NEEDLEINATABLE: Exploring Long-Context Capability of Large Language Models towards Long-Structured Tables

Lanrui Wang[1*†], **Mingyu Zheng**[1,2*], **Hongyin Tang**[3*†], **Zheng Lin**[1,2‡],
Yanan Cao[1,2], **Jingang Wang**[3†], **Xunliang Cai**[3†], **Weiping Wang**[1]
[1]Institute of Information Engineering, Chinese Academy of Sciences
[2]School of Cyber Security, University of Chinese Academy of Sciences
[3]Meituan
oliveerwang@tencent.com
{zhengmingyu,linzheng}@iie.ac.cn tanghongyin@meituan.com

Abstract

Processing structured tabular data, particularly large and lengthy tables, constitutes a fundamental yet challenging task for large language models (LLMs). However, existing long-context benchmarks like Needle-in-a-Haystack primarily focus on unstructured text, neglecting the challenge of diverse structured tables. Meanwhile, previous tabular benchmarks mainly consider downstream tasks that require high-level reasoning abilities, and overlook models' underlying fine-grained perception of individual table cells, which is crucial for practical and robust LLM-based table applications. To address this gap, we introduce NEEDLEINATABLE (NIAT), a new long-context tabular benchmark that treats each table cell as a "needle" and requires models to extract the target cell based on cell locations or lookup questions. Our comprehensive evaluation of various LLMs and multimodal LLMs reveals a substantial performance gap between popular downstream tabular tasks and the simpler NIAT task, suggesting that they may rely on dataset-specific correlations or shortcuts to obtain better benchmark results but lack truly robust long-context understanding towards structured tables. Furthermore, we demonstrate that using synthesized NIAT training data can effectively improve performance on both NIAT task and downstream tabular tasks, which validates the importance of NIAT capability for LLMs' genuine table understanding ability. [4].

1 Introduction

The long-context modeling ability has laid the foundation for a wide range of LLM-based applications, such as lifelong conversational chatbots, document analysis tools, and advanced agent-based systems [1–4]. Correspondingly, lots of effort has been dedicated to scaling up the context window of LLMs, improving their proficiency in handling extremely long input texts [5–9]. In parallel, various benchmarks like Needle-in-a-Haystack (NIAH) [10] have been developed to evaluate the long-context capabilities of LLMs from diverse perspectives, such as extracting key information from long texts, answering questions about extended passages or multiple documents [11–15].

* Equal contribution. Work was done when Lanrui Wang was a graduate student of IIE, CAS.
† Joint work with Meituan
‡ Zheng Lin is the corresponding author.
[4]Our code and data are available at: https://github.com/wlr737/NeedleInATable

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

 https://doi.org/10.52202/085713-4121

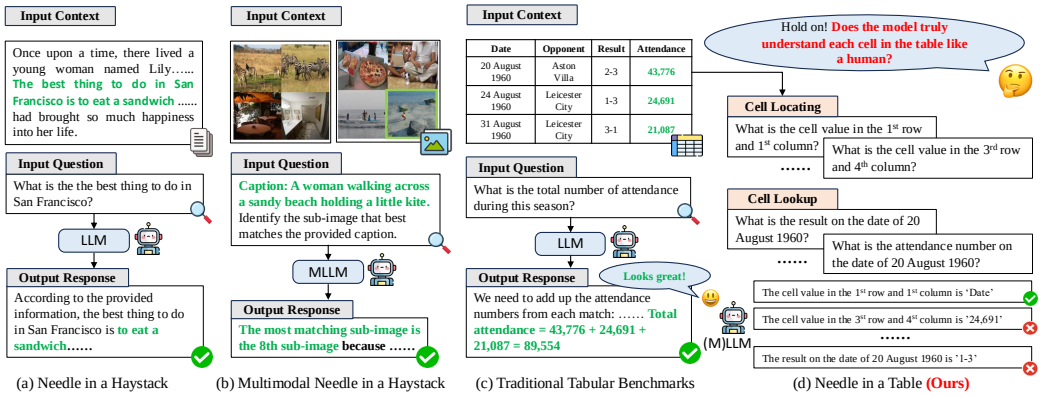

Figure 1: Comparison of previous long-context benchmarks, tabular benchmarks and the proposed NIAT benchmark. Existing long-context benchmarks overlook the structured tabular data, while traditional tabular benchmarks mainly focus on high-level complex reasoning ability. Both of them ignore the model's basic fine-grained comprehension of individual cells in the table context.

Beyond lengthy unstructured textual data, structured tables also play a critical role in real-world long-context scenarios. Featured with unique two-dimensional structures and diverse formats, tables are widely used in different domains to organize and present information, ranging from financial documents, scientific research to medical records [16–18]. Moreover, LLM-based chatbot users often feed serialized tables in different formats like Markdown into LLMs' context windows to perform various table understanding tasks such as table question answering (TQA) and table fact verification (TFV) [19, 20]. Consequently, the lengths of these table-related task requests often exceed that of common instructions with a few sentences, presenting a distinct long-text challenge for LLMs.

However, existing long-context benchmarks primarily focus on evaluating the ability of LLMs to comprehend long unstructured text, overlooking in-depth exploration of long-context scenarios of structured tables. On the other hand, previous tabular LLM studies mainly assess LLMs' proficiency with traditional benchmarks of downstream tabular tasks that demand high-level reasoning abilities based on partial table information, but neglect the importance of pressure testing LLMs' underlying fine-grained understanding of each individual table cell. This is crucial for practical LLM-based table applications as real-world users could input questions about any table cells. Besides, it can help determine whether we have developed a truly robust table understanding ability or if we are barking up the wrong tree by overly pursuing the model performance on specific benchmarks.

To bridge this gap, we introduce a new task and construct a benchmark based on publicly available tables, termed NEEDLEINATABLE (NIAT). As illustrated in Figure 1, unlike previous long-context benchmarks and tabular benchmarks, the proposed NIAT task treats each table cell as a "needle" and requires the model to extract the target cell according to two types of questions, respectively. The cell-locating questions ask the model to locate the specific cell at the given position, and the cell-lookup questions require the model to retrieve the answer cell based on the information lookup demand. Our benchmark contains 750 tables and up to 287K questions in total, covering three table structures (flat, horizontal and hierarchical tables), three table formats (Markdown, HTML and image) and diverse table sizes (fixed and arbitrary).

With the help of NIAT benchmark, we evaluate a wide spectrum of LLMs and MLLMs including mainstream open-source models and close-sourced GPT-4o. These models demonstrate varying performance on the NIAT task and they obtain much better results on cell-lookup questions than cell-locating questions. This shows that current LLMs can retrieve answer cells from input tables using the attention mechanism, but they struggle in interpreting the basic two-dimensional table structures in a human-like perspective. More importantly, our thorough evaluation reveals a substantial performance gap between more complex downstream tabular tasks (like TQA and TFV) and the NIAT task, suggesting that they may rely on dataset-specific correlations or shortcuts to obtain better benchmark results but lack truly robust long-context understanding towards structured tabular data.

https://doi.org/10.52202/085713-4121

Furthermore, we explore whether enhancing NIAT capabilities can fundamentally improve the overall table understanding ability. To this end, we propose a strong2weak data synthesis method to create NIAT fine-tuning data. Specifically, we first utilize the in-context learning with GPT-4o to generate various NIAT task queries based on tables from training splits, and then synthesize detailed chain-of-thought (CoT) reasoning processes as target responses, which instruct weak LLMs to progressively understand structured tabular data. Experimental results with Llama3.1-8B-Instruct and Qwen2.5-7B-Instruct demonstrate that our synthesized 12K NIAT training data can not only improve models' long-context performance on the NIAT task, but also achieve a significant performance gain (14.55%↑ in average accuracy) on four downstream benchmarks (WTQ, TabFact, HiTab and TABMWP), outperforming strong baselines such as long-context LLMs and specialist tabular LLMs, which further underscores the importance of NIAT capabilities. We hope that this work could foster advancements in both long-context LLM and tabular LLM communities, facilitating the development of LLMs with robust understanding of complex and lengthy structured tables.

We conclude our contributions as follows:

- Targeted at the limitations of existing benchmarks for long-context and table understanding, we introduce a new task named NIAT and construct the first benchmark for evaluating LLMs' long-context table understanding ability.
- On this basis, we conduct a thorough evaluation of various types of LLMs and MLLMs, revealing their shortcomings in long-context understanding of lengthy tables and providing valuable insights.
- We propose a simple yet effective data synthesis method to enhance LLMs' long-table comprehension capabilities, which also significantly improves their performance on downstream table-related tasks, outperforming recent state-of-the-art baselines.

2 Related Work

Long Context LLMs and Benchmarks. Recently, the long-context capabilities of LLMs have garnered significant attention in the research community. Numerous efforts have focused on enhancing positional embeddings to develop more powerful long-context LLMs [21–24]. Extensive studies have also explored and evaluated LLMs' ability to understand long-context data across various modalities [25, 26, 13] and generate long-form content [27, 28]. To evaluate the long-context capabilities of large language models (LLMs) in the text modality, benchmarks such as RULER [15], Infini-Bench [11], and LongBench [1] have introduced diverse long-context understanding tasks across varying context lengths, providing a comprehensive evaluation framework for developing robust long-context LLMs. However, these benchmarks primarily focus on unstructured text, largely overlooking structured table data. In this paper, we address this gap by proposing the NIAT tasks.

Notably, a concurrent long-context benchmark, LongBench-v2 [12], evaluates the high-level reasoning ability of LLMs over long structured tables using 18 question-answering samples. In contrast, our work focuses on assessing LLMs' comprehension of every individual cell within the global context of long tables, making it complementary to LongBench-v2.

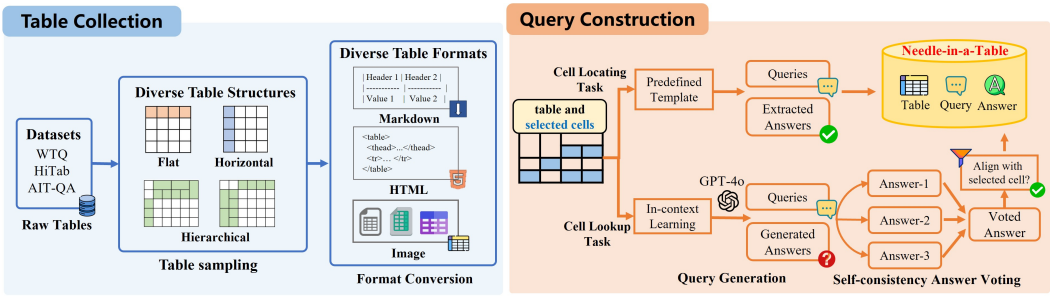

Figure 2: The construction pipeline of Needle-in-a-Table Benchmark.

Tabular LLMs and Benchmarks. Recent years have witnessed a paradigm shift to developing Tabular LLMs for table understanding, where different strategies have been applied to enhance LLMs'

https://doi.org/10.52202/085713-4121

ability to fulfill tabular tasks, such as prompt engineering [29, 30] and instruction tuning [31–34]. Moreover, a series of benchmarks have been constructed to evaluate LLMs' proficiency in table understanding tasks within diverse scenarios [35–37]. Nonetheless, these benchmarks primarily focus on assessing LLMs' high-level table-based reasoning ability with complex tasks as testbeds, such as table question answering and table-based fact verification, and fail to evaluate the fine-grained understanding of each input table cell from the long-context perspective, which is the foundation for humans to comprehend tables and perform more complex tasks. To this end, we propose the NIAT benchmark to address this issue and provide a comprehensive landscape of LLMs' table understanding capacity over the whole input table.

3 NEEDLEINATABLE Benchmark

3.1 Benchmark Construction

3.1.1 Table Collection

To fill the gap in existing benchmarks and provide the community with the first long-context benchmark towards structured tables, it is most cost-effective to build such a benchmark based on tables from publicly aviable datasets. Moreover, real-world tables possess various structures, ranging from simple flat tables to hierarchical tables with multi-level headers and merged cells [16, 38, 39]. With these considerations in mind, we collect tables of three different structures from existing tabular benchmarks. **(1) Flat tables**: we randomly select 250 tables from WTQ benchmark [19] which contains flat tables from Wikipedia. **(2) Hierarchical tables**: we randomly select 220 tables and 30 tables from HiTab [40] and AIT-QA [41] benchmarks respectively, which contain diverse hierarchical tables from Wikipedia, statistic reports, and annual reports of airline companies. **(3) Horizontal tables**: we write Python scripts to transpose flat tables in WTQ so that the first column serves as row headers, resulting in 250 horizontal tables.

Another unique feature of structured tables lies in the diversity of table representation formats [30, 42, 34]. Chatbot users may input tables in different formats into LLMs' context windows, such as Markdown, HTML, csv, Latex, and even image format. Therefore, we carefully design Python scripts to transform the collected tables into three commonly used formats including **Markdown, HTML and image** formats. Table sequences in Markdown and HTML formats are used to evaluate textual LLMs. For image format, we render tabular data into images following the previous work [34], and the resulting table images serve as inputs for multimodal LLMs to evalute their NIAT capabilities.

3.1.2 Query Construction

Despite the remarkable performance of LLMs on complex tabular benchmarks, even surpassing human levels, does this really represent that LLMs possess the same robust long-context table understanding abilities as humans? To answer this question, we take a step back to the fundamental or atomic ability of humuan-like table comprehension, focusing on fine-grained perception of individual table cells. Concretely, we treat the input table as the context, the single table cell as the target "needle", and design two types of NIAT queries that are simple and trivial for humans but constitute the foundation of more complex table-based tasks.

Cell Locating queries require the model to extract the cell content according to the given cell location (i.e., row index and column index), which aims at evaluating the understanding of basic two-dimensional table structures. An example query could be "What is the cell value in the 3rd row and 4th column". If the model has a correct understanding of the basic table structure, it should be easy to locate the target cell by enumerating along the row and column directions.
Cell Lookup queries are simple lookup questions where answers are specific table cells and do not need any further aggregate operations (such as min/max/sum/count) or multi-step reasoning. Using the table in Figure 1 as an example, multiple lookup questions can be raised based on different table cells, e.g., "What is the result on 20 August 1960". This task requires the model to use the keywords in the questions as cues and retrieve the answer cell from the table by intersecting the row and the column with overlapped keywords.

To save the cost of model inference, we randomly select 60% table cells from each table as target cells to create NIAT queries. Since the cell locating task only takes the table and the target cell location as input, we fill in the pre-defined prompt template with selected row indices and column

https://doi.org/10.52202/085713-4121

Table 1: Performance of LLMs on NIAT benchmark. 'Flat', 'Hori.' and 'Hie.' denote flat, horizontal and hierarchical tables, respectively. 'MD.' and 'HT.' indicates tables in Markdown and HTML formats. The best results in each category are highlighted in **bold**.

Model	Cell-Locating							Cell-Lookup							Overall
	Flat		Hor.		Hie.		Ave	Flat		Hor.		Hie.		Ave	
	MD.	HT.	MD.	HT.	MD.	HT.		MD.	HT.	MD.	HT.	MD.	HT.		
Open-source LLMs															
Mistral-7B-Instruct-v0.3	7.75	6.79	4.35	6.33	2.11	0.28	4.60	33.72	38.79	27.99	27.55	25.62	33.79	31.98	18.29
Deepseek-llm-7B	2.88	0.16	2.31	2.03	1.13	0.70	1.54	47.46	19.37	32.21	27.03	38.29	26.05	31.74	16.64
MiniCPM-3-4B	1.48	2.01	0.87	1.57	0.73	0.40	1.18	64.60	66.51	49.65	51.54	59.12	57.38	58.13	29.66
InternLM2.5-7B-chat	1.36	1.36	1.05	1.34	0.54	0.09	0.96	36.31	37.34	27.85	27.03	43.15	39.89	35.26	18.11
Yi-1.5-9B-chat-16K	7.29	5.36	4.31	7.45	1.77	0.26	4.41	52.81	45.26	43.14	44.26	54.31	54.52	49.05	26.73
GLM-4-9B-chat	8.33	9.76	5.76	7.32	2.45	**6.83**	6.74	55.09	48.48	42.40	36.89	61.20	50.98	49.17	27.96
Qwen2.5-7B-Instruct	16.24	13.64	7.63	15.85	2.90	0.49	9.46	44.80	48.20	38.92	42.39	56.40	54.90	47.60	28.53
Qwen2.5-14B-Instruct	19.18	18.28	11.41	26.27	4.69	0.39	13.37	42.73	36.80	38.29	36.46	56.16	51.84	43.71	28.54
Llama3.1-8B-Instruct	10.38	8.75	7.18	7.70	2.54	0.38	6.16	67.30	65.10	57.10	66.25	72.50	66.20	65.74	35.95
Qwen3-14B	17.6	18.73	18.41	23.81	4.33	0.81	13.95	74.41	70.76	64.14	62.2	72.79	79.59	70.65	42.30
Qwen3-32B	22.41	**22.6**	21.32	**31.05**	**5.77**	1.19	**17.39**	76.6	71.24	69.23	63.8	80	80.6	73.58	45.48
Qwen3-30B-A3B	**23.17**	20.41	**24.12**	26.2	3.65	1.39	16.49	**78.87**	**78.83**	**72.2**	**69.8**	**85.41**	**86.59**	**78.62**	**47.55**
Tabular LLMs															
StructLLM	2.00	2.10	1.30	4.20	1.20	0.10	1.82	58.65	46.70	25.85	29.65	51.85	43.55	42.72	22.27
TableGPT2	**12.67**	**12.83**	**6.32**	**17.52**	**3.14**	**0.52**	**8.84**	**77.02**	**71.31**	**61.03**	**71.64**	**80.73**	**81.50**	**73.87**	**41.36**
Reasoning LLMs and GPT-4o															
GPT-4o	38.50	29.70	36.50	39.40	10.40	1.50	26.00	61.50	63.80	60.30	59.50	83.60	81.10	68.30	47.15
Qwen-QwQ-32B	63.76	54.40	36.36	52.29	3.90	1.80	35.42	48.78	45.21	45.14	45.84	56.55	56.90	49.74	42.58
DeepSeek- R1	**76.20**	**79.50**	**75.46**	**82.30**	**72.39**	**9.60**	**65.91**	**81.00**	**77.82**	**79.19**	**77.50**	**85.60**	**84.80**	**80.99**	**73.45**

indices to create final input queries. For cell lookup task, we provide GPT-4o with the table and the selected target cell, and utilize in-context learning with GPT-4o to generate corresponding lookup questions. We add JSON output requirements in prompts of two tasks to minimize errors during answer parsing. To guarantee the validity of synthesized cell lookup questions, we use GPT-4o with self-consistency [43] to answer these questions and filter out questions whose answers do not align with the target cells. Finally, we obtain 142K cell-lookup queries and 145K cell-locating queries.

3.2 Benchmark Statistics

We have developed NEEDLEINATABLE comprising 750 table and 287K test cases, with an average of 382.67 test cases per table. 80% tables have a row size ranging from 1 to 40, and a column size spaning from 1 to 23. 80% test data have a prompt length within 4.2K, with the maximum length up to 120K. For tables in image format, we randomly select 120 tables from each structure and their corresponding queries as test data for multimodal LLMs. The key statistics about collected tables and constructed queries are shown in Tables 5 and 6, such as average row number and column number, average input length, and so on. Traditional tabular benchmarks mainly evaluate models' complex tabular reasoning with limited test cases, which can only span a small number of table cells. By contrast, the NIAT benchmark evaluates the underlying fine-grained perception of individual table cells with much more test cases, covering various table structures, formats and sizes. **Detailed information about NIAT benchmark are shown in Appendix B.**

3.3 Evaluation and Analysis

Baselines. We evaluate models across four categories: (1) *Generalist LLMs and MLLMs*, such as Mistral-7B-Instruct-v0.3 [44], Llama3.1-8B-Instruct [8], and InternVL-2.5-8B [45]. (2) *Tabular LLMs and MLLMs* that are fine-tuned with table instruction-tuning data, including StructLLM [46], TableGPT2 [33] and Table-LLaVA [34]. (3) *Recent Reasoning LLMs* including DeepSeek-R1 [47], QwQ-32B [48] with enhanced reasoning ability derived from reinforcement learning. (4) *Close-sourced GPT-4o*. Considering the high cost of GPT-4o and DeepSeek-R1 API, for tables in Markdown,

 https://doi.org/10.52202/085713-4121

HTML and image formats, we sample 3K test samples of two types of NIAT tasks, respectively. For all baselines, we adopt the zero-shot setting during evaluation.

Performance of LLMs. From the results in Table 1, we have the following findings. (1) Different open-source LLMs demonstrate varying performance on NIAT tasks, with Qwen2.5 family performing best on cell-locating task and Llama3.1-8B-Instruct on cell-lookup task. The competitive performance of Qwen2.5 series models on both NIAT tasks could be stemmed from the specially collected table-related post-training data for enhancing table understanding capability [49]. MiniCPM3-4B also obtains strong performance on cell-lookup task and even greatly surpasses some LLMs with larger sizes, showcasing the potential for developing efficient LLMs with great table understanding ability.

(2) Existing LLMs and tabular LLMs demonstrate much better performance on cell-lookup task compared to cell-locating task. This suggests that current LLMs can utilize semantic co-occurrence between the question and the table to identify question-related rows and columns, and subsequently extract the answer cells via row-column intersections. However, they struggle in accurately comprehending the basic table structures, indicating the fundamental gap between the ways of LLMs and humans in understanding tabular data. To shed more light on the LLMs' perspective of structured tables, we further analyze the attention weights distribution from representative LLMs.

As illustrated in Fig. 3, we have observed certain attention patterns. This figure visualizes the attention weights, where the input table is serialized into a sequence in a left-to-right, top-to-bottom order. The horizontal and vertical coordinates (m, n) correspond to the cell in the m-th row and n-th column, and the color intensity represents the attention weight—brighter colors indicate larger weights. Based on this, we categorized the observed patterns into the **Multi-Slash** and **Local-Triangle** patterns. In the *Multi-Slash* pattern, the attention weights are concentrated on the cell tokens in the same column, thereby exhibiting multiple slash lines at fixed intervals (i.e., the number of cells in one row). In the *Local-Triangle* pattern, the attention weights are concentrated on the cell tokens in the same row within local windows, especially the row header. With these attention patterns, LLMs could retrieve the target cell located in the row and the column that contain keywords of cell-lookup questions. Nevertheless, such a table understanding approach may still fail to guarantee the robust interpretation of table structures, leading to failed edge cases that restrict the reliability of LLM-based applications.

(3) Results in Table 1 on tables of arbitrary sizes show that seemingly powerful LLMs actually lack robust long-context table understanding ability, especially table structure comprehension. To more intuitively demonstrate the influence of different table sizes and increased context length, we crop flat Markdown tables into fixed sizes (15 tables for each size), ranging from 8x8, 12x12, to 32x32, and evaluate the ability of representative LLMs to extract cells from each table position with newly constructed 48K cell-locating queries.

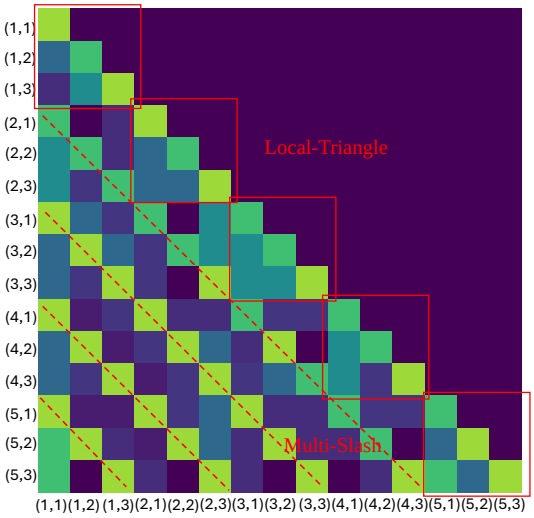

Figure 3: The illustration of LLMs' attention patterns for structured tables. The input tables are in Markdown format and '(m,n)' indicates cell tokens in the *m-th* row and *n-th* column.

From the per-cell accuracy heatmap illustrated in Figure 4, we can find that GPT-4o demonstrate much better table structure understanding capacity than open-source LLMs like Qwen-2.5-7B-Instruct and Llama3.1-8B-Instruct, with more green color in the heatmap. More importantly, we observe that there exists **lost-in-the-middle-table** phenomenon for LLMs including GPT-4o in understanding lengthy tables [50], i.e., LLMs possess better perception of table cells in the first row and the last row, but struggle with cells in the middle part of tables. Besides, as the table size increases, LLMs suffer a significant performance decline, highlighting the unique challenge of lengthy tabular data to LLMs' long-context abilities. Unlike traditional 'Needle-in-a-haystack' where current LLMs have achieved near-perfect performance, there remains significant room for improvement in the NIAT task.

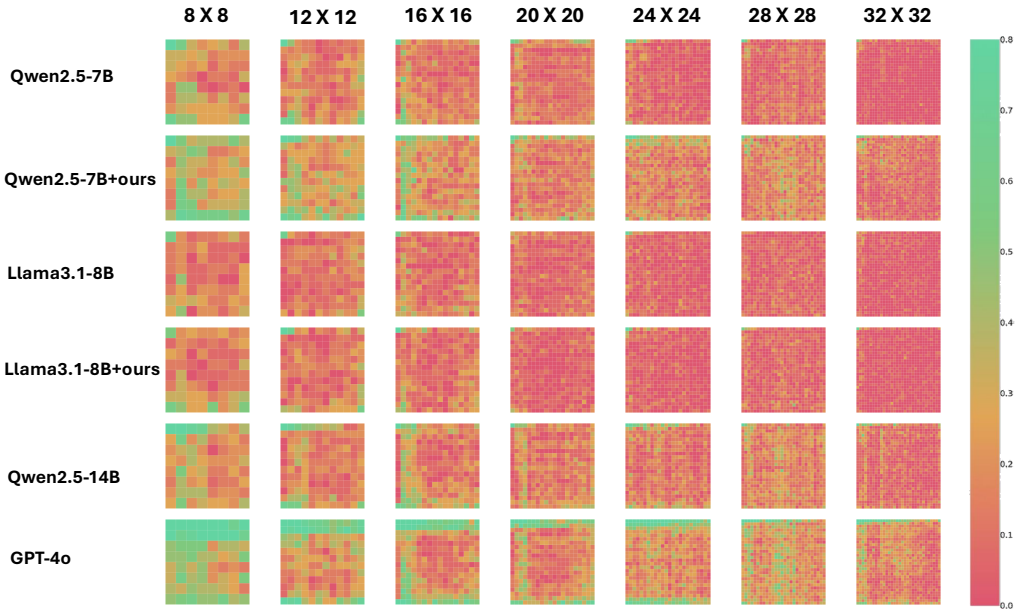

Figure 4: The per-cell accuracy heat map of cell-locating task on tables of fixed sizes, with redder indicating lower accuracy and greener indicating higher accuracy. The x-axis represents tables of different sizes (e.g., 8×8 denotes tables with 8 rows and 8 columns), while the y-axis lists evaluated LLMs in instruct version. + *ours* denotes models fine-tuned with our synthesized NIAT training data.

(4) The performance of hierarchical tables is relatively worse than the other two table structures, demonstrating the increased difficulty due to multi-level headers and merged cells. Compared with Markdown format, most models obtain worse results with HTML format, showcasing the unrobustness of current LLMs in handling various table formats and structures, which is consistent with findings from prior works [42, 30].

(5) Compared to generalist LLMs, recent tabular LLM such as TableGPT2 achieve better results, validating the effectiveness of its 2.36M in-house table instruction tuning data. Among various baselines, DeepSeek-R1 achieves the state-of-the-art performance across different settings, demonstrating that test-time scaling techniques could effectively enhance LLMs' ability to process long-structured tabular data. By analyzing model outputs, we find that its in-depth reasoning capability enables more accurate understanding of table structure and extraction of target cells, thereby facilitating more complex downstream tasks.

Performance of MLLMs. MLLMs directly take table images as input, which more closely mimic the way humans understand tabular data. From the results in Table 2, we have the following observations. (1) Recent state-of-the-art MLLMs like Skywork-R1V3-38B achieve superior performance in NIAT tasks compared to earlier models, demonstrating the advantages of their optimizations in understanding larger table images, such as supporting dynamic high image resolution and introducing extra post-training data for text-intensive images.

(2) Similar to the performance trend observed in LLMs, multimodal LLMs also perform significantly worse on cell-locating tasks compared to cell-lookup tasks, which reveals their defects in perceiving basic table structures. We speculate that the way the visual encoder of MLLMs tokenizes input images could disrupt the integrity of table structures, e.g., traditional ViT models segment input images into fixed-size grid patch tokens, thereby resulting in fragmented table images. Developing advanced semantic equivalent visual tokenizers that can maintain the completeness and the structure of input table images may alleviate this problem [51].

(3) Combining with the experimental analysis of LLMs, we can infer an important fact: **although seemingly powerful large models including table-oriented models have achieved good per-**

https://doi.org/10.52202/085713-4121

Table 2: Performance of representative MLLMs on NIAT benchmark. 'Flat', 'Hori.' and 'Hie.' denote flat, horizontal and hierarchical tables, respectively. The best and suboptimal results are highlighted in **bold** and underlined.

Model	Cell-Locating				Cell-Lookup			
	Flat	Hori.	Hie.	Ave.	Flat	Hori.	Hie.	Ave.
GPT-4o	28.46	15.34	11.4	18.40	66.9	61.43	76.65	68.33
DeepSeek-VL2-tiny-3B	5.89	4.08	5.47	5.15	55.29	25.44	54.68	45.14
GLM-4V-9B-Chat	4.73	3.72	2.68	3.71	52.63	24.41	50.00	42.35
InternVL-2.5-8B-Chat	15.53	10.76	12.44	12.91	37.89	33.37	24.39	31.88
Llava-1.5-7B	0.69	0.56	0.93	0.73	2.71	4.63	1.50	2.95
Table-Llava-7B	7.65	4.90	11.62	8.06	9.58	5.59	2.68	5.95
Llava-1.6-7B	2.57	1.61	3.50	2.56	16.88	5.39	7.80	10.02
MiniCPM-V-2.6-8B	23.99	16.21	**20.12**	20.11	59.50	36.10	37.37	44.32
Phi-3.5-Vision-Instruct-4.2B	2.28	2.61	4.13	3.01	41.09	22.17	36.48	33.25
Phi-4-multimodal-instruct	4.06	4.49	6.47	5.01	57.86	31.22	26.87	38.65
Qwen2-VL-7B-Instruct	5.53	4.43	5.07	5.01	63.04	21.83	53.24	46.04
Qwen2.5-VL-3B-Instruct	8.01	5.45	7.77	7.08	59.71	38.02	65.96	54.56
Qwen2.5-VL-7B-Instruct	10.53	7.92	9.39	9.28	70.24	51.83	72.32	64.80
Llama-3.2-11B-Vision-Instruct	4.46	4.25	5.99	4.90	37.35	36.46	51.74	41.85
Qwen2.5-VL-32B-Instruct	23.30	21.41	14.60	19.77	74.82	44.01	84.77	67.87
Qwen2.5-VL-72B-Instruct	25.80	25.13	18.60	23.18	72.72	48.94	**87.68**	69.78
Skywork-R1V3-38B	**36.07**	26.11	11.33	24.50	56.87	**65.86**	69.04	63.92
GLM-4.1V-Thinking-9B	27.77	**31.54**	18.24	**25.85**	**76.93**	64.15	84.28	**75.12**

formance on many tabular benchmarks involving complex tasks, their success may not be grounded on a robust and reliable long-context table understanding ability, i.e., they indeed can solve certain complex problems in existing benchmarks, but they still can not reliably perceive and understand the information of each cell in lengthy input tables. This also shows that existing models may rely on dataset-specific correlations or shortcuts to obtain better benchmark results or there may be serious data leakage problems for some models.

4 Experiments on Downstream Tasks

4.1 Data Synthesis Method

Based on above findings, we further explore **whether enhancing NIAT capabilities can fundamentally improve models' overall table understanding ability**. If the long-context table understanding truly lays the foundation of more advanced high-level table-related capacities, its improvements should also benefit the performance on downstream tasks. To this end, we propose a strong2weak data synthesis method to create NIAT fine-tuning data. Specifically, we collect Markdown tables from the training split, and generate various NIAT queries and corresponding responses using GPT-4o as instruction-tuning data. For cell-locating task, we adopt the same prompt templates in the NIAT benchmark and automatically construct queries based on randomly selected table cells. For cell-lookup task, to avoid overlapping with lookup questions in the test data of downstream benchmarks, we synthesize queiries of 6 more difficult lookup tasks to improve data diversity, which are listed in the Appendix C. These tasks present greater challenges than the simplest lookup questions, requiring LLMs to develop a more thorough understanding of table structures and header information. For instance, cell-retrieval task requires LLMs to perform lookup operations within the whole input table and identify all locations of a given cell content.

Besides, previous tabular LLMs often directly transform existing tabular datasets into fine-tuning data with short answers as target responses [31, 52], which can not provide enough supervised information for LLMs and may also lead to overfitting towards dataset-specific shortcuts. Therefore, we meticulously craft demonstrations and utilize GPT-4o with in-context learning to synthesize detailed chain-of-thought (CoT) reasoning processes as target responses, which aims at instructing weak LLMs to progressively understand tabular data and complete NIAT tasks. In the end, 6K instruction-tuning data were constructed for cell-locating and cell-lookup tasks, respectively.

https://doi.org/10.52202/085713-4121

Table 3: Performance results on downstream tasks. The best results in different model categories are highlighted in **bold**, and + *NIAT* denotes models further fine-tuned on our synthetic data.

Method	NIAT Task			Downstream Task				
	Cell-Locating	Cell-Lookup	Ave. Acc	WTQ	TabFact	HiTab	TABMWP	Ave. Acc
Open-source LLMs								
GLM4-9B-Chat	**6.74**	49.17	27.96	**45.60**	43.50	25.90	47.51	40.63
MiniCPM3-4B	1.18	**58.13**	**29.66**	40.58	**62.88**	24.43	**69.39**	**49.32**
InternLM2.5-7B-Chat	0.96	35.26	18.11	34.76	33.00	18.78	54.83	35.34
Yi-1.5-9B-Chat	4.41	49.05	26.73	34.00	45.40	**32.70**	34.93	36.76
Tabular LLMs								
StructLLM	1.82	42.72	22.27	31.08	29.45	15.74	39.37	28.91
TableLLM	-	-	-	35.86	31.47	17.96	28.91	28.55
TableGPT2	**8.84**	**73.87**	**41.36**	**60.01**	**61.17**	**36.04**	**56.19**	**53.35**
Ours								
Qwen2.5-7B-Instruct	9.46	47.60	28.53	52.90	70.00	30.50	54.42	51.96
+NIAT	**10.84**	58.41	34.63	60.28	61.28	**62.28**	**72.39**	64.06
Llama3.1-8B-Instruct	6.16	65.74	35.95	49.90	62.80	26.10	54.78	48.40
+NIAT	8.38	**66.46**	**37.42**	**67.43**	**78.57**	49.41	66.15	**65.39**
Reasoning LLMs and GPT-4o								
GPT-4o	26.02	68.30	47.16	83.50	65.80	39.10	**84.38**	68.19
QwQ-32B	35.42	49.74	42.58	78.26	**75.23**	61.35	53.76	67.15
DeepSeek-R1	**65.91**	**80.99**	**73.45**	**84.21**	66.44	**66.86**	64.74	**70.56**

4.2 Experimental Setup

Baselines. Apart from open-source LLMs, reasoning LLMs like DeepSeek-R1, and close-sourced GPT-4o, we mainly compare our method with tabular LLMs that are fine-tuned with table instruction tuning data, including TableLLM [53] with 80K synthesized training data of downsteam tabular tasks, TableGPT2 [33] with 86B tokens for continuous pre-training (CPT) and 2.36M in-house data for table instruction tuning, and StructLLM fine-tuned with 1.1M samples of structured knowledge grounding tasks. We use the released model weights for inference, and fine-tune Llama3.1-8B-Instruct and Qwen2.5-7B-Instruct with 12K synthesized data as our methods.

Benchmarks. We evaluate model performance on 4 downstream benchmarks. WTQ [19] and HiTab [40] are TQA benchmarks based on flat tables and hierarchical tables, which contain lookup questions and reasoning questions that require additional aggregation operations. TabFact [20] is a TFV benchmark that requires models to verify whether a textual hypothesis holds based on the given evidence in a table. TABMWP [54] is a table-based math word problem benchmark that evaluates mathematical reasoning ability over structured tabular data. Tables are represented in the Markdown format for WTQ and TabFact, and HTML format for HiTab and TABMWP to maintain hierarchical table structures. We use exact match accuracy as the evaluation metric and report model performance under the zero-shot setting. **Complete implementation details are provided in Appendix C.**

4.3 Results and Analysis

Table 3 compares model performance on NIAT tasks and downstream tabular tasks. We can find that there is a substantial performance gap between NIAT tasks and more complex downstream tasks, demonstrating their weaknesses in the robust long-context table understanding ability. The proposed NIAT benchmark helps mitigate the data leakage issue among rapidly evolving LLMs, where training data and even test data in downstream tabular benchmarks could be excessively utilized for fine-tuning to achieve better benchmark results. We believe that a model with truly robust table understanding ability should not only perform well on traditional downstream tasks, but also maintain strong capability in simple but foundamental NIAT tasks, which is the foundation for reliable LLM-based table applications, especially in scenarios involving lengthy tables.

Compared to vanilla models, fine-tuning with our synthesized 12K NIAT data brings substantial performance enhancements for Qwen2.5-7B-Instruct and Llama3.1-8B-Instruct, improving the average

 https://doi.org/10.52202/085713-4121

accuracy in both NIAT tasks and downstream tasks by 3.78% and 14.55%. Notably, our synthesized data does not contain training data of downstream benchmarks and solely focuses on enhancing the basic long-context understanding ability towards table structures and content. However, our fine-tuned models achieve better overall performance than baseline LLMs of similar scale, even surpassing tabular LLMs like TableGPT2 that use much more training data. This validates the effectiveness and efficiency of our synthesized data, and verifies our initial assumption that NIAT capabilities constitute the very foundation of more advanced table-related tasks. Figure 4 also intuitively demonstrates the improvement of our synthesized data on every cell position within cropped tables. **Due to space limitation, more experimental results such as ablation experiment are provided in Appendix D.**

4.4 Ablation Study

Table 4: Ablation results of different tasks further fine-tuned on *Llama3.1-8B-Instruct*. The best results are highlighted in **bold**. WTQ, TabFact, and HiTab are the three downstream datasets selected for fine-tuning with chain-of-thought (CoT) reasoning processes generated by GPT-4o on foundation LLMs, as listed in the table.

Model	NIAT							WTQ	TabFact	HiTab	TABMWP
	8	12	16	20	24	28	32				
Llama3.1-8B-Instruct	16.88	**13.56**	**12.94**	**8.98**	**6.34**	**7.25**	**3.83**	49.90	62.80	26.10	54.78
+ Cell-Locating & Cell-Lookup	**20.10**	12.75	10.21	5.51	4.51	4.88	2.49	**67.43**	**78.57**	**49.41**	66.15
+ Cell-Locating	19.76	13.49	12.06	7.86	5.31	6.22	3.07	67.33	67.45	33.44	70.50
+ Cell-Lookup	18.85	13.13	12.15	8.36	6.22	5.83	3.21	59.00	53.50	35.00	69.44
+ 4 downstream datasets	16.28	11.12	11.41	6.11	5.92	5.40	2.17	64.78	78.13	48.22	**81.79**

We conduct ablation experiments to investigate the impact of different training tasks systematically. The experimental results are presented in Table 4. We compare the foundation model fully trained on our proposed mixed synthetic data to variants trained on individual sub-tasks, as well as to models fine-tuned on the training splits of WTQ, TabFact, and HiTab, which serve as oracle experiments representing the performance ceiling.

Fine-tuning on our synthesized training data achieves comparable performance to variants fine-tuned on three downstream datasets. Surprisingly, when fine-tuned on our mixed synthetic data, *Llama3.1-8B-Instruct* achieves comparable accuracy on the TabFact dataset and even outperforms other models on the remaining downstream tasks. For variants trained solely on Cell-Locating and Cell-Lookup tasks (+ Cell-Locating and + Cell-Lookup), removing NIAT training data significantly drops performance across all four downstream tasks.

5 Conclusion

This paper makes the first systematic exploration of long-context table understanding problem that has been overlooked by prior work, together with a new benchmark NEEDLEINATABLE, which covers tables of diverse structures, formats and sizes, and can serve as a valuable testbed for evaluating models' underlying fine-grained understanding and perception of individual cells within tabular context. On this basis, we conduct extensive evaluation of existing LLMs and MLLMs, revealing that current large models lack truly robust long-context table understanding ability, which could influence the reliability of LLM-based table applications. Furthermore, we propose a data synthesis method for enhancing LLMs' basic NIAT capabilities, and demonstrate its improvements in models' overall table understanding abilities, outperforming strong baselines including recent tabular LLMs.

6 Acknowldagment

This work was supported by the National Natural Science Foundation of China (Nos. 62472419, 62472420)

References

[1] Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. Longbench: A bilingual, multitask benchmark for long context understanding. In Lun-Wei Ku, Andre Martins, and Vivek

Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 3119–3137. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.172. URL https://doi.org/10.18653/v1/2024.acl-long.172.

[2] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. Agentbench: Evaluating llms as agents. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=zAdUB0aCTQ.

[3] Jun Zhao, Can Zu, Hao Xu, Yi Lu, Wei He, Yiwen Ding, Tao Gui, Qi Zhang, and Xuanjing Huang. Longagent: Scaling language models to 128k context through multi-agent collaboration. *CoRR*, abs/2402.11550, 2024. doi: 10.48550/ARXIV.2402.11550. URL https://doi.org/10.48550/arXiv.2402.11550.

[4] Yushi Bai, Jiajie Zhang, Xin Lv, Linzhi Zheng, Siqi Zhu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. Longwriter: Unleashing 10,000+ word generation from long context llms. *CoRR*, abs/2408.07055, 2024. doi: 10.48550/ARXIV.2408.07055. URL https://doi.org/10.48550/arXiv.2408.07055.

[5] OpenAI. Gpt-4 and gpt-4 turbo - documentation. 2024. URL https://platform.openai.com/docs/models/gpt-4-and-gpt-4-turbo. Accessed on January 23, 2024.

[6] Anthropic. claude-3-family. 2024. URL https://www.anthropic.com/news/claude-3-family. Accessed on March 4, 2024.

[7] GeminiTeam. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. 2024. URL https://storage.googleapis.com/deepmind-media/gemini/gemini_v1_5_report.pdf. Accessed on March 17, 2024.

[8] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

[9] An Yang, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoyan Huang, Jiandong Jiang, Jianhong Tu, Jianwei Zhang, Jingren Zhou, Junyang Lin, Kai Dang, Kexin Yang, Le Yu, Mei Li, Minmin Sun, Qin Zhu, Rui Men, Tao He, Weijia Xu, Wenbiao Yin, Wenyuan Yu, Xiafei Qiu, Xingzhang Ren, Xinlong Yang, Yong Li, Zhiying Xu, and Zipeng Zhang. Qwen2.5-1m technical report, 2025. URL https://arxiv.org/abs/2501.15383.

[10] Greg Kamradt. Needle in a haystack - pressure testing llms. https://github.com/gkamradt/LLMTest_NeedleInAHaystack, 2023. URL https://github.com/gkamradt/LLMTest_NeedleInAHaystack.

[11] Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Khai Hao, Xu Han, Zhen Leng Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. ınftybench: Extending long context evaluation beyond 100k tokens. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 15262–15277. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.814. URL https://doi.org/10.18653/v1/2024.acl-long.814.

[12] Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. Longbench v2: Towards deeper understanding and reasoning on realistic long-context multitasks, 2024. URL https://arxiv.org/abs/2412.15204.

[13] Mo Li, Songyang Zhang, Yunxin Liu, and Kai Chen. Needlebench: Can llms do retrieval and reasoning in 1 million context window? *CoRR*, abs/2407.11963, 2024. doi: 10.48550/ARXIV.2407.11963. URL https://doi.org/10.48550/arXiv.2407.11963.

[14] Julian Schnitzler, Xanh Ho, Jiahao Huang, Florian Boudin, Saku Sugawara, and Akiko Aizawa. Morehopqa: More than multi-hop reasoning. *CoRR*, abs/2406.13397, 2024. doi: 10.48550/ARXIV.2406.13397. URL https://doi.org/10.48550/arXiv.2406.13397.

[15] Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: what's the real context size of your long-context language models? *CoRR*, abs/2404.06654, 2024. doi: 10.48550/ARXIV.2404.06654. URL https://doi.org/10.48550/arXiv.2404.06654.

[16] Yucheng Ruan, Xiang Lan, Jingying Ma, Yizhi Dong, Kai He, and Mengling Feng. Language modeling on tabular data: A survey of foundations, techniques and evolution, 2024. URL https://arxiv.org/abs/2408.10548.

[17] Xi Fang, Weijie Xu, Fiona Anting Tan, Jiani Zhang, Ziqing Hu, Yanjun Qi, Scott Nickleach, Diego Socolinsky, Srinivasan Sengamedu, and Christos Faloutsos. Large language models(llms) on tabular data: Prediction, generation, and understanding – a survey, 2024. URL https://arxiv.org/abs/2402.17944.

[18] Weizheng Lu, Jing Zhang, Ju Fan, Zihao Fu, Yueguo Chen, and Xiaoyong Du. Large language model for table processing: a survey. *Frontiers Comput. Sci.*, 19(2):192350, 2025. doi: 10.1007/S11704-024-40763-6. URL https://doi.org/10.1007/s11704-024-40763-6.

[19] Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1470–1480. The Association for Computer Linguistics, 2015. doi: 10.3115/V1/P15-1142. URL https://doi.org/10.3115/v1/p15-1142.

[20] 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. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=rkeJRhNYDH.

[21] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023. URL https://arxiv.org/abs/2104.09864.

[22] Himashi Peiris, Munawar Hayat, Zhaolin Chen, Gary Egan, and Mehrtash Harandi. Hybrid window attention based transformer architecture for brain tumor segmentation, 2022. URL https://arxiv.org/abs/2209.07704.

[23] Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. Yarn: Efficient context window extension of large language models, 2023. URL https://arxiv.org/abs/2309.00071.

[24] Xiaoran Liu, Hang Yan, Shuo Zhang, Chenxin An, Xipeng Qiu, and Dahua Lin. Scaling laws of rope-based extrapolation, 2024. URL https://arxiv.org/abs/2310.05209.

[25] Zijia Zhao, Haoyu Lu, Yuqi Huo, Yifan Du, Tongtian Yue, Longteng Guo, Bingning Wang, Weipeng Chen, and Jing Liu. Needle in A video haystack: A scalable synthetic framework for benchmarking video mllms. *CoRR*, abs/2406.09367, 2024. doi: 10.48550/ARXIV.2406.09367. URL https://doi.org/10.48550/arXiv.2406.09367.

[26] Hengyi Wang, Haizhou Shi, Shiwei Tan, Weiyi Qin, Wenyuan Wang, Tunyu Zhang, Akshay Nambi, Tanuja Ganu, and Hao Wang. Multimodal needle in a haystack: Benchmarking long-context capability of multimodal large language models. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3221–3241, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. URL https://aclanthology.org/2025.naacl-long.166/.

[27] Chau Pham, Simeng Sun, and Mohit Iyyer. Suri: Multi-constraint instruction following in long-form text generation. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 1722–1753. Association for Computational Linguistics, 2024. URL https://aclanthology.org/2024.findings-emnlp.94.

[28] Haoran Que, Feiyu Duan, Liqun He, Yutao Mou, Wangchunshu Zhou, Jiaheng Liu, Wenge Rong, Zekun Moore Wang, Jian Yang, Ge Zhang, Junran Peng, Zhaoxiang Zhang, Songyang Zhang, and Kai Chen. Hellobench: Evaluating long text generation capabilities of large language models. *CoRR*, abs/2409.16191, 2024. doi: 10.48550/ARXIV.2409.16191. URL https://doi.org/10.48550/arXiv.2409.16191.

[29] Wenhu Chen. Large language models are few(1)-shot table reasoners. In Andreas Vlachos and Isabelle Augenstein, editors, *Findings of the Association for Computational Linguistics: EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, pages 1090–1100. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-EACL.83. URL https://doi.org/10.18653/v1/2023.findings-eacl.83.

[30] 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, 2024. URL https://arxiv.org/abs/2305.13062.

[31] Tianshu Zhang, Xiang Yue, Yifei Li, and Huan Sun. Tablellama: Towards open large generalist models for tables. In Kevin Duh, Helena Gómez-Adorno, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 6024–6044. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.NAACL-LONG.335. URL https://doi.org/10.18653/v1/2024.naacl-long.335.

[32] Qian Liu, Fan Zhou, Zhengbao Jiang, Longxu Dou, and Min Lin. From zero to hero: Examining the power of symbolic tasks in instruction tuning. *CoRR*, abs/2304.07995, 2023. doi: 10.48550/ARXIV.2304.07995. URL https://doi.org/10.48550/arXiv.2304.07995.

[33] Aofeng Su, Aowen Wang, Chao Ye, Chen Zhou, Ga Zhang, Gang Chen, Guangcheng Zhu, Haobo Wang, Haokai Xu, Hao Chen, Haoze Li, Haoxuan Lan, Jiaming Tian, Jing Yuan, Junbo Zhao, Junlin Zhou, Kaizhe Shou, Liangyu Zha, Lin Long, Liyao Li, Pengzuo Wu, Qi Zhang, Qingyi Huang, Saisai Yang, Tao Zhang, Wentao Ye, Wufang Zhu, Xiaomeng Hu, Xijun Gu, Xinjie Sun, Xiang Li, Yuhang Yang, and Zhiqing Xiao. Tablegpt2: A large multimodal model with tabular data integration, 2024. URL https://arxiv.org/abs/2411.02059.

[34] Mingyu Zheng, Xinwei Feng, Qingyi Si, Qiaoqiao She, Zheng Lin, Wenbin Jiang, and Weiping Wang. Multimodal table understanding. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9102–9124, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.493. URL https://aclanthology.org/2024.acl-long.493/.

[35] Xianjie Wu, Jian Yang, Linzheng Chai, Ge Zhang, Jiaheng Liu, Xinrun Du, Di Liang, Daixin Shu, Xianfu Cheng, Tianzhen Sun, Guanglin Niu, Tongliang Li, and Zhoujun Li. Tablebench: A comprehensive and complex benchmark for table question answering, 2024. URL https://arxiv.org/abs/2408.09174.

[36] Wei Zhou, Mohsen Mesgar, Heike Adel, and Annemarie Friedrich. FREB-TQA: A fine-grained robustness evaluation benchmark for table question answering. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2479–2497, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.137. URL https://aclanthology.org/2024.naacl-long.137/.

[37] Justin Payan, Swaroop Mishra, Mukul Singh, Carina Negreanu, Christian Poelitz, Chitta Baral, Subhro Roy, Rasika Chakravarthy, Benjamin Van Durme, and Elnaz Nouri. InstructExcel: A benchmark for natural language instruction in excel. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4026–4043, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.265. URL https://aclanthology.org/2023.findings-emnlp.265/.

[38] Wei Zhou, Mohsen Mesgar, Heike Adel, and Annemarie Friedrich. FREB-TQA: A fine-grained robustness evaluation benchmark for table question answering. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2479–2497, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.137. URL https://aclanthology.org/2024.naacl-long.137/.

[39] Mingyu Zheng, Yang Hao, Wenbin Jiang, Zheng Lin, Yajuan Lyu, QiaoQiao She, and Weiping Wang. IM-TQA: A Chinese table question answering dataset with implicit and multi-type table structures. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5074–5094, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.278. URL https://aclanthology.org/2023.acl-long.278/.

[40] 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. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 1094–1110. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.ACL-LONG.78. URL https://doi.org/10.18653/v1/2022.acl-long.78.

[41] Yannis Katsis, Saneem Chemmengath, Vishwajeet Kumar, Samarth Bharadwaj, Mustafa Canim, Michael Glass, Alfio Gliozzo, Feifei Pan, Jaydeep Sen, Karthik Sankaranarayanan, and Soumen Chakrabarti. AIT-QA: Question answering dataset over complex tables in the airline industry. In Anastassia Loukina, Rashmi Gangadharaiah, and Bonan Min, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track*, pages 305–314, Hybrid: Seattle, Washington + Online, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-industry.34. URL https://aclanthology.org/2022.naacl-industry.34/.

[42] Ananya Singha, José Cambronero, Sumit Gulwani, Vu Le, and Chris Parnin. Tabular representation, noisy operators, and impacts on table structure understanding tasks in llms, 2023. URL https://arxiv.org/abs/2310.10358.

[43] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2023. URL https://arxiv.org/abs/2203.11171.

[44] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

[45] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, Lixin Gu, Xuehui Wang, Qingyun Li, Yimin Ren, Zixuan Chen, Jiapeng Luo, Jiahao Wang, Tan Jiang, Bo Wang, Conghui He, Botian Shi, Xingcheng Zhang, Han Lv, Yi Wang, Wenqi Shao, Pei Chu, Zhongying Tu, Tong He, Zhiyong Wu, Huipeng Deng, Jiaye Ge, Kai Chen, Kaipeng Zhang, Limin Wang, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahua Lin, Yu Qiao, Jifeng Dai, and Wenhai Wang. Expanding

performance boundaries of open-source multimodal models with model, data, and test-time scaling, 2025. URL https://arxiv.org/abs/2412.05271.

[46] Alex Zhuang, Ge Zhang, Tianyu Zheng, Xinrun Du, Junjie Wang, Weiming Ren, Stephen W. Huang, Jie Fu, Xiang Yue, and Wenhu Chen. Structlm: Towards building generalist models for structured knowledge grounding, 2024. URL https://arxiv.org/abs/2402.16671.

[47] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, and et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

[48] Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025. URL https://qwenlm.github.io/blog/qwq-32b/.

[49] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

[50] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

[51] Shengqiong Wu, Hao Fei, Xiangtai Li, Jiayi Ji, Hanwang Zhang, Tat-Seng Chua, and Shuicheng Yan. Towards semantic equivalence of tokenization in multimodal llm. *ICLR*, 2025.

[52] Naihao Deng and Rada Mihalcea. Rethinking table instruction tuning, 2025. URL https://arxiv.org/abs/2501.14693.

[53] Xiaokang Zhang, Jing Zhang, Zeyao Ma, Yang Li, Bohan Zhang, Guanlin Li, Zijun Yao, Kangli Xu, Jinchang Zhou, Daniel Zhang-Li, et al. Tablellm: Enabling tabular data manipulation by llms in real office usage scenarios. *arXiv preprint arXiv:2403.19318*, 2024.

[54] Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=DHyHRBwJUTN.

[55] Zengzhi Wang, Fan Zhou, Xuefeng Li, and Pengfei Liu. Octothinker: Mid-training incentivizes reinforcement learning scaling. *CoRR*, abs/2506.20512, 2025. doi: 10.48550/ARXIV.2506.20512. URL https://doi.org/10.48550/arXiv.2506.20512.

[56] Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *CoRR*, abs/2503.20783, 2025. doi: 10.48550/ARXIV.2503.20783. URL https://doi.org/10.48550/arXiv.2503.20783.

NeurIPS Paper Checklist

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

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

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

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

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

IMPORTANT, please:

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

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract provides a concise summary of the key findings and experiment results. The introduction in Sec. 1 outlines the research questions and objectives.

   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: The paper discusses the limitations of the work performed by the authors in detail in Appendix. A.2

   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: [No]

   Justification: This paper did not contain theory assumptions and proofs.

   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: The paper provides a detailed description of the experimental data setup and hyperparameter settings in Section 4 and Appendix B and C.

   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.

 https://doi.org/10.52202/085713-4121

- 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: The datasets, baseline methods, and models used in the paper are fully open-source and available on Hugging Face. The complete code and data is provided in n the supplementory materials.

   Guidelines:

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

6. **Experimental setting/details**

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

Answer: [Yes]

Justification: The paper provides a detailed description of the experimental data setup and in Sec. 4.2, and more information is provided in Appendxi C.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [No]

Justification: Error bars are not reported because it would be too computationally expensive, but we calculate the average performance of the LLMs and shown in 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: In Shown in Appendxi and C, we report the GPUs we used, the memory, and detailed training and inference information.

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.

 https://doi.org/10.52202/085713-4121

- 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: Appendix B and C

   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: The discussion of the ethics and impact can be consulted in Appendix. A.2. We are open and transparent throughout the study and do not design for human subjects. privacy data bias, or other issues.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [Yes]

    Justification: The safeguard of our paper is dissussed in Appendix A.

    Guidelines:

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

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: This article uses assets reasonably in compliance with the license, and the assets used are cited in the article.

    Guidelines:

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

13. **New assets**

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

    Answer: [Yes]

    Justification: The documentation is provided in the supplementory files.

    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: The paper does not involve crowdsourcing nor research with human subjects.

 https://doi.org/10.52202/085713-4121

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: The paper does not involve crowdsourcing nor research with human subjects.

    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: LLMs are only used for editing grammar.

    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 Discussion

A.1 Limitations

Although this work presents a comprehensive exploration of the long-context table understanding problem, certain limitations could be alleviated by future research. First, the proposed benchmark focuses on a single table in English. The long-context understanding in multi-table scenarios with broader language coverage deserves future explorations. Second, real-world applications such as document understanding could also require processing the hybrid content of tabular, textual, and image data. Therefore, it is valuable to explore the long-context understanding ability towards data consisting of multiple modalities. Third, we build the NIAT benchmark based on tables from public datasets to save cost, which primarily contain medium-sized tables. Future work could collect larger tables that span diverse structures to provide a longer context for long-context table understanding, especially tables with more than thousands of rows.

A.2 Ethical Considerations

The proposed NIAT benchmark is constructed based on the academic datasets like WTQ and TabFact, which are free and open datasets for research use with licenses like MIT License[5] or CC-BY-SA4.0 License[6]. The resulting synthetic training data is also a free and open resource for the community to study long-structured table understanding. Considering that LLMs may generate harmful content, we used Llama3.1-70B-instruct to conduct LLM-as-a-judge to check the content of synthetic data and the generated data by API-available models, and we also randomly sampled a part of the data for manual checking and did not observe unsafe data in the synthetic samples. Thus, the authors foresee no ethical concerns with the research in this paper.

A.3 Different fine-tuning benefits between Qwen2.5 and Llama3.1

One notable finding from our experiments is the significant disparity in performance gains on the cell-locating task after NIAT fine-tuning (Figure 4). Specifically, Qwen2.5-7B demonstrated substantially greater improvement than Llama3.1-8B. We hypothesize this phenomenon is attributable to the models' differing pre-existing capabilities, stemming from two potential factors. First, the pre-training regimen of the models plays a crucial role. As detailed in its technical report [49], the Qwen2.5 series underwent specialized post-training on structured tabular data. This inherent familiarity with table structures may provide a more effective foundation for our CoT-based fine-tuning. Consequently, the fine-tuning process could more readily "activate" and enhance this latent ability, leading to a more pronounced boost in table structure comprehension. Second, existing research in reinforcement learning [55, 56] has indicated that the Qwen2.5 series possesses strong reasoning capabilities, even without instruction tuning. A superior innate reasoning capacity could make the model more adept at acquiring complex, multi-step procedural tasks, such as the cell-locating process of interpreting a table row-by-row and then pinpointing a cell by its column index. In contrast, the Llama3.1 series might require additional mid-training to achieve a comparable level of performance on such tasks.

A.4

B Details of NIAT Benchmark

B.1 Data Statistics

To construct a benchmark for long-context understanding of tabular data at a low cost, we leveraged existing open-source datasets to sample tabular data. Table 5 represents the statistics results of sampled tables, we calculate the average length of tokenized tabular data. Figure 5 shows the distribution of input length of the NIAT benchmark.

[5] https://opensource.org/license/mit/

[6] https://creativecommons.org/licenses/by-sa/4.0/deed.en

https://doi.org/10.52202/085713-4121

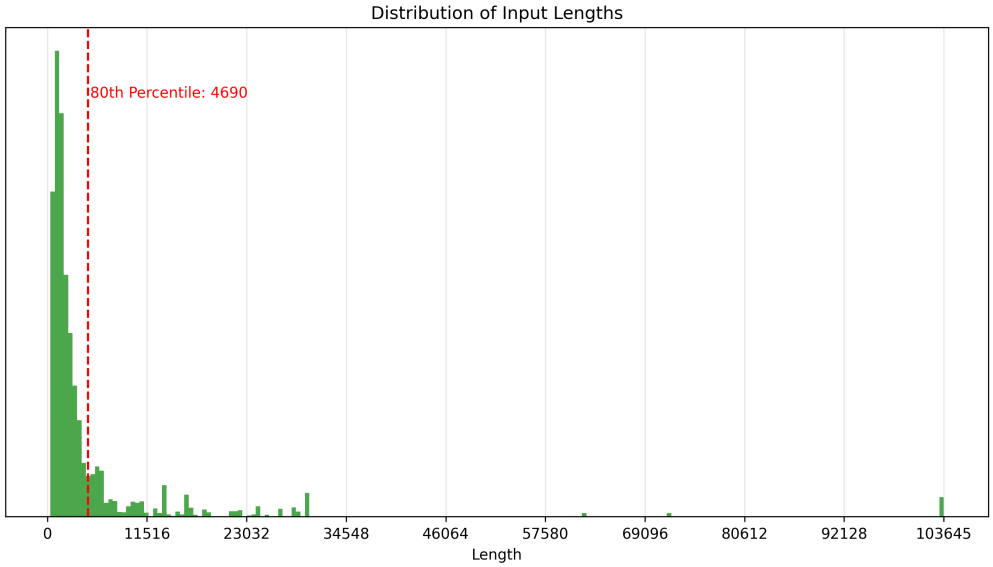

Figure 5: The distribution of input lengths for our proposed NIAT benchmark. The tokenizer of Llama3.1-8B-Instruct is adopted to calculate the token length.

Although the largest tables reach a length of 100k tokens, the average length of tables sampled from public dataset is not long enough. We anticipate that future researchers will build upon this work to develop more comprehensive benchmarks for long-table understanding tasks with longer tabular text.

Table Types	Format	Table Num.	Avg. Row Num.	Avg. Col Num.	Avg. Length	Max Lengh
Horizontal	**HTML**	325	6.43	26.78	1831.27	29791
	Mardown	358	6.38	25.77	981.92	16012
Flat	**HTML**	362	26.32	6.51	5226.04	103430
	Markdown	362	26.24	6.38	843.05	13203
Hierarchical	**HTML**	1086	19.81	8.87	878.39	5339
	Markdown	1086	19.81	8.87	1481	9420
Total	-	-	18.56	11.46	1594.32	103430

Table 5: Detailed table statistics of sampled tables of NIAT benchmark. The **Avg. Length** and **Max Length** represent the average and maximum token lengths of encoded tabular text for specific table categories. We use the tokenizer of Llama3.1-8B-Instruct to calculate the token length.

B.2 Prompt Templates of Evaluating LLMs

For text-LLMs and MLLMs, we adopt unified prompt templates to evaluate the performance of them. For Cell-Locating the prompt templates are shown in Figure 9, for Cell-Lookup task, the adopted template are provided in Figure 10.

C Implementation Details

C.1 Implementation Details

To investigate the relationship between the fundamental table structure comprehension capabilities of LLMs and their performance in downstream table understanding tasks, we selected four popular datasets spanning various table understanding tasks. These datasets include **Table Question Answering** (HiTab, WTQ), **Tabular Numerical Reasoning** (TABMWP), and **Table Fact Verification** (TabFact). Each language model was instructed to generate a chain-of-thought reasoning process before providing its final answer. To evaluate performance, we calculated accuracy based on the last 30 characters of the generated content for open-source LLMs, which corresponds to the model's final answer.

We provide an example of a Cell-locating question and answer pair, along with the chain-of-thought (CoT) reasoning process, in Figure 11.

C.2 Training Details

To efficiently fine-tune *Qwen2.5-7B-Instruct* and *Llama3.1-8B-Instruct* on our synthetic data, we employ the NVIDIA Megatron framework. To avoid out-of-memory errors caused by long-context input sequences, we configure tensor parallelism (tp) to 8 and model pipeline (pp) to 4. Each model is fully fine-tuned on the mixed data for 2 epochs.

C.3 Evaluation Details

For the Data Mixing Strategy, we randomly sample 8,400 NIAT training instances and 6,497 table-lookup question-answer pairs, combining them into the training dataset. The longest sample in our training data reaches 92k tokens. **For evaluating all foundation models and the further fine-tuned models**, we adopt the original generation configurations of the models and utilize the vLLM framework for deployment on a machine equipped with 8 NVIDIA A100-80G GPUs, as the input sequences can be extremely long.

C.4 Details of Cell-Lookup Synthesis Tasks

Locate single cell Given a row header and a column header, LLMs are tasked with identifying the cell value at the intersection of these two headers. The uniqueness of the ground truth answer is strictly enforced.
Synthetic Data Generation Prompt Template: f"In the table above, what is the element located in the cell at the intersection of the row header «row header» and the column header «column header»?"

Retrieve the Nth cell based on row header Counting ability is crucial for LLMs when interpreting tabular data. In this task, given a row header (a specific cell value in the first column), the LLM is required to return the value of the Nth cell in the corresponding row.
Synthetic Data Generation Prompt Template: f"What is the Nth cell value with header row «row header»?"

Retrieve the Nth cell based on column header : Similarly, LLMs are expected to locate the Nth cell value within a column specified by a given column header. Both tasks require LLMs to retrieve target cell contents simultaneously based on natural language queries and row/column IDs.
Synthetic Data Generation Prompt Template: f"What is the Nth cell value with header col «col header»?"

Retrieve all location IDs of a certain key In some cases, cell values may be duplicated in a large table. This task requires identifying the row and column IDs (location IDs) of all cells that contain the target value.
Synthetic Data Generation Prompt Template: f"In the table above, how many cells contain the value «cell to ask»?"

Cell value counting In this task, LLMs are asked to count the number of cells that contain a specific value, further testing their ability to process and summarize tabular data.
Synthetic Data Generation Prompt Template: f'How many cell contains the value «random element»? Please provide all row IDs and column IDs of all cells contain "«random element»".'

Table navigation Understanding the relative positions of two different cells in a table is a challenging task for LLMs. Given a specific base cell position and row/column offsets, LLMs are required to navigate the two-dimensional table by calculating the target position IDs and retrieving the corresponding cell values.
Synthetic Data Generation Prompt Template:"You are provided with a two-dimensional table and need to locate the content of a specific cell. The following information is given: Base Position: Row index: «base row» Column index: «base col» Relative Position: Row offset: «relative row» Column offset: «relative col» Search Instructions: If «relative row» > 0, move downwards from the base row index. If «relative row» < 0, move upwards from the base row index. If «relative col» > 0, move rightwards from the base column index. If «relative col» < 0, move leftwards from the base column index. "Calculate the new target position («new row», «new col») using these offsets from the base position, and return the content of the cell located at this new position"

 https://doi.org/10.52202/085713-4121

	Format	Avg. Queries Num.
Horizontal	**HTML**	317.56
	Markdown	376.33
Flat	**HTML**	290.57
	Markdown	372.67
Hierarchical	**HTML**	909
	Markdown	1041

Table 6: The average number of queries of question-based NIAT. We prompt GPT-4o to generate simple questions to extract cells of a given table and check the correctness of

Prompt Template of 6 Synthesized Cell-Lookup tasks We randomly sampled to examples of Cell-Lookup training data synthesized by GPT-4o are shown in Figure 12 (Retrieve the Nth cell based on column header) and 13 (Table navigation).

D Additional Experiments

D.1 Results on Cropped Tables

Table 8 presents the detailed performance metrics of various MLLMs. In Table 7, we compare LLMs fine-tuned on our proposed NIAT synthetic data against mainstream LLMs and TableLLMs on Cell-Locating tasks with cropped tables. This task demands a deep structural understanding of two-dimensional tabular data, so we evaluated model performance across varying context window sizes. Consistent with expectations, longer context windows improve accuracy in Cell-Locating tasks.

The **Qwen2.5 series**dominates performance, likely due to its specialized training on tabular data (as noted in its technical report). Notably, **Qwen2.5-7B-Instruct-1M**, a long-context variant, significantly outperforms its standard counterpart (**Qwen2.5-7B-Instruct**), suggesting benefits from structured pre-training and post-training data.

Among **Thinker LLMs**, DeepSeek-R1 achieves state-of-the-art results, leveraging its Long-CoT capability to accurately retrieve target cells in large 32×32 tables using row/column identifiers. This underscores the potential of test-time scaling techniques for processing long-structured tabular data. Surprisingly, Qwen2.5-7B-Coder-Instruct (with only 7B parameters) excels on NIAT tasks, likely due to: 1) Structural alignment between tabular data and code (both rely on separators for semantic organization). 2) Extensive pre-training on markdown, which may have included markdown tables, priming it for Cell-Locating NIAT tasks.

Multi-Modal LLMs (MLLMs), which process table images directly, more closely mirroring human tabular understanding, demonstrate superior performance in Cell-Locating tasks. Notably, recent state-of-the-art models like Qwen2.5-VL-7B-Instruct and MiniCPM-V-2.6.8B significantly outperform earlier approaches (e.g., Llava-1.5-7B), while they still underperform GPT-4o with pure text input.

As illustrated in Figure 6, both mainstream **MLLMs** and **text-LLMs** exhibit the **lost-in-the-middle-tables** phenomenon—their performance degrades monotonically as table size increases. For MLLMs, this limitation arises because visual encoders struggle to accurately parse table structures when rows and columns expand beyond 32×32, leading to unreliable target cell retrieval. Nevertheless, recent MLLMs (e.g., MiniCPM-V-2.6-8B) show marked improvements in Cell-Locating tasks, suggesting advances in structural understanding.

D.2 Case Study

We provide two cases of the model further fine-tuned on our synthesized data on **Cell-Locating** and **Cell-Lookup**, respectively in Figure 7 and 8.

https://doi.org/10.52202/085713-4121

Table 7: Performance results on Cropped Tables of mainstream text-LLMs, TableLLM, MLLMs, and foundation models fine-tuned on our synthetic data. The best results are highlighted in **bold**, and + *ours* denotes models further fine-tuned on our proposed synthetic data.

Model	Cell-Locating on Cropped Tables						
	8	**12**	**16**	**20**	**24**	**28**	**32**
Open Source LLMs							
GLM4-9b-chat	9.9	10.88	10.26	8.43	4.91	8.65	4.29
GLM4-9b-chat-1M	5.63	6.39	5.94	5.43	3.84	3.99	2.57
MiniCPM3-4B	1.98	4.35	5.73	5.75	4.28	9.51	5.02
InternLM2.5-7B	4.9	6.25	4.84	5.23	3.24	3.85	2.25
Mistral-7B-v0.3	8.54	10.65	9.17	6.2	3.47	5.49	5.81
Yi-1.5-9B-16K	12.81	8.98	5.55	5.67	3.32	4.53	2.12
Qwen2.5-Instruct							
Qwen2.5-7B-Instruct	26.35	18.98	15.91	12.83	7.55	9.72	2.9
Qwen2.5-7B-Instruct-1M	32.08	24.44	20.6	13.8	9.99	12.13	4.93
Qwen2.5-14B-Instruct	28.96	19.81	19.69	15.93	14.38	17.54	9.31
Qwen2.5-14B-Instruct-1M	**39.69**	**33.84**	**33.83**	**26.7**	**20.98**	**25.54**	16.54
Qwen2.5-7B-Instruct + ours	37.60	30.83	25.76	17.67	18.66	18.24	10.85
Llama3.1-Instruct							
Llama-3.1-8B-Instruct	16.88	**13.56**	**12.94**	**8.98**	**6.34**	**7.25**	**3.83**
Llama-3.1-8B-Instruct + ours	**20.1**	12.75	10.21	5.51	4.51	4.88	2.49
TableLLM							
TableLLM	-	-	-	-	-	-	-
StructLLM	5.62	5.23	4.29	4.02	2.07	3.26	0.92
TableGPT2	21.15	15.46	12.99	10.12	8.34	9.57	3.4
Thinker LLM and Coder LLM							
GPT-4o	50.21	33.36	30.89	24.15	27.54	26.01	17.32
QwQ-32B	51.15	45.23	43.91	14.57	6.01	6.32	6.56
DeepSeek-R1	**73.33**	**64.63**	**64.60**	**41.87**	**38.07**	**54.06**	**51.17**
Qwen2.5-7B-Coder	65.42	54.86	44.90	28.42	17.79	29.88	28.67
MLLMs							
GPT-4o(Text)	41.67	18.34	19.94	9.75	8.84	18.46	3.92
GLM-4V-9B	2.80	2.61	4.30	1.39	1.43	0.89	0.09
InternVL-2.5-8B	28.52	16.64	12.60	10.24	10.96	13.18	5.09
Llava-1.5-7B	2.68	4.31	1.19	0.20	1.26	0.39	0.30
Table-Llava-7B	1.67	2.37	2.65	2.12	2.97	1.82	2.01
Llava-1.6-7B	6.60	4.03	4.24	5.57	4.11	4.33	0.26
MiniCPM-V-2.6-8B	43.51	30.57	15.57	11.74	10.01	12.88	3.94
Phi-3.5-Vision-Instruct-4.2B	4.25	2.84	5.92	2.68	4.68	1.56	0.82
Phi-4-multimodal-instruct	6.04	8.86	8.84	5.77	6.12	8.69	3.68
Qwen2-VL-7B-Instruct	7.27	8.72	8.62	9.64	8.19	8.07	2.30
Qwen2.5-VL-3B-Instruct	15.66	13.60	10.95	10.53	9.70	8.11	2.67
Qwen2.5-VL-7B-Instruct	19.80	14.64	11.98	10.67	5.29	9.59	3.02
Llama-3.2-11B-Vision-Instruct	10.63	8.96	9.89	6.98	4.34	5.30	3.03

Table 8: Performance results of mainstream MLLMs on cropped tables. The best results are highlighted in **bold**.

Model	Cell-Locating NIAT task on cropped tables							
	8*8	**12*12**	**16*16**	**20*20**	**24*24**	**28*28**	**32*32**	**Ave.**
Multi-Modal LLMs								
DeepSeek-VL2-tiny-3B	2.8	4.08	2.46	4.27	3.47	2.49	1	2.94
GLM-4V-9B	2.8	2.61	4.3	1.39	1.43	0.89	0.09	1.93
Llava-1.5-7B	2.68	4.31	1.19	0.2	1.26	0.39	0.3	1.48
Table-Llava-7B	1.67	2.37	2.65	2.12	2.97	1.82	2.01	2.23
Llava-1.6-7B	6.6	4.03	4.24	5.57	4.11	4.33	0.26	4.16
Phi-3.5-Vision-Instruct-4.2B	4.25	2.84	5.92	2.68	4.68	1.56	0.82	3.25
Phi-4-multimodal-instruct-5.6B	6.04	8.86	8.84	5.77	6.12	8.69	3.68	6.86
Qwen2-VL-7B-Instruct	7.27	8.72	8.62	9.64	8.19	8.07	2.3	7.54
Qwen2.5-VL-3B-Instruct	15.66	13.6	10.95	10.53	9.7	8.11	2.67	10.17
Qwen2.5-VL-7B-Instruct	19.8	14.64	11.98	10.67	5.29	9.59	3.02	10.71
Llama-3.2-11B-Vision-Instruct	10.63	8.96	9.89	6.98	4.34	5.3	3.03	7.02
MiniCPM-V-2.6-8B	**43.51**	**30.57**	15.57	**11.74**	10.01	12.88	3.94	**18.32**
InternVL-2.5-8B	28.52	16.64	12.6	10.24	10.96	13.18	**5.09**	13.89
GPT-4o	41.67	18.34	**19.94**	9.75	8.84	**18.46**	3.92	17.27

https://doi.org/10.52202/085713-4121

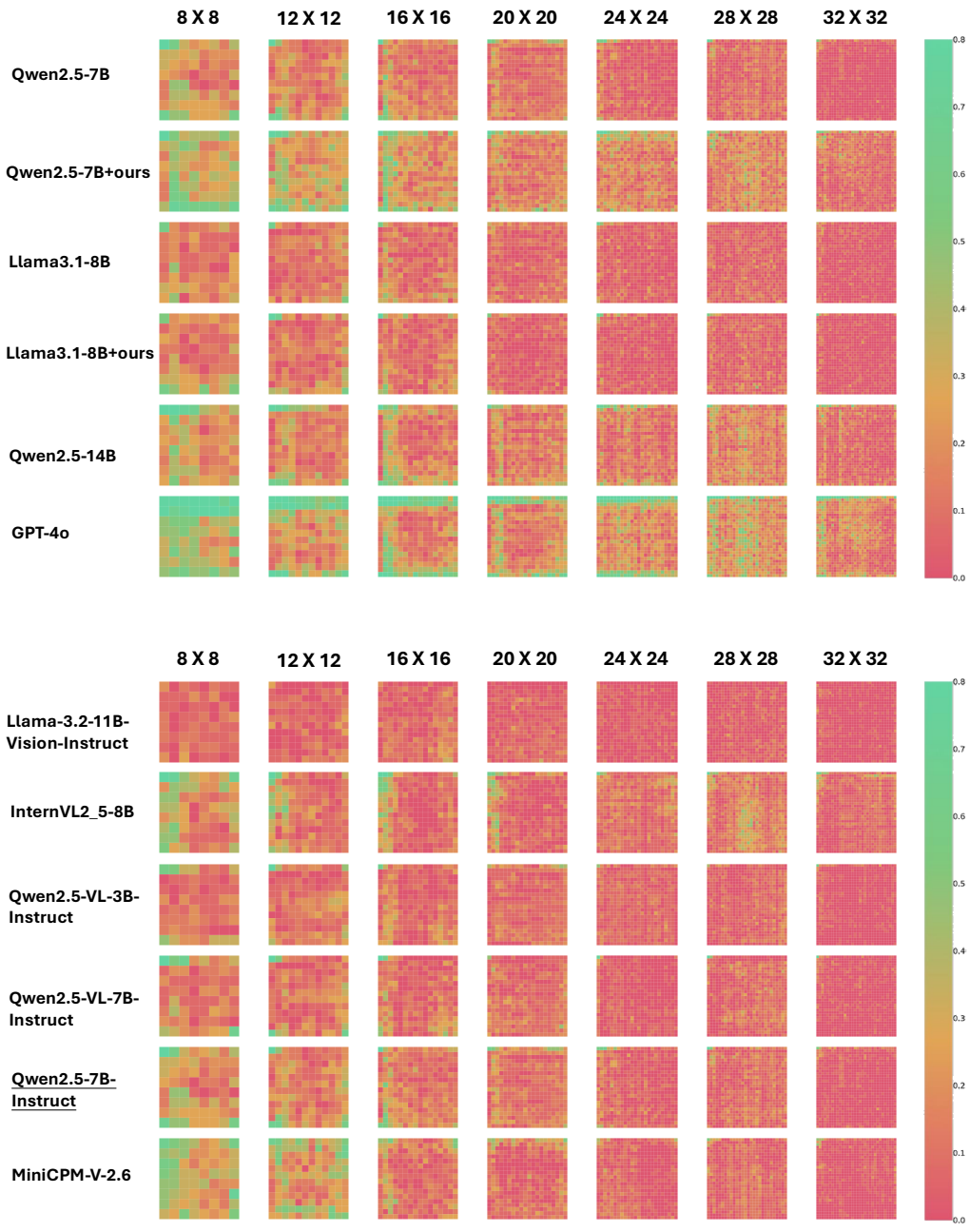

Figure 6: The figure presents a NEEDLEINATABLE evaluation performance comparison on cropped tables between MLLMs and text-LLMs.

An example of finetuned Llama3.1-8B-Instruct output:

To determine the value at the specified cell, we will follow these steps:

1. **Identify the Row:**
   - We are looking for the row with a row ID of 8. In the provided table, the rows are numbered from 1 to 8, corresponding to each province listed.
   - The row with a row ID of 8 corresponds to the province "Seville".

2. **Identify the Column:**
   - We are looking for the column with a column ID of 1. The columns are labeled as follows:
     - Column 1: Province
     - Column 2: Capital
     - Column 3: Population
     - Column 4: Density
     - Column 5: Municipalities
     - Column 6: Legal districts

3. **Locate the Cell:**
   - With the identified row (Seville, row ID 8) and column (Province, column ID 1), we locate the cell at the intersection of these two.
   - The cell at row ID 8, column ID 1 contains the value "Seville".

4. **Output the Result:**
   - The cell value located at the specified position is "Seville".

Therefore, the final answer is:

```
{"row_ID": 8, "column_ID": 1, "cell_value": "Seville"}
```

This matches the provided ground truth answer, confirming that the value in the specified cell is "Seville".

Figure 7: A case response from Llama3.1-8B-Instruct, further fine-tuned on synthesized data, demonstrating its performance on the Cell-Locating task.

https://doi.org/10.52202/085713-4121

An example of finetuned Llama3.1-8B-Instruct output:

To determine the opponent in Week 16, we need to carefully examine the table provided. Let's go through a step-by-step reasoning process:

1. **Identify Week 16:**
   - We need to locate Week 16 in the table. The weeks are listed sequentially from Week 1 to Week 17. We look for the entry that corresponds to Week 16.

2. **Locate the Date for Week 16:**
   - In the table, the date for Week 16 is December 23, 2007.

3. **Find the Opponent for Week 16:**
   - Once we have identified the correct week, we look at the "Opponent" column for that row. For Week 16, the opponent listed is "Philadelphia Eagles."

4. **Verify the Details:**
   - To ensure accuracy, we check the other details in the same row to confirm consistency. The "Game Site" is "Louisiana Superdome," the "TV" channel is "Fox," and the "Result/Score" is "L 38-23," which matches the pattern of losses (L) followed by a score in the format of "Score - Score."

5. **Conclusion:**
   - Based on the information extracted from the table, the opponent in Week 16 is confirmed to be the Philadelphia Eagles.

Final Answer:

```json
{{
    "answer": "Philadelphia Eagles"
}}
```

Figure 8: A case response from Llama3.1-8B-Instruct, further fine-tuned on synthesized data, demonstrating its performance on the Cell-Lookup task.

Read the table image carefully and retrieve the value from a specified cell, based on a given row ID and a column ID.

Let's first think step by step and then output a final answer in the following JSON format:

```json
{{
    "row_ID": "m",
    "column_ID": "n",
    "cell_value": "<Corresponding Cell Value>"
}}
```

Please note:
- Row and column IDs start from 1.
- The header row is considered as the first row, with a row_ID of 1, so ensure you include it when locating the target row.
- If the target cell is empty, the cell_value should be an empty string ("").

Now, retrieve the cell value at the position where the row_ID is {} and the column_ID is {}.

Your Response:

Figure 9: Prompt template for evaluating Cell-Locating Task.

```
Read the table image carefully and answer the following question.
Let's first think step by step and then output a final answer in the JSON
 format:
```json
{{
    "answer": "<the final answer to the question>"
}}
```
Question:
{}

Your Response:
```

Figure 10: Prompt template for evaluating Cell-Locating Task.

https://doi.org/10.52202/085713-4121

Question:

Retrieve the cell value at the position where the row_ID is 5 and the column_ID is 26.

Answer:

To find the target cell value in the given table, where the row_ID is 5 and the column_ID is 26, we should follow these detailed step-by-step instructions:

1. Identify the Header Row:
- The header row is the first row in the table and provides the names for each column. In the provided table, this is the first row of data:

| Name(s) | Council | Core County | Other Counties | 2006 Tot | 2011 Tot | % change Tot | Tot area | Tot dens | 2006 Mun | 2011 Mun | % change Mun | Mun area | Mun dens | 2009 elect | 2006 Sub | 2011 Sub | Mun dens | 2011 Tot | 2006 Mun | Other Counties | 2011 Sub | Tot area | % change Mun | Council | Core County | Mun area | Name(s) | 2009 elect | % change Tot | 2006 Sub | 2011 Mun |

2. Identify Row 5 (not skipping the header row):
- Based on the given instructions, the header should be counted as row 1. Thus, row_ID 5 corresponds to the fifth row in the table, including the header row.
- Looking at the provided table, the fifth row is:

| Waterford | City | Waterford | Kilkenny | 49,213 | 51,519 | 4.7 | 44 | 1,171 | 45,748 | 46,732 | 2.2 | 38.35Z| 1,219 | 26,598 | 3,465 | 4,787 | 1,219 | 51,519 | 45,748 | Kilkenny | 4,787 | 44 | 2.2 | City | Waterford | 38.35 | Waterford | 26,598 | 4.7 | 3,465 | 46,732 |

3. Identify Column 26:
- Each vertical group of data is identified as a column. Starting from the first column, we count until we reach the 26th column as specified.
- According to the header, the columns are ordered as follows:
  1. Name(s)
  2. Council
  3. Core County
  4. Other Counties
  5. 2006 Tot
  6. 2011 Tot
  7. % change Tot
  8. Tot area
  9. Tot dens
  10. 2006 Mun
  11. 2011 Mun
  12. % change Mun
  13. Mun area
  14. Mun dens
  15. 2009 elect
  16. 2006 Sub
  17. 2011 Sub
  18. Mun dens
  19. 2011 Tot
  20. 2006 Mun
  21. Other Counties
  22. 2011 Sub
  23. Tot area
  24. % change Mun
  25. Council
  26. **Core County <- Target Cell**

4. Cell Location and Determination:
- In the fifth row, column 26 corresponds to the "Core County" column.
- The cell at the intersection of row 5 and column 26 in this row contains the value: "Core County."

Thus, following this step-by-step process, the value located at row_ID 5 and column_ID 26 is **Core County**.

Carefully review the table, question, and provided ground truth answer.

Table: **<table_str>**

Question: In the table above, what is the element located in the cell at the intersection of the row header 12 and the column header game?

Ground truth Answer: 12
Generate a detailed, step-by-step reasoning process to accurately answer the question. Conclude with the final answer.

GPT-4o Answer:
To determine the element located in the specified cell, follow these steps:

1. **Understand the Table Structure:**
   - The table is organized with columns labeled as: `game`, `date`, `team`, `score`, `high points`, `high rebounds`, `high assists`, `location attendance`, and `record`.
   - Each row corresponds to a particular game's data, with the rows numbered from 1 to 13.

2. **Identify the Cell to Examine:**
   - The question asks for the element at the intersection of the row header 12 and the column header `game`.
   - The row header refers to the game's sequence or identifier within this particular dataset; the row number is essentially the order of games played.

3. **Locate Row 12:**
   - Go to the row labeled with the number 12 in the table. This corresponds to all the data associated with the 12th game played.

4. **Find Column Header `game`:**
   - The `game` column is the first column in the table. This column indicates the game number or identifier for that particular set of data.

5. **Extract the Information:**
   - At the intersection of row 12 and the column `game`, locate the number, which indicates the sequence or position of that game.

6. **Interpret the Table Data:**
   - If we review the table, we can see that in row 12 under the column `game`, the cell contains the number "12". This means that the 12th row corresponds to the 12th game played.

Finally, using this step-by-step examination, the element located in the cell at the intersection of the row header 12 and the column header `game` is indeed:

Final Answer: 12

Figure 12: An example of the chain-of-thought (CoT) reasoning process for synthesized Cell-lookup tasks.

https://doi.org/10.52202/085713-4121

Carefully review the table, question, and provided ground truth answer.

Table: <table_str>

Question:
You are provided with a two-dimensional table and need to locate the content of a specific cell. The following information is given:

Base Position:

Row index: 32
Column index: 4
Relative Position:

Row offset: -17
Column offset: 0
Search Instructions:

If relative_row > 0, move downwards from the base row index.
If relative_row < 0, move upwards from the base row index.
If relative_col > 0, move rightwards from the base column index.
If relative_col < 0, move leftwards from the base column index.
Calculate the new target position (new_row, new_col) using these offsets from the base position, and return the content of the cell located at this new position

Ground truth Answer: One of the original 19 parishes.
Generate a detailed, step-by-step reasoning process to accurately answer the question. Conclude with the final answer.

GPT-4o Answer:
To determine the content of a specific cell using the given information, we need to follow the search instructions carefully. Let's go through each step:

1. **Identify the Base Position:**
   - The base row index is 32.
   - The base column index is 4.

2. **Understand the Relative Position:**
   - The row offset is -17, meaning we need to move 17 rows upwards from the base row.
   - The column offset is 0, meaning there is no movement in the columns, and we stay in the same column as the base column.

3. **Apply the Row Offset:**
   - Starting from the base row index of 32, move 17 rows upwards.
   - Calculate the new row index: \(32 - 17 = 15\).

4. **Apply the Column Offset:**
   - The base column index is 4 and the column offset is 0, so the new column index remains 4.

5. **Locate the New Position in the Table:**
   - The new row index is 15.
   - The new column index is 4.

6. **Find the Content of the Cell at the New Position:**
   - Check the table at row index 15 and column index 4.
   - According to the table, this corresponds to the "Origin" column for the 16th parish in the list, which is East Baton Rouge Parish since the table index starts from zero.

7. **Verify and Extract the Content:**
   - The content at row index 15 and column index 4 is "One of the original 19 parishes."

8. **Conclusion:**
   - The content of the cell located at the new position is "One of the original 19 parishes."

Thus, the final answer is:

"One of the original 19 parishes."