B-Instrct + finetune	62.10	51.32	36.39
Δ	↑ 8.18	↑ 16.87	↑ 15.75

How about using* AUTOT2T*-generated data for training? **A flexible data generation framework can provide substantial support for improving tabular mathematical reasoning.** To improve tabular mathematical reasoning, we leverage AUTOT2T to generate targeted training data. Starting from the GSM8K training set, we construct approximately 6K augmented samples and fine-tune Qwen2.5-7B and Qwen2.5-3B. As shown in Table 6, fine-tuning yields an average improvement of about 15% on TabularMath. These gains further transfer to other tabular reasoning benchmarks (TAT-QA, FinQA, and TabMWP), with an average improvement of 4%(Table 12), particularly on complex tables, demonstrating the effectiveness of controllable data generation.

6 Conclusion

In this work, we introduce TabularMath, a comprehensive benchmark for systematically studying tabular mathematical reasoning in LLMs. It comprises three subsets of increasing complexity and an imperfect subset to evaluate both performance and robustness. We further propose a neural–symbolic framework, AUTOT2T, which controllably transforms word problems into scalable and validated tabular reasoning tasks. Extensive experiments on 18 models reveal key challenges in tabular reasoning from the perspectives of table complexity, table quality, and table representation, and offer insights for future research.

Limitations. One limitation of our work is that the difficulty profile of a benchmark is inherently influenced by the complexity of the initial mathematical application problems from which it is constructed. While the pipeline can be systematically expanded by varying table size, structure, and information distribution, the intrinsic reasoning depth of the resulting tasks remains partially dependent on the underlying seed problems. However, we note that real-world table-based reasoning scenarios rarely involve highly esoteric or abstract chains of mathematical reasoning. In practice, most applications emphasize accurate retrieval, alignment, and moderately multi-step computations over structured data. From this perspective, the difficulty level of the initial problems is sufficient to support a meaningful and realistic assessment of tabular mathematical reasoning.

References

Haniel Barbosa, Clark Barrett, Martin Brain, Gereon Kremer, Hanna Lachnitt, Makai Mann, Abdalrhman Mohamed, Mudathir Mohamed, Aina Niemetz, Andres Nötzli, and 1 others. 2022. cvc5: A versatile and industrial-strength smt solver. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 415–442. Springer.

Clark Barrett, Aaron Stump, Cesare Tinelli, and 1 others. 2010. The smt-lib standard: Version 2.0. In *Proceedings of the 8th international workshop on satisfiability modulo theories*, volume 13, page 14.

Ethan Bradley, Muhammad Roman, Karen Rafferty, and Barry Devereux. 2024. Synfintabs: a dataset of synthetic financial tables for information and table extraction. *arXiv preprint arXiv:2412.04262*.

Paola Cerchiello and Paolo Giudici. 2016. Big data analysis for financial risk management. *Journal of Big Data*, 3(1):18.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Wang, and William W Cohen. 2020. Open question answering over tables and text. *arXiv preprint arXiv:2010.10439*.

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

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and 1 others. 2021. Finqa: A dataset of numerical reasoning over financial data. *arXiv preprint arXiv:2109.00122*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Leonardo Mendonça de Moura and Nikolaj S. Bjørner. 2008. Z3: an efficient SMT solver. In *Procddings of the 14th Tools and Algorithms for the Construction and Analysis of Systems International Conference*, volume 4963 of *Lecture Notes in Computer Science*, pages 337–340.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In *Proceedings of the 40th International Conference on Machine Learning*, pages 10764–10799.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Juanzi Li, and 37 others. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *Preprint*, arXiv:2406.12793.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Marta Hasny, Maxime Di Folco, Keno Bressem, and Julia Schnabel. 2025. Tgv: Tabular data-guided learning of visual cardiac representations. *arXiv preprint arXiv:2503.14998*.

Xinyi He, Mengyu Zhou, Xinrun Xu, Xiaojun Ma, Rui Ding, Lun Du, Yan Gao, Ran Jia, Xu Chen, Shi Han, and 1 others. 2024. Text2analysis: A benchmark of table question answering with advanced data analysis and unclear queries. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18206–18215.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 523–533.

Yannis Katsis, Saneem Chemmengath, Vishwajeet Kumar, Samarth Bharadwaj, Mustafa Canim, Michael Glass, Alfio Gliozzo, Feifei Pan, Jaydeep Sen, Karthik Sankaranarayanan, and 1 others. 2021.

Ait-qa: Question answering dataset over complex tables in the airline industry. *arXiv preprint arXiv:2106.12944*.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. Mawps: A math word problem repository. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics*, pages 1152–1157.

Zenan Li, Zhi Zhou, Yuan Yao, Yu-Feng Li, Chun Cao, Fan Yang, Xian Zhang, and Xiaoxing Ma. 2024a. Neuro-symbolic data generation for math reasoning. *arXiv preprint arXiv:2412.04857*.

Zenan Li, Zhi Zhou, Yuan Yao, Xian Zhang, Yu-Feng Li, Chun Cao, Fan Yang, and Xiaoxing Ma. 2024b. Neuro-symbolic data generation for math reasoning. In *Advances in Neural Information Processing Systems*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. 2023. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. In *The Eleventh International Conference on Learning Representations*.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2024. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv preprint arXiv:2410.05229*.

OpenAI. 2023. Gpt-4. Technical report.

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

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. *arXiv preprint arXiv:1508.00305*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems? *arXiv preprint arXiv:2103.07191*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. *arXiv preprint arXiv:2302.00093*.

Shi-Yu Tian, Zhi Zhou, Kun-Yang Yu, Ming Yang, Lin-Han Jia, Lan-Zhe Guo, and Yu-Feng Li. 2025. Vcsearch: Bridging the gap between well-defined and ill-defined problems in mathematical reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 12721–12742.

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

Xianjie Wu, Jian Yang, Linzheng Chai, Ge Zhang, Jiaheng Liu, Xeron Du, Di Liang, Daixin Shu, Xianfu Cheng, Tianzhen Sun, and 1 others. 2025. Tablebench: A comprehensive and complex benchmark for table question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25497–25506.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024a. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, and 1 others. 2024b. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Elias Zavitsanos, Dimitris Mavroeidis, Eirini Spyropoulou, Manos Fergadiotis, and Georgios Paliouras. 2024. Entrant: A large financial dataset for table understanding. *Scientific Data*, 11(1):876.

Liangyu Zha, Junlin Zhou, Liyao Li, Rui Wang, Qingyi Huang, Saisai Yang, Jing Yuan, Changbao Su, Xiang Li, Aofeng Su, and 1 others. 2023. Tablegpt: Towards unifying tables, nature language and commands into one gpt. *arXiv preprint arXiv:2307.08674*.

Xiaokang Zhang, Sijia Luo, Bohan Zhang, Zeyao Ma, Jing Zhang, Yang Li, Guanlin Li, Zijun Yao, Kangli Xu, Jinchang Zhou, and 1 others. 2024. Tablellm: Enabling tabular data manipulation by llms in real office usage scenarios. *arXiv preprint arXiv:2403.19318*.

Jun Zhao, Jingqi Tong, Yurong Mou, Ming Zhang, Qi Zhang, and Xuan-Jing Huang. 2024. Exploring the compositional deficiency of large language models in mathematical reasoning through trap problems. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 16361–16376.

Zihao Zhou, Qiufeng Wang, Mingyu Jin, Jie Yao, Jianan Ye, Wei Liu, Wei Wang, Xiaowei Huang, and Kaizhu Huang. 2024. Mathattack: Attacking large language models towards math solving ability. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19750–19758.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. Tat-qa: A question answering benchmark on a hybrid of tabular and textual content in finance. *arXiv preprint arXiv:2105.07624*.

Alex Zhuang, Ge Zhang, Tianyu Zheng, Xinrun Du, Junjie Wang, Weiming Ren, Stephen W Huang, Jie Fu, Xiang Yue, and Wenhu Chen. 2024. Structlm: Towards building generalist models for structured knowledge grounding. *arXiv preprint arXiv:2402.16671*.

A Appendix

A.1 Use of LLMs

This work investigates the boundaries of large language models (LLMs) in tabular mathematical reasoning. In this process, LLMs serve a dual role. On the one hand, they act as both the subject of study and a tool for dataset construction and experimental evaluation, supporting data generation (Sec. 3) and benchmarking representative models (Sec. 4). On the other hand, LLMs are additionally employed to polish the writing and enhance the clarity of English expression.

All key research ideas, theoretical analysis, experimental design, and writing of the main body of the paper were independently completed by the authors. We did not use the large language model to generate the scientific content of the manuscript, nor did we contribute to the formulation of the research hypotheses or the interpretation of the findings. The authors bear full responsibility for the accuracy, originality, and completeness of all content in the paper.

A.2 Details of TableGsm8k Dataset

To evaluate the reasoning ability of the models on structured data, we construct the TabularMath benchmark including four subsets: *Easy, Medium, Hard,* and *Imperfect*.The *Imperfect* subset contains 50% solvable problems (corresponding to medium difficulty) and 50% unsolvable problems (25% with contradictory conditions + 25% with missing information).

Taking the following problem as an example (Example 1), we use the four categories mentioned in 3.1.3 to generate five tables(Table 7 8 9 10 11) based on the seed row and corresponding generalized problem, so that the table and generalized problem form a question pair as our dataset.

- **Easy:** Apply RowAug (10 times) and Shuffling to the seed row
- **Medium:** Apply RowAug (20 times), Shuffling to the seed row
- **Hard:** Apply RowAug (20 times), Shuffling, and ColAug (4 columns) (adding irrelevant information marked with gray)
- **Imperfect:** Apply RowAug (20 times), Shuffling, and InfMod(two situations) to the seed row
  - **Contra:** Apply Contradictory Condition Modification, adding row *"eggs_for_sale"*, which is an implicit variable that can be obtained from the formula "*eggs_for_sale = eggs_per_day - eggs_eaten - eggs_for_muffins*". Modify this implicit variable (original value is marked with blue) to create conflicts with existing constraints, making the problem unsolvable.
  - **Missing:** Apply Missing Condition Modification, removing a key data (marked with yellow)(set as null) from seed row.

Example 1

Original Problem: Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?

Generalized Problem: Janet's ducks lay x eggs per day. She eats y for breakfast every morning and bakes muffins for her friends every day with z. She sells the remainder at the farmers' market daily for $w per fresh duck egg. How much in dollars does she make every day at the farmers' market?

Seed Row: (red word in table)

Name	Eggs per Day	Eggs Eaten	Eggs for Muffins	Price per Egg
Janet	16	3	4	2

A.2.1 Detaild explanation of augmentations

The detaild explanation of augmentations used in $\mathcal{A}$ are as follows.

- **Row Augmentation(RowAug)**: Select an existing row as a seed and modify its column data to simulate information of different individuals (e.g., changing names, adjusting numerical values). As augmented rows only serve to expand the dataset size without affecting the original problem's solvability, no additional validation of numerical rationality is required.
- **Column Augmentation(ColAug)**: A new column is added to the existing table. Since each row is constructed based on the protagonist of the mathematical problem, column augmentation enriches the description of the entities by adding information such as "height", "blood pressure", and other attributes.
- **Order Shuffling(OrdShf)**: Randomly shuffle row or column sequences to increase the difficulty of data retrieval.
- **Information Modification(InfMod)**: This strategy affects solvability in two ways:
  - **Missing Condition Modification**: Remove one or more key data points from seed rows (set as null), rendering the original problem unsolvable due to insufficient conditions.
  - **Contradictory Condition Modification**: Modify implicit variables (values not explicitly stated but derivable from given conditions) to create conflicts with existing constraints, making the problem unsolvable due to logical contradictions.

Table 7: Easy Table

Name	Eggs per Day	Eggs Eaten	Eggs for Muffins	Price per Egg
Sebastian	72	8	16	7
Sofia	73	1	17	7
Elijah	5	4	14	10
Mia	73	9	19	7
Ava	46	0	7	5
Samuel	3	8	6	7
Logan	47	0	9	7
Henry	95	9	9	6
Janet	16	3	4	2
Ella	65	8	15	4
Elizabeth	54	0	2	4

A.3 Experiment Details

A.3.1 Setup

Models. We evaluated four major categories of LLMs within TabularMath, including open-source general-purpose models (e.g., the Qwen series (Yang et al., 2024a)(including Qwen 3 and Qwen 2.5) and Llama3 series (Grattafiori et al., 2024)), open-source math-specialized models (DeepSeek-Math (Shao et al., 2024) and Qwen-Math (Yang et al., 2024b)), open-source table-specialized models (TableGPT (Zha et al., 2023) and StructLM (Zhuang et al., 2024)), and proprietary API models (GPT-4 (OpenAI, 2023), DeepSeek-v3 (Liu et al., 2024), and GLM-4-plus (GLM et al., 2024)).

Formats. We evaluate three widely used ways of organizing tables: *serialized format Markdown format* and *json format*. In the serialized format, each table row is converted into key-value pairs (e.g., `"name: Janet, Eggs_per_day: 16, Eggs_eat_morning: 4..."`). In the Markdown format, the table is presented using standard Markdown syntax, with the first row as column headers and subsequent rows listing values in order, using the `"|"` symbol as the column delimiter. In the *json format*, Each row of the table will be organized into a format similar to JSON key-value pairs for the large model. Details prompts can be found in the appendix or the code section.

Computing Resources. We use NVIDIA A100 servers as our primary computing platform, along with a few additional machines equipped with RTX 4090 GPUs.

A.3.2 Additional results

AUTOT2T-generated data improves performance on other tabular reasoning datasets. We evaluate on three other tabular math reasoning datasets on Qwen-2.5-7B model, namely TAT-QA (Zhu et al., 2021), FinQA (Chen et al., 2021), and TabMWP (Lu et al., 2023), which primarily test mathematical reasoning over tables. Under the same number of training steps, we further compare two settings: (i) training only on the target dataset's official training set (*Pure-finetune*), and (ii) training on a mixed dataset that combines the target dataset with data generated by AUTOT2T (*Mix-finetune*). As shown in Table 12, the *Mix-finetune* consistently outperforms the *Pure-finetune* setting. The improvement is particularly pronounced on more complex tables, which highlights the versatility and generalization ability of our AUTOT2T across diverse datasets.

Model performance degrades with increased retrieval difficulty. First, we want to explore the relationship between performance degradation and table complexity. Through experiements in main text, we get an initial observation that model performance decreases monotonically as table complexity increases—from easy to hard levels. To further investigate the underlying mechanisms, we conducted a supplementary analysis based on two data augmentation strategies: *ColAug* and *RowAug*. We generated a series of augmented tables by fixing either the number of rows or columns and varying the other, to examine how model performance responds to changes in table structure. As shown in Figure 9, while the inference performance fluctuates as the number of columns (*ColAug*) or rows

Table 8: Medium Table

Price/Egg	For Muffins	Name	Eaten	Eggs/Day
1	18	Jacob	1	17
8	7	Sebastian	0	20
9	12	Lillian	3	59
6	14	Aiden	1	80
7	20	Joseph	2	11
7	4	James	9	20
9	16	Grace	4	36
10	17	Mia	6	90
8	8	Oliver	3	43
6	13	Charlotte	0	18
9	3	Mia	3	79
10	15	Mason	1	34
1	1	Jacob	9	85
4	8	Lucas	8	95
3	13	Liam	8	56
10	5	James	8	84
1	0	Oliver	5	47
6	19	Eleanor	5	40
3	10	Victoria	9	68
5	5	Samuel	5	16
2	4	Janet	3	16

(*RowAug*) increases, a clear downward trend is evident. We attribute this degradation to the increased presence of irrelevant or distracting information, which raises the difficulty of information retrieval and subsequently impairs the model's reasoning ability.

Traps within the reasoning process make the model more prone to hallucinations. Which types of traps are more challenging for the model to detect? To explore this question, we introduce an additional set of experiments by incorporating two types of adversarial scenarios: Direct Missing (DM) and Direct Contra (DC). Unlike the trap questions involved in TabularMath, these traps are more explicit and thus easier to detect for humans. In the DM questions, the table lacks the "name" attribute required by the question, which means the name of the target person mentioned in the question and their corresponding information are entirely absent from the table. In the DC questions, the table contains two columns with the same header (i.e., duplicate column names) but with conflicting values. These conflicting entries can lead to different answers depending on which value is used. We present our experimental results in Table 13. For missing traps, the model exhibits a significantly higher success rate in identifying direct traps compared to indirect ones. In contrast, performance on contra traps remains consistently poor across models, with some degree of variability. These results indicate that traps embedded within the reasoning process are inherently more difficult to detect.

A.3.3 Detailed results

We provide comprehensive experimental results here that are not given in the main text due to space limitations. Table 14 is the complete version of Table 3 and Table 4. Table 16 corresponds to Figure 4 , which shows the performance comparison of table reasoning and single-step table retrieval. Table 17 18 19 and 20 correspond to Figure 9, which shows the relationship between model performance changes and table complexity.

Table 9: Hard Table

Age	Heart Rate	Eggs/Day	Price/Egg	Name	Eaten	For Muffins	Body Temp	Sleep Hours
23	81	75	1	Emma	3	1	38	4
46	79	81	5	Chloe	7	0	38	9
43	88	10	3	Emma	2	14	39	6
42	73	41	9	Madison	3	11	40	10
51	87	98	3	Eleanor	0	4	40	9
50	97	94	6	Olivia	1	7	37	10
63	67	93	3	Lily	5	16	40	8
38	70	51	5	David	5	11	39	6
70	87	19	10	Isabella	3	17	40	7
64	99	11	1	Avery	8	9	38	10
72	67	81	7	Emily	1	20	38	4
57	69	38	6	Ella	7	16	36	4
25	62	94	10	John	3	11	39	5
71	91	29	5	Camila	6	7	38	4
42	73	62	9	Layla	7	17	36	8
62	96	32	7	Harper	2	19	38	6
36	78	77	8	Olivia	6	3	39	9
48	85	7	7	Aiden	8	10	38	8
60	82	20	5	Joseph	9	19	38	6
30	94	77	7	Logan	2	18	40	7
25	72	16	2	Janet	3	4	39	5

Table 10: Table with Contradictory Conditions

eggs_per_day	eggs_for_sale (real_eggs_for_sale)	eggs_eaten	name	eggs_for_muffins	price_per_egg
65	10 (38)	7	Noah	20	4
87	9 (65)	5	Wyatt	17	1
95	13 (83)	0	Jayden	12	1
47	13 (27)	8	Lucas	12	4
34	15 (18)	9	Ethan	7	7
72	13 (53)	7	Liam	12	2
79	8 (53)	10	Sofia	16	4
12	7 (-1)	5	Lily	8	9
58	13 (45)	5	Sophia	8	1
31	12 (28)	0	Jayden	3	5
90	12 (78)	10	Ava	2	3
86	16 (58)	8	Sophia	20	10
45	14 (42)	1	Amelia	2	8
44	16 (37)	7	Victoria	0	10
84	10 (64)	7	Mason	13	9
16	12 (7)	3	Janet	4	2
74	7 (60)	7	Oliver	7	10
43	15 (31)	3	Aiden	9	8
82	16 (70)	7	Michael	5	5
57	16 (45)	0	Riley	12	7
52	12 (31)	5	Henry	16	8

Table 11: Table with Missing Information

eggs_per_day	eggs_eaten	name	price_per_egg	eggs_for_muffins
66	2	Riley	8	10
97	2	Hannah	3	0
70	3	Olivia	3	20
51	8	Charlotte	6	9
79	0	Elizabeth	2	13
16	null	Janet	2	4
14	10	Ava	8	3
48	4	Ethan	3	14
73	7	Olivia	3	20
32	0	Chloe	3	14
41	8	James	3	0
1	1	Benjamin	3	4
8	0	Sophia	3	13
20	6	Victoria	8	14
93	10	John	9	8
62	0	Penelope	10	10
21	2	Harper	6	5
17	1	Oliver	10	10
60	3	John	4	4
14	0	David	9	3
76	0	Jayden	1	7

Table 12: Performance of fine-tuned models on different tabular reasoning benchmarks

Setting	TabMWP		FinQA		TAT-QA	
	Average	Top10%	Average	Top10%	Average	Top10%
Baseline	74.09	74.07	57.72	48.00	37.34	30.77
Pure-finetune	91.60	91.90	74.80	71.19	67.38	51.28
Mix-finetune	94.70	98.11	77.59	77.97	69.53	56.41
Δ	↑ 3.10	↑ 6.21	↑ 2.79	↑ 6.78	↑ 2.15	↑ 5.13

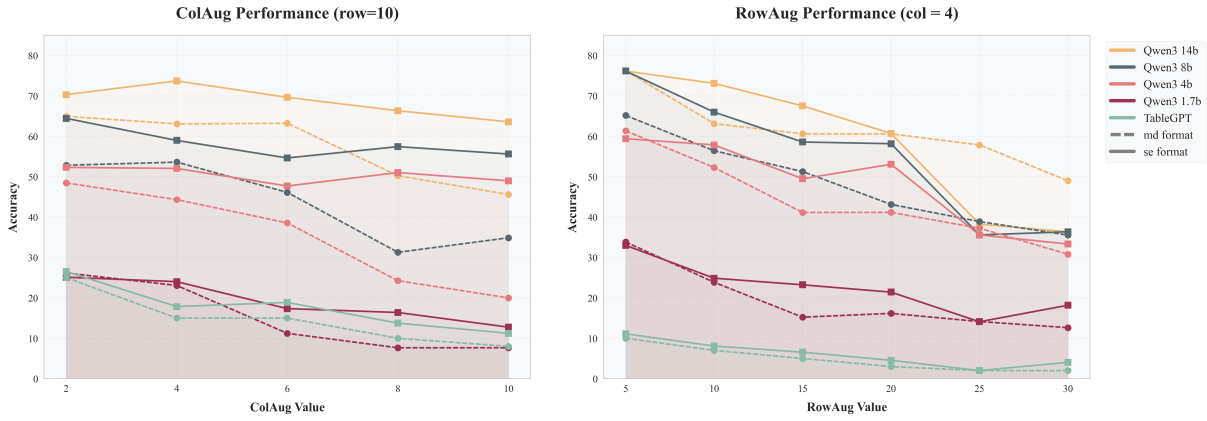


Figure 9: Relationship between model performance changes and table complexity

Table 13: Comparison between performance on Direct trap and Hidden trap problems

Model	Fmt	Missing			Contra		
		Direct trap	Hidden trap	Δ	Direct trap	Hidden trap	Δ
Qwen3 14B	Se	92.30	69.23	-23.07	28.68	28.57	-0.11
	Md	89.18	67.03	-22.15	21.47	28.21	6.74
Qwen3 8B	Se	93.79	70.97	-22.82	40.87	41.25	0.38
	Md	89.38	68.57	-20.81	31.47	34.54	3.07
Qwen3 4B	Se	91.72	68.67	-23.05	40.20	39.02	-1.18
	Md	88.19	69.75	-18.44	34.69	32.12	-2.57
Qwen3 1.7B	Se	50.17	37.33	-12.84	20.00	16.86	-3.14
	Md	46.85	22.40	-24.45	16.72	14.40	-2.32

Table 14: Main reults on TabularMath benchmark

Dataset	GSM8k	Fmt	Pure test				Robustness test			
			Easy	Medium	Hard	Avg	Well	Contra	Missing	Avg
Open source General Model										
Qwen3 14B	94.54	Se	79.94	70.94	69.79	73.55	58.55	28.57	69.23	54.15
		Md	77.87	70.21	61.59	69.89	55.89	28.21	67.03	51.88
Qwen3 8B	93.30	Se	75.17	62.62	56.99	64.92	44.44	41.25	70.97	50.35
		Md	73.18	55.63	47.30	58.70	36.71	34.54	68.57	44.07
Qwen3 4B	91.79	Se	73.53	56.85	46.47	58.95	39.63	39.02	68.67	46.77
		Md	71.57	52.68	41.73	55.32	39.31	32.12	69.75	45.15
Qwen3 1.7B	81.25	Se	54.40	31.13	19.20	34.92	26.81	16.86	45.71	29.03
		Md	50.76	30.62	18.84	33.40	33.33	14.40	37.33	29.56
Qwen2.5 14B	93.40	Se	79.90	68.06	62.34	70.10	59.40	6.40	22.40	36.90
		Md	79.21	64.10	49.09	64.13	60.00	6.80	20.00	36.70
Qwen2.5 14B coder	90.68	Se	71.59	60.38	49.90	60.62	47.40	30.00	52.40	44.30
		Md	72.63	57.74	45.61	58.66	47.40	23.60	51.60	42.50
Qwen2.5 7B	82.86	Se	35.56	21.36	19.39	25.43	39.20	13.60	34.00	31.50
		Md	53.92	34.45	20.64	36.33	37.40	16.00	34.80	31.40
Qwen2.5 7B coder	84.71	Se	62.35	42.01	29.79	44.47	30.80	24.40	43.60	32.40
		Md	64.78	42.13	23.52	43.47	33.40	20.80	34.00	30.40
Qwen2.5 3B	80.28	Se	36.37	22.71	16.94	25.34	2.20	84.00	91.20	44.90
		Md	39.74	23.96	15.68	26.46	6.20	69.60	79.20	40.30
LLama3.1 8B	83.69	Se	42.84	34.93	30.01	35.92	29.00	6.40	8.40	18.20
		Md	48.61	33.37	32.15	38.04	29.60	10.40	9.60	19.80
LLama3 8B	55.34	Se	28.92	15.22	10.63	18.25	12.80	30.80	37.20	23.40
		Md	36.30	21.12	20.68	26.03	16.60	19.20	35.20	21.90
Open-Source Math Model										
Qwen math 7B	95.45	Se	53.69	31.09	14.59	33.12	28.60	26.40	36.40	30.00
		Md	53.69	30.37	14.59	32.88	27.00	20.80	48.93	30.93
DeepSeek math 7B	80.13	Se	13.93	6.24	3.96	8.04	2.60	50.40	51.20	26.70
		Md	12.81	6.60	2.04	7.15	4.00	60.40	53.60	30.50
Open-Source Tabular Model										
TableGPT 7B	24.33	Se	30.13	18.86	12.60	20.53	26.20	26.80	44.80	31.00
		Md	30.60	16.44	17.64	21.56	30.60	23.20	46.40	32.70
StructLM 7B	32.97	Se	13.74	6.12	3.24	7.70	7.20	0	0	3.60
		Md	14.78	8.28	4.44	9.17	9.60	0	0	4.80
Closed-Source API										
DeepSeek v3	96.36	Se	88.45	87.27	85.71	87.14	68.60	68.40	85.20	72.70
		Md	88.63	87.63	85.83	87.37	68.60	68.00	82.80	72.00
GLM-4-plus	95.07	Se	83.37	81.15	79.83	81.45	68.80	32.80	69.60	60.00
		Md	84.52	81.03	78.27	81.27	71.40	27.60	65.60	59.00
GPT 4	94.46	Se	83.97	82.57	77.41	81.32	66.39	22.48	74.01	57.00
		Md	85.54	78.42	75.23	79.73	64.25	21.11	80.20	57.80

Table 15: Performance comparison under different reasoning settings across difficulty levels

Model	Difficulty Level								
	Easy			Medium			Hard		
	Direct	Code+Hint	Code	Direct	Code+Hint	Code	Direct	Code+Hint	Code
Qwen2.5 7B	53.92	67.12	23.95	34.45	67.25	26.73	20.64	64.74	37.01
Qwen2.5 7B Coder	64.78	72.33	49.13	53.32	71.89	53.32	23.52	70.64	59.72
Qwen2.5 14B	79.21	72.70	64.89	64.10	71.52	66.88	49.09	72.40	65.50
Qwen2.5 14B Coder	72.63	75.68	50.00	57.74	76.29	46.17	45.61	76.91	25.60
Qwen3 14B	77.87	72.46	59.93	70.21	70.51	55.71	61.59	71.14	60.81
Qwen3 8B	73.18	73.45	54.71	55.63	72.15	46.42	47.30	72.77	52.32
Qwen3 4B	71.57	69.73	33.87	52.68	67.50	35.01	41.73	67.63	38.77
Qwen3 1.7B	50.76	51.49	29.78	30.62	48.31	24.72	18.84	45.04	16.06

Table 16: Model Performance comparison of table reasoning and single-step table retrieval

Difficulty	Model	se-Retrival	se-Reason	md-Retrival	md-Reason
Easy	Qwen3 14b	95.04	79.94	91.54	77.87
	Qwen3 8b	92.44	75.17	93.02	73.18
	Qwen3 4b	93.02	73.53	92.15	71.57
	Qwen3 1.7b	81.10	54.40	85.71	50.76
Medium	Qwen3 14b	83.28	70.94	88.85	70.21
	Qwen3 8b	88.26	62.62	86.80	55.63
	Qwen3 4b	75.00	56.85	79.41	52.68
	Qwen3 1.7b	63.82	31.13	69.50	30.62
Hard	Qwen3 14b	75.73	69.79	81.36	61.59
	Qwen3 8b	86.98	56.99	82.24	47.30
	Qwen3 4b	60.65	46.47	66.27	41.73
	Qwen3 1.7b	58.28	19.25	56.80	18.84

Table 17: Model Performance Comparison with ColAug (md row=10)

Model	ColAug2	ColAug4	ColAug6	ColAug8	ColAug10
Qwen3 14b	64.94	63.07	63.21	50.25	45.59
Qwen3 8b	52.82	53.60	46.11	31.28	34.87
Qwen3 4b	48.45	44.32	38.54	24.26	20.00
Qwen3 1.7b	26.15	23.07	11.22	7.65	7.65
LLaMA3 8b	33.67	31.12	31.63	23.97	18.87
TableGPT	20.40	11.73	12.24	9.18	10.20

Table 18: Model Performance Comparison with ColAug (se row=10)

Model	ColAug2	ColAug4	ColAug6	ColAug8	ColAug10
Qwen3 14b	70.31	73.71	69.63	66.32	63.58
Qwen3 8b	64.43	58.97	54.63	57.43	55.61
Qwen3 4b	52.30	52.04	47.69	51.02	48.97
Qwen3 1.7b	25.12	24.01	17.34	16.38	12.75
LLaMA3 8b	33.67	36.22	34.18	35.71	32.14
TableGPT	26.53	17.85	18.87	13.75	11.22

Table 19: Model Performance Comparison with RowAug md

Model	RowAug5	RowAug10	RowAug15	RowAug20	RowAug25	RowAug30
Qwen3 14b	76.26	63.13	60.60	60.60	57.86	48.98
Qwen3 8b	65.15	56.41	51.26	43.14	38.89	35.53
Qwen3 4b	61.34	52.28	41.14	41.16	37.24	30.80
Qwen3 1.7b	33.83	23.85	15.22	16.16	14.14	12.62
LLaMA3 8b	40.40	38.88	36.36	36.68	28.28	20.20
TableGPT	9.18	7.65	7.65	4.59	4.59	2.04

Table 20: Model Performance Comparison with RowAug se

Model	RowAug5	RowAug10	RowAug15	RowAug20	RowAug25	RowAug30
Qwen3 14b	76.14	73.09	67.51	60.71	38.25	36.36
Qwen3 8b	76.14	65.98	58.58	58.16	35.57	36.36
Qwen3 4b	59.39	57.86	49.49	53.06	35.57	33.33
Qwen3 1.7b	32.99	24.87	23.23	21.42	14.09	18.18
LLaMA3 8b	43.43	40.40	33.32	29.29	25.25	26.76
TableGPT	11.11	8.08	6.56	4.54	2.02	4.04

Semantic Decoupling Prompt

"system_prompt":

> You are an experienced mathematician, and you are familiar with formal languages. I would like you to generate the formal form of a mathematical problem.
>
> You should express all logic in **SMT-LIB syntax**, using **prefix notation**. For example, multiplication should be written as ‘(* a b)‘ instead of ‘a * b‘.
>
> HIGHLIGHT!!!: **All numbers appearing after ’assert’ are written as floating point numbers.For example ’2’ is wrong and it should be replaced with ’2.0’.**

EXAMPLE INPUT:

- "problem": "Weng earns $12 an hour for babysitting. Yesterday, she just did 50 minutes of babysitting. How much did she earn?"

EXAMPLE OUTPUT:

- "problem": "Weng earns $12 an hour for babysitting. Yesterday, she just did 50 minutes of babysitting. How much did she earn?",
- "formal-problem": "(declare-fun hourly_rate () Int)
  (declare-fun minutes_worked () Real)
  (declare-fun hours_worked () Real)
  (declare-fun earnings () Real)
  (assert (= hourly_rate 12.0))
  (assert (= minute_worked 50.0))
  (assert (= minutes_per_hour 60.0))
  (assert (= hours_worked (/ minutes_per_hour)))
  (assert (= earnings (* hourly_rate hours_worked)))
  (check-sat)
  (get-value (earnings))"

"user_prompt:"

- "problem": {Question}

Table Transformation Prompt

"system_prompt":

The user will provide a problem and its formal representation. You need to convert the **explicitly assigned known data** of the problem into a tabular form.

The table should **only include variables that are directly assigned values in the problem** (e.g., via assertions like `(= variable value)`).

The table should include all variables that appear in the formal definition and their corresponding values: ("Given" or "Calculated").

Please wrap the value of this variable and the method of obtaining it in a list like: `[5,"Given"]`

Replace the variables that appear in the table in the original problem with unknowns to generate a generalized problem (i.e., `table + generalization = original problem`).

Set a **value range for each variable**, ensuring the ranges conform to common sense (they can be fixed values if appropriate).

EXAMPLE INPUT:

- "problem": "Weng earns $12 an hour for babysitting. Yesterday, she just did 50 minutes of babysitting. How much did she earn?",
- "formal_problem":
  "(declare-fun hourly_rate () Real)
  (declare-fun minutes_worked () Int)
  (declare-fun hours_worked () Real)
  (declare-fun earnings () Real)
  (assert (= hourly_rate 12.0))
  (assert (= minutes_worked 50))
  (assert (= minutes_per_hour 60))
  (assert (= hours_worked (/ minutes_worked minutes_per_hour)))
  (assert (= earnings (* hourly_rate hours_worked)))
  (check-sat)
  (get-value (earnings))"

EXAMPLE OUTPUT:

- "problem": "Weng earns $12 an hour for babysitting. Yesterday, she just did 50 minutes of babysitting. How much did she earn?",
- "table": ["name": "Weng",
  "hourly_rate": [12,"Given"],
  "minutes_worked": [50,"Given"],
  "minutes_per_hour": [60,"Given"],
  "hours_worked": [0.8333,"Calculated"],
  "earnings": [10,"Calculated"]],
- "generalization": "Weng earns $x an hour for babysitting. Yesterday, she just did t minutes of babysitting. How much did she earn?",
- "value_ranges":
  "name": null,
  "hourly_rate": "min": 7.25, "max": 100, "unit": "dollars"
  "minutes_worked": "min": 10, "max": 1440, "unit": "minutes"
  "minutes_per_hour": "min": 60, "max": 60, "unit": "minutes"

"user_prompt:"

- "problem": {Question}
- "formal_problem": {Formal_problem}

A.4 Case Analysis

A.4.1 Case Analysis of Hard Subset

- **Open source General model**: Take **Qwen14b** as an example (Figure 10)
- **Open-Source Math model**: Take **Qwen math 7b** as an example (Figure 11)
- **Open-Source Tabular model**: Take **TableGPT** as an example (Figure 12)
- **Closed source API**: Take **DeepSeek v3** as an example (Figure 13)

A.4.2 Case Analysis of Robust Subset

- **Open source General model**: Take **Qwen14b** as an example (Figure 14)
- **Open-Source Math model**: Take **Qwen math 7b** as an example (Figure 15)
- **Open-Source Tabular model**: Take **TableGPT** as an example (Figure 16)
- **Closed source API**: Take **DeepSeek v3** as an example (Figure 17)

Figure 10: Hard –Qwen14b

Reason 1: Retrieval Failed

Problem: Eliza's rate per hour for the first x hours she works each week is $y. She also receives an overtime pay of z times her regular hourly rate. If Eliza worked for t hours this week, how much are her earnings for this week?

OverRatio	OverHour	Name	BodyTemp	Age	RegularRate	Height	HeartRate	RegularHour
1.2	5	Eliza	38	62	10.0	179	97	40

Model Response: *From the provided table, we can find that Eliza's regular hourly rate (y) is $10.0, and her overtime multiplier (z) is 1.5. The number of regular working hours before overtime starts is not directly given but implied to be within the total hours worked as anything beyond that is considered overtime. .....*

Reason 2: Reasoning Failed

Problem: Charlie wants to sell beeswax candles. For every pound of beeswax, he can make x tapered candles. One pound of beeswax and the wicks cost $y in supplies. If he sells each candle for $z each, what is his net profit if he makes and sells t candles?

HeartRate	BodyTemp	Name	Age	CandlesMade	cost/pound	Height	price	candles/pound
83	36	Charlie	74	20	10.0	199	2.0	10

Model Response: *.....we find that Charlie's specific information is: Cost_Per_Pound (y): $10.0 Selling_Price (z):$2.0 Candles_Per_Pound (x): 10. However, there seems to be an inconsistency as the selling price per candle (z) should ideally be higher than the cost of producing one candle to ensure a profit. Typically, the selling price (z) would be higher than the cost per unit, but here, it appears to be lower which would imply a loss rather than profit.....*

Figure 11: Hard –Qwen math 7b

Reason 1: Retrieval Error

Problem: Janet's ducks lay x eggs per day. She eats y for breakfast every morning and bakes muffins for her friends every day with z. She sells the remainder at the farmers' market daily for $w per fresh duck egg. How much in dollars does she make every day at the farmers' market?

Age	HeartRate	eggs/day	price/egg	BodyTemp	eaten	name	for_muffins	SleepHours
				⋮				
25	72	16	2	39	3	Janet	4	5
				⋮				

Model Response: *.....For Janet: Eggs per day: 94 ,Eggs eaten: 3,Eggs for muffins: 4. First, calculate the number of eggs Janet has left after eating and making muffins: 94 - 3 - 4 = 87 .....*

Figure 12: Hard –TableGPT

Reason 1: No Ability to Solve

Problem: Kylar went to the store to buy glasses for his new apartment. One glass costs $x, but every second glass costs only y% of the price. Kylar wants to buy n glasses. How much does he need to pay for them?

price/glass	glasses	Weight	name	HeartRate	discount	BodyTemp	Height
				⋮			
5.0	16	98	Kylar	98	0.6	37	78
				⋮			

Model Response: *This problem is unsolvable.*

Reason 2: Retrieval Failed

Problem: Mishka bought x pairs of shorts, y pairs of pants, and z pairs of shoes. One pair of shorts costs $a. One pair of pants costs $b and one pair of shoes costs $c. How many dollars did Mishka spend on all the clothing items?

pants	pants price	SleepHours	shorts price	shoes price	shorts	Age	Weight	name	Height	shoes
					⋮					
3	22.5	8	16.5	42.0	3	21	53	Mishka	198	3
					⋮					

Model Response: *.....The table provides information about various individuals and their clothing costs, but it does not directly provide the values for x, y, and z (the number of pairs of shorts, pants, and shoes Mishka bought).....*

Figure 13: Hard –DeepSeek v3

Reason 1: Misunderstanding the Question

Problem: John plans to sell all his toys and use the money to buy video games. He has x lego sets and he sells them for $y each. He ends up buying z video games for $w each and has $v left. How many lego sets does he still have?

name	lego price	Sleep Hours	Body Temp	videogame price	video games	lego sets	Heart Rate	money left	Height
				⋮					
John	15	8	40	20	8	13	70	5	159
				⋮					

Model Response: .....*Since John plans to sell all his toys, he sells all 13 lego sets. Therefore, he has 0 lego sets left.*

Reason 2: Reasoning Error

Problem: Sasha notices that prices for lumber have gone up x% in the last few months after she bought some lumber. Since she has leftovers, she decides to sell them. She has leftover s small boards that cost her $c each. She also has l large boards she bought for $d each. If she sells them all, how much profit does she make?

small boards	cost large boards	BodyTemp	name	cost small boards	Age	large boards	Height	Weight	percentage increase
				⋮					
10	16.0	39	Sasha	10	37	5	150	51	0.5
				⋮					

Model Response: .....*Selling price for small boards: (c times (1 + x/100) = 10.0 times (1 + 0.5/100) = 10.0 times 1.005 = 10.05.*

Figure 14: Robust –Qwen14b

Reason 1: Retrieval Failed

Problem: Jordan has x children who wear diapers. Each child requires y diaper changes per day. Jordan's wife changes half of the diapers. How many diapers does Jordan change per day?

children	diaper_changes_per_child	name
	⋮	
2	5	Jordan
	⋮	

Model Response: *.....we see various entries for different names and their respective children and diaper change requirements. From the table, the only mention of Jordan is as the name of a child with 2 children needing 5 diaper changes per day. But since the question implies that Jordan (as a parent) has x children, we need to clarify which set of children belong to Jordan as a parent. Since this information isn't clearly specified in the question or table, it's not possible to determine exactly how many children Jordan has as a parent or their specific diaper needs.. .....*

Reason 2: Fail to Identify Contradictions

Problem: Bud makes homemade macaroni and cheese once a week. The pasta costs $x a box, and he spends $y on cheddar cheese and twice that amount for the gruyere cheese. How much money does Bud spend on making macaroni and cheese in one year?

cost gruyere cheese	weeks per year	total cost week	cost cheddar cheese	cost pasta	name
		⋮			
10.0	52	5.0	3.0	1.0	Bud
		⋮			

Model Response: *.....The cost of cheddar cheese (y) = 3.0. The cost of gruyere cheese = 10.0. Since the cost of gruyere cheese is twice the cost of cheddar cheese, this aligns with the provided values. The weekly cost for Bud is thus:Total weekly cost = x + y + 2y = 1.0 + 3.0 + 2(3.0) = 1.0 + 3.0 + 6.0 = 10.0.....*

Figure 15: Robust –Qwen math 7b

Reason 1: Contradictions Identified but Wrong Answer

Problem: Andrew plans a road trip from New Jersey to Rochester. It takes x days to travel by bus and half as many days to travel by car from New Jersey to Rochester. How many days will he travel if he plans to go to Rochester by bus and return from Rochester by car?

name	bus_days	car_days
⋮		
Andrew	6	2
⋮		

Model Response: *.....From the table, we can see that Andrew has a bus days value of 6 and a car days value of 2. The problem states that it takes half as many days to travel by car as it does to travel by bus. However, the table provides a different value for car days, which is 2. This suggests that the table may have an error or that the problem may have an additional condition that we need to consider.Given the information in the table, we can use the bus days value to calculate the total travel time. The total travel time is the sum of the bus days and the car days.Total travel time = Bus days + Car days = 6 + 2 = 8. Therefore, the total number of days Andrew will travel is 8.....*

Reason 2: Missing Identified but Wrong Answer

Problem: Aleksandra went to a restaurant for dinner. She ordered some soup, a bagel, and a piece of cake. The bagel cost $x, and the soup 25% more. The cake is only half of the price of the bagel. How much did Aleksandra need to pay for the dinner she ordered?

bagel_cost	name
3.2	Alexander
15.8	Aiden
10.7	Sophia
⋮	
null	Aleksandra
⋮	

Model Response: *.....we are interested in the cost of the bagel ordered by Aleksandra, which is not specified in the table. However, we can assume that the table is meant to provide a reference for the cost of the bagel, and we can use the average cost of the bagel from the table to find a reasonable estimate.....*

Figure 16: Robust –TableGPT

Reason 1: Fail to Identify Contradiction

Problem: Andrew plans a road trip from New Jersey to Rochester. It takes x days to travel by bus and half as many days to travel by car from New Jersey to Rochester. How many days will he travel if he plans to go to Rochester by bus and return from Rochester by car?

name	bus_days	car_days
⋮		
Andrew	6	2
⋮		

Model Response: .....*The table provides the bus and car days for several individuals, including Andrew. From the table: Andrew's bus days: 6 , Andrew's car days: 2. If Andrew plans to go to Rochester by bus and return from Rochester by car, the total number of days he will travel is the sum of his bus days and car days. Total days = Bus days + Car days = 6 + 2 = 8*.....

Reason 2: Regard Missing Info as 0

Problem: Samwell owns an appliances store. For this week, one-fourth of their sales are smart TVs, one-eighth are analog TVs, and the rest are OLED TVs. If they were able to sell a total of x TVs, how many OLED TVs were they able to sell?

total_tvs	name
708	Sebastian
503	Sophia
935	Emily
⋮	
null	Aleksandra
⋮	

Model Response: .....*According to the table, Samwell's total TV sales are listed as 0 (None). Since Samwell's total TV sales are 0, we can calculate the number of OLED TVs sold by using the given fractions for smart TVs and analog TVs.One-fourth of the sales are smart TVs: 1/4 times 0 = 0. One-eighth of the sales are analog TVs:1/8 times 0 = 0. The rest of the sales are OLED TVs: 0 - 0 - 0 = 0*.....

Figure 17: Robust –DeepSeek v3

Reason 1: Contradictions Identified but Reasoning Error

Problem:

first_hour_cost	name	subsequent_hour_cost	total_hours
		⋮	
15	James	35	4
		⋮	

Model Response: *.....we find the row corresponding to James: First_Hour_Cost (x): 15, Subsequent_Hour_Cost:35* ***(Note: This is not directly used in the calculation since the problem states each subsequent hour is twice the previous hour's cost, not a fixed subsequent cost)*** *Total_Hours: 4. First paid hour:15. Second paid hour:2 × 15 = 30. Third paid hour:2 × 30 = 60. Total Cost = 15+ 3 + 60 = 105..... (The correct answer should be 15+30+30=75)*

Reason 2: Regard Missing Info as 0

Problem: Elaina is holding the final concert in her tour. To celebrate her final concert, she makes the concert twice as long as her usual concerts. At the end of the concert, she also performs a t-minute encore. If the runtime of this final concert is f minutes then how long, in minutes, do her usual concerts run for?

name	final_concert_length	encore_length
James	146	22
Ella	115	5
Matthew	92	24
	⋮	
Elaina	65	null
	⋮	

Model Response: *.....From the table, we have the following data for Elaina: Final_Concert_Length (f): 65. Encore_Length (t): None (which we can interpret as 0 minutes, since no encore is performed) According to the problem, the final concert is twice as long as the usual concert, thus x = 65/2 = 32.5.....*