# TableDreamer: Progressive and Weakness-guided Data Synthesis from Scratch for Table Instruction Tuning

Anonymous ACL submission

## Abstract

Despite the commendable progress of recent LLM-based data synthesis methods, they faces two limitations in generating table instruction tuning data. First, they can not thoroughly explore the vast input space of table understanding tasks, leading to limited data diversity. Second, they ignore the weaknesses in table understanding ability of the target LLM and blindly pursue the increase of data quantity, resulting in suboptimal data efficiency. In this paper, we introduce a progressive and weakness-guided data synthesis framework tailored for table instruction tuning, named TableDreamer, to mitigate the above issues. Specifically, we first synthesize diverse tables and related instructions as seed data, and then perform an iterative exploration of the input space under the guidance of the newly identified weakness data, which eventually serve as the final training data for fine-tuning the target LLM. Extensive experiments on 10 tabular benchmarks demonstrate the effectiveness of the proposed framework, which boosts the average accuracy of Llama3.1-8B-instruct by 11.62% (49.07% $\rightarrow$ 60.69%) with 27K GPT-4o synthetic data and outperforms state-of-the-art data synthesis baselines which use more training data.

## 1 Introduction

Table understanding technique aims to enable models to automatically comprehend tables and complete various table-related tasks (Lu et al., 2025; Shigarov, 2023). With the recent advancement of large language models (LLMs), the dominant paradigm for table understanding has shifted to instruction tuning general LLMs with tabular task data, leading to the rise of powerful Tabular LLMs (Zhang et al., 2024a; Li et al., 2023).

In early work on tabular LLMs, instruction-tuning samples were manually collected by human annotators or converted from public datasets using fixed instruction templates. However, the reusing of existing datasets often leads to poor task and data diversity, while human annotation also faces the challenge of prohibitively expensive cost. Therefore, researchers turned to employ LLMs to generate table instruction tuning data. For instance, Zhang et al. (2024b) uses GPT-3.5 to generate questions based on benchmark tables, which serve as the training data for fine-tuning CodeLlama (Rozière et al., 2024). The resulting TableLLM model outperforms general LLMs on several tabular benchmarks, demonstrating the potential of synthetic data in table instruction tuning.

Although existing data synthesis approaches have achieved commendable performance, they still face two limitations in generating table instruction tuning data. First, **existing data synthesis methods are unable to fully explore the vast input space composed of input tables and instructions, leading to limited data diversity.** On the one hand, general data generation methods like Self-Instruct (Wang et al., 2023) primarily focus on generating unstructured text data, and they did not adequately consider the unique characteristics of structured tables (e.g., diverse table structures, different table formats). As a result, they tend to produce simple tables and instructions of limited tabular tasks. On the other hand, existing studies on tabular LLMs only explore how to synthesize more instructions based on directly available tables from public datasets to improve instruction diversity, but they lack the ability to synthesize more diversified tabular data, which also limits the diversity of the final table instruction tuning data.

Second, **existing data synthesis methods ignore the LLM's weaknesses in table understanding ability, resulting in suboptimal efficiency of synthetic data.** The combination of the input table and the instruction allows us to easily create a large amount of table instruction tuning data, e.g., we can utilize an LLM to generate dozens of questions based on a single table. However, published

studies have indicated that merely pursuing an increase in the quantity of instruction tuning data does not necessarily yield performance improvement (Zhou et al., 2023; Si et al., 2023). Given the vast input space for table understanding tasks, it is more efficient to synthesize valuable data points that expose the deficiencies of the target LLM, rather than blindly increase the amount of synthetic data, which may result in both a waste of training resources and a decline in model performance.

To address these issues, we introduce a progressive and weakness-guided data synthesis framework for table instruction tuning, named **TableDreamer**, which can not only generate diverse tables and instructions from scratch, but can also continuously explore the input space under the guidance of newly identified weakness data to more effectively enhance the model performance. As illustrated in Figure 1, our framework consists of two stages. In stage 1, we first synthesize various table titles of different topics and subtopics, and then employ the LLM to create diverse tables. In stage 2, based on synthetic tables and tabular task descriptions, a group of seed data is generated and will undergo data evolution in three directions. The synthesized new samples are evaluated by LLM-as-a-judge to identify weakness-exposing data, which is used as the seed data for the next round of data evolution. This process can be iterated multiple times, with the accumulated weakness data serving as the final table instruction tuning data.

We compare TableDreamer with a series of data synthesis methods, general LLMs and tabular LLMs on 10 tabular benchmarks. Experimental results demonstrate the effectiveness of the proposed framework, which boosts the average accuracy of Llama3.1-8B-instruct by 11.62% (49.07% $\rightarrow$ 60.69%) with 27K GPT-4o synthetic data and outperforms the state-of-the-art data synthesis baseline by 8.46%. We also demonstrate the effectiveness of TableDreamer as data augmentation for the few-shot learning scenario, where only a small number of original training samples are available (e.g., 20 samples for each benchmark). Extensive ablation experiments are conducted to reveal the contributions of different components in the framework (e.g., the influence of weakness data selection and data evolution). We hope this work could establish a strong base for future research on the table instruction tuning data synthesis and help researchers improve models' table understanding ability especially with limited annotation budget.

We conclude our contributions as follows:

1) We introduce a data synthesis framework TableDreamer tailored for table instruction tuning with better data diversity and efficiency, mitigating the limitations of current approaches.

2) We construct and release 27K table instruction tuning data, which include diverse tables and instructions of a wide range of tabular tasks that the current open-source community lacks.

3) We make a systematic investigation of existing methods to show the effectiveness of TableDreamer, which outperforms strong baselines on 10 tabular benchmarks including recent tabular LLMs.

## 2 Related Work

### 2.1 Table Instruction Tuning

In addition to directly prompting LLMs to fulfill tabular tasks (Chen, 2023; Wang et al., 2024b; Lu et al., 2023a), researchers are increasingly dedicated to developing tabular LLMs with carefully constructed table instruction tuning data. TableLlama (Zhang et al., 2024a) collected 2.6M instruction-tuning pairs from 14 academic tabular datasets, and TableBenchLLM (Wu et al., 2024) even spent $12,000 US dollars on hiring annotators for answering labeling and quality checking. Besides, LLM-based data synthesis methods were also adopted to generate table instruction tuning data. TableGPT (Li et al., 2023) proposed a Synthesis-then-Augment framework which uses GPT-3.5 to generate instructions based on public tables and then performs data augmentations such as instruction paraphrasing for better data diversity. TableLLM (Zhang et al., 2024b) introduced a similar distant supervision approach which first synthesizes instructions and selects high-quality responses with the cross-way Validation of different reasoning methods. However, compared with other areas like code and math, data synthesis for table instruction tuning is still in infancy, with numerous issues deserving further exploration. In this paper, we introduce a novel data synthesis method, and also conduct a comprehensive investigation of relevant baselines, providing valuable insights about this emergent yet promising direction.

### 2.2 LLM-based Data Synthesis

The large amount of high-quality human-collected data has facilitated the development of deep learning in recent years. Nevertheless, purely depending on human data always involves a trade-off between

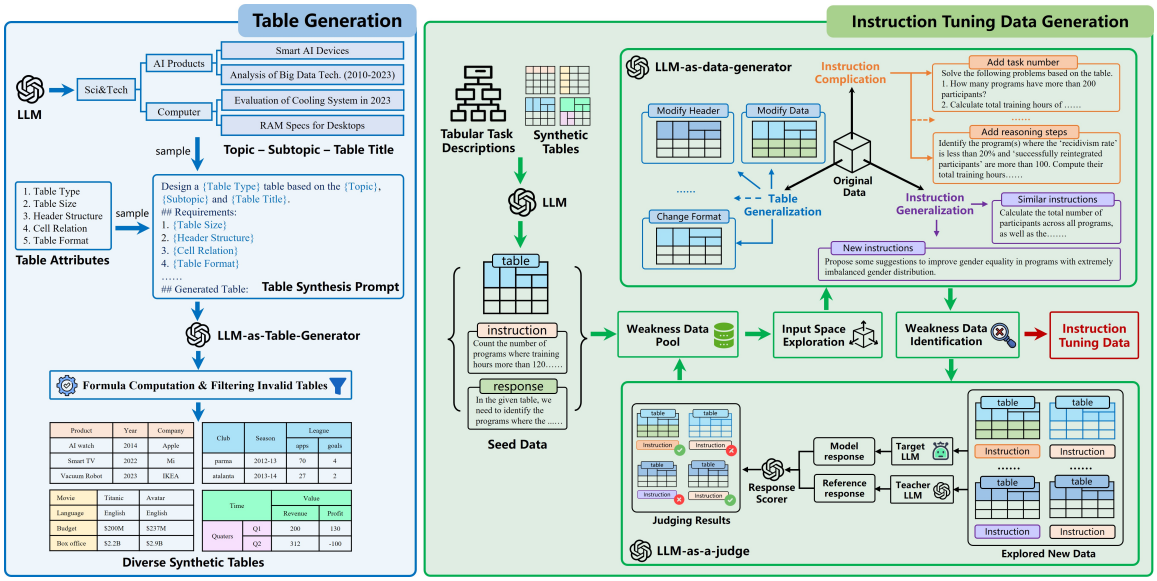


Figure 1: The overview of the proposed TableDreamer framework, which includes two stages. In stage 1, we first synthesize table titles based on different topics and subtopic, and then employ the LLM to generate diverse tables covering a wide range of key table attributes such as table structures and sizes. In stage 2, starting from a group of seed data, we perform an iterative exploration of the input space under the guidance of the newly discovered weakness data, which eventually serve as the table instruction tuning data.

**Instruction**

Filter out the challenges with a severity level of 4 or higher, translate them into French, and evaluate the effectiveness of the technologies mentioned in each challenge. Include your evaluation in the table along with the translated text.

**Table title**

Challenges Faced by the Global Apparel Industry Due to Supply Chain Disruptions, 2021

**Flat Table**

| Challenge | Region | Severity (1-5) | Technology Used | Long-term Strategy |
|---|---|---|---|---|
| Raw Material Shortage | North America | 4 | AI Forecasting | Diversification |
| Port Congestion | Asia | 5 | IoT Tracking | Port Expansion |
| Labor Shortages | Europe | 3 | Robotics | Workforce Training |
| Transportation Costs | South America | 4 | Route Optimization | Green Logistics |
| Exchange Rate Fluctuations | Oceania | 3 | ERP Systems | Financial Diversification |

Figure 2: Example of TableDreamer synthetic data. The synthetic table are clipped due to space limitation.

data quality and quantity due to factors such as costs or privacy issues (Long et al., 2024). Given the excellent ability to output human-like text, the advanced LLMs offer an alternative data source with synthetic data generation to mitigate drawbacks of human data. One of most prominent application of LLM-based data generation is to synthesize large-scale and diverse instruction tuning data in a cost effective way (Wang et al., 2023; Taori et al., 2023; Xu et al., 2023; Li et al., 2024a). Based on a handful human-created instructions as the initial seed data, Self-instruct (Wang et al., 2023) synthesizes new instructions by prompting an LLM with randomly selected instructions from the candidate pool as few-shot demonstrations. Magpie (Xu et al., 2024) leverages the autoregressive nature of LLMs and elicits instructions from fine-tuned LLMs by feeding them a pre-query chat template. Unlike textual tasks, table understanding tasks poses new challenges for LLM-based data synthesis due to the hybrid input of unstructured text and structured table. Unfortunately, existing approaches usually simplify the problem setting by ignoring the demand for synthesizing diverse tables and can only generate questions using public benchmark tables. By contrast, we take a step further and explore how to synthesize both tables and relevant instructions from scratch.

## 3 TableDreamer Framework

### 3.1 Problem Definition

Given a table $T$ including its metadata like the table title and a user instruction $Inst$ about the table, the table understanding problem requires the model $f(\cdot)$ to output a response $R$ that correctly complete the specified table-related tasks in the instruction, i.e., $R = f(T, Inst)$. The goal of the table instruction tuning data synthesis is to obtain a synthetic training dataset $D_{syn}$ of $N$ triples for fine-tuning LLMs, i.e., $D_{syn} = \{(Inst_i, T_i, R_i) \mid$

$i = 1, 2, \ldots, N\}$. Existing data synthesis methods often simplify the problem setting by assuming that tables are always directly available, and thus only focus on generating table-related instructions. By contrast, we retain the original setting and endeavor to synthesize diverse tables and instructions from scratch without relying on any public datasets.

### 3.2 Table Generation

Existing general data synthesis methods like self-instruct can not fully capture the complexity and diversity inherent in structured tabular data, leading to limited variety of synthesized tables. Therefore, we meticulously design a table synthesis prompt that fully considers the important table attributes. First of all, various **topics**, **subtopics** and corresponding **table titles** of different domains are elicited from an established LLM, which then serve as the guidance for generating table content of different domains. For example, given the topic 'Science and Technology' and the subtopic 'AI Applications', a viable table title could be 'Detailed Analysis of AI Integration in Auto. Vehicles, 2022'.

On this basis, we further incorporate key table attributes in the prompt to enhance the diversity of synthetic tables. (1) **table type**. We randomly sample one table type from three common candidates including flat tables, horizontal tables and hierarchical tables (Cheng et al., 2022; Liu et al., 2024; Gupta et al., 2020). (2) **table size**. We randomly choose the row number and the column number of the table within an appropriate range to create tables of various sizes. (3) **header structure**. For hierarchical tables with multi-level row headers and column headers, we randomly appoint the expected row header and column header structure from common combinations. For instance, a hierarchical table could have a 3-level column header and 2-level row header. (4) **cell relation**. There may be dependency relations between different table cells, e.g., in a business revenue table, the value of 'net profit' should be the difference between the 'revenue' and the 'cost'. Thus, we require the LLM to utilize markdown formulas to represent such relations in the target cells if necessary, which can be automatically extracted and computed by scripts to obtain the final results. (5) **table format**. We use the HTML format to represent the synthesized hierarchical tables in order to accurately reflect merged cells and hierarchical headers and the Markdown-style format to represent flat and horizontal tables.

Taking into account the above table attributes, we employ the LLM as a table generator to synthesize diverse tables, which are further processed to compute results of potential formulas and are filtered to remove invalid tables such as incomplete tables with missing cells.

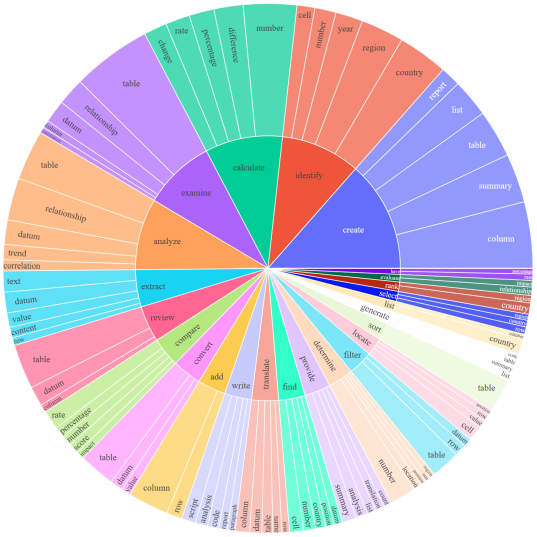


Figure 3: The top 25 most prevalent root verbs (the inner circle) and their top 5 direct nouns (the outer circle) in the synthetic instructions of TableDreamer-27K.

### 3.3 Instruction Tuning Data Generation

To provide a better foundation for instruction generation, we collect 20 different table understanding tasks and their descriptions from published studies (Ruan et al., 2024; Sui et al., 2024; Zhao et al., 2022, 2023b), such as table-based numerical reasoning, table structure understanding and so on. The full list of seed tabular tasks are shown in the Table 7. On the basis of synthetic tables and the task descriptions, we use the LLM to generate a set of task instructions which serve as the initial seed instructions for subsequent data evolution.

**Input Space Exploration.** To achieve a more comprehensive exploration of the input space, each sample in the seed data will undergo LLM-based data evolution in three directions respectively, thereby synthesizing more diverse data.

**Instruction Complication.** Inspired by previous instruction generation methods (Xu et al., 2023; Luo et al., 2024), we devise different evolution strategies to create more complex instructions based on the original table and the instruction. For instance, 'increasing the task number' will create new instructions that ask the LLM to complete multiple tabular tasks at once, and 'adding the rea-

soning steps' will generate multi-step problems. As LLMs' capabilities continue to improve, increasing the difficulty of input instructions assists us in uncovering the potential weaknesses in the table understanding ability of state-of-the-art LLMs, which enables us to enhance the model's capabilities in a more targeted manner.

**Instruction Generalization.** Considering that the instructions in the seed data are primarily limited to 20 predefined tabular tasks, we use the LLM to synthesize instructions of new tasks that are different from the original ones. We find that the LLM could create instructions of interesting and creative tabular tasks, e.g., analyzing the original table and providing recommendations, translating several columns into a new language and so on. Such task instructions are often not included in the public table-related datasets but can greatly improve the diversity of the instruction tuning data. In addition to generating new tabular task instructions, we also generate instructions that possess the same task type to the original one in order to improve model robustness towards instruction variations.

**Table Generalization.** Prior studies have found that current LLMs lack the robustness towards content and structural perturbations of input tables (Liu et al., 2024; Zhou et al., 2024; Singha et al., 2023). For instance, LLMs may experience significant performance fluctuations with changes in table formats and the order of rows and columns. This robustness is crucial for the practical application of tabular LLMs, as input tables from real-world users can vary greatly. To this end, we design table evolution strategies to create more table variations based on previously synthesized tables, e.g., changing the original table format, modifying the table header, reordering rows and columns and so on. This table generalization further improves the table diversity in the final training data which helps the model learn to maintain its performance despite these perturbations.

**Weakness Data Identification.** Although the input space exploration can generate a large variety of data, some of these samples may already be well-handled by the target LLM. Fine-tuning with such data could yield little performance improvements while consuming additional training resources. Thus, we utilize the LLM-as-a-judge (Zheng et al., 2023) to evaluate the response from the target LLM and identify samples where the target LLM underperforms. Concretely, given the response from the target LLM (e.g., Llama3.1-8B-instruct) and the reference response from a more powerful LLM (e.g., GPT-4o), an LLM rates the correctness of the model response on a 5 point likert scale, with lower scores indicating poorer performance. The samples with scores below 3 points are considered as weakness data, which will be used as the seed data for the next round of input space exploration and thus guide the overall data synthesis direction towards valuable data points that exposes the the model's deficiencies in table understanding ability. This iterative process between the input space exploration and the weakness detection can be performed multiple times, and the accumulated weakness data together with reference responses are used as the final table instruction tuning data.

### 3.4 Dataset Statistics and Cases

Unless otherwise specified, we use GPT-4o to synthesize tables, instructions and corresponding responses and select the Llama3.1-8B-instruct as the target LLM for weakness data detection. Starting from 3,272 seed data over 1,541 synthetic tables, we perform 2 rounds of iterative data synthesis process, ending in 27,083 instruction tuning data over 7,950 tables after filtering the invalid samples (e.g., failed data evolution results), which is denoted as TableDreamer-27K. Besides, we also replace GPT-4o with Llama3.1-70B-instruct to synthesize 27K training data, which is used for a fair comparison with other data synthesis baselines that we also reimplemented with Llama3.1-70B-instruct. Figure 2 demonstrates an example of the synthetic data. The diversity of the generated 27K instructions from GPT-4o is illustrated in Figure 3, where we plot the top 25 most prevalent root verbs and their top 5 direct nouns that appears at least 15 times. We can find that TableDreamer could generate diverse instructions and tables that encompass a broad range of tabular tasks and domains. More dataset statistics and examples are given in Appendix A. The detailed data evolution strategies and prompts are shown in Appendix B.1.

## 4 Experiments

### 4.1 Experimental Setup

**Benchmarks.** We select 9 public benchmarks: TABMWP (Lu et al., 2023b), WTQ (Pasupat and Liang, 2015), HiTab (Cheng et al., 2022), AIT-QA (Katsis et al., 2021), TabMCQ (Jauhar et al., 2016), TabFact (Chen et al., 2020), In-

foTabs (Gupta et al., 2020), FeTaQA (Nan et al., 2022) and QTSumm (Zhao et al., 2023a), which cover three tasks including table question answering (TQA), table-based fact verification (TFV) and table-to-text generation (T2T). The original question and the table in these benchmarks are serialized into a input text with various instruction templates and common table formats (e.g., HTML, Markdown, csv) for evaluating the LLM's robust table understanding ability. Besides, we also consider the synthetic benchmark from TableGPT (Li et al., 2023) which contains many unusual tabular tasks such as data imputation and thus can be used to evaluate the model's out-of-distribution (OOD) generalization ability. The summarization of all selected benchmarks is shown in Table 6.

**Evaluation Metrics.** For TQA, TFV and TableGPT benchmarks, the input instructions ask LLMs to output the final answer in the JSON format, which can be automatically extracted with regular expressions to compute exact match accuracy. For T2T benchmarks that are hard to accurately evaluate the correctness of the model response with automatic text generation metrics like BLEU (Papineni et al., 2002), we use LLM-as-a-judge evaluation, where GPT-4o-mini determines the accuracy of the model's responses based on the gold answer. The zero-shot setting is adopted for 9 public benchmarks except the TableGPT, as it provides test data in zero-shot and few-shot settings. Thus we report the average accuracy of two settings.

**Baselines.** We consider baselines of four genres. **(1) General LLMs** such as Llama3.1-8B-instruct (Grattafiori et al., 2024) and Mistral-7B-Instruct-v0.3 (Jiang et al., 2023). **(2) General Instruction Tuning Data Synthesis Methods** including the Self-Instruct (Wang et al., 2023), Dynasour (Yin et al., 2023), Evol-Instruct (Xu et al., 2023), GenQA (Chen et al., 2024) and Magpie (Xu et al., 2024). **(3) Data Synthesis Methods for Table Instruction Tuning**. We consider traditional tabular question generation methods including the OmniTab (Jiang et al., 2022), ReasTap (Zhao et al., 2022) and UCTR-ST (Li et al., 2024b), as well as recent LLM-based synthetic data from the TableGPT (Li et al., 2023) and the TableLLM (Zhang et al., 2024b), which use GPT-3.5 to generate instructions based on public tables. **(4) Tabular LLMs** including the TableBench-LLM (Wu et al., 2024) which is fine-tuned from Llama3.1-8B-base with 20K manually collected data, and the TableLLM (Zhang et al., 2024b) which is fine-tuned from CodeLlama-7B with 80K synthetic data. We also evaluate the powerful TableGPT2-7B (Su et al., 2024) that is fine-tuned from Qwen2.5-7B-instruct (Qwen et al., 2025) with 2.36M in-house query-table-output tuples. Implementation details are given in the Appendix B.2.

### 4.2 Results and Analysis

**Main Results.** *Performance of general LLMs.* As shown in Table 1, recent LLMs demonstrate varying proficiency in the table understanding ability, with the Llama3.1-8B-instruct exhibiting the best performance while models like Baichuan2-7B-Chat showing comparatively weaker performance. Their performance difference is likely due to the construction table-related fine-tuning data during the post-training stage. Moreover, we can find that small language model can also possess great table understanding ability, e.g., MiniCPM3-4B achieves better performance than large models like GLM4-9B-Chat, which opens up new possibilities for developing powerful and efficient tabular LLMs.

*Performance of tabular LLMs.* Compared with general LLMs, recent tabular LLMs such as TableBenchLLM exhibit surprisingly poorer performance on the benchmarks where they should be experts, even after being fine-tuned with the corresponding training dataset. Moreover, they can not effectively handle the unseen tabular tasks in the TableGPT benchmark. This shows that these tabular LLMs actually possess limited generalization ability especially out-of-distribution generalization, which is consistent with the findings from Deng and Mihalcea (2025). After a careful inspection, we find that this is due to the insufficient diversity in their instruction tuning data, e.g., the training data of TableBenchLLM only contain flat tables with a fixed Python dictionary-style table format and the instructions are primally limited to pre-defined tabular tasks. As a result, they can only perform well under the in-distribution setting, which highly constrains their application scenarios. By contrast, the TableGPT2 delivers the best overall results particularly on the TableGPT benchmark, showcasing the effectiveness of the 2.36M in-house high-quality training data, which includes not only public tabular datasets but also substantial synthetic data that are further refined by human annotators.

*Performance of data synthesis methods.* General instruction tuning data synthesis methods could be successfully extended to generate table instruction tuning data and bring considerable perfor-

| Method | # IFT Data | TQA | | | | | TFV | | T2T | | TableGPT | Ave. Acc. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | TABMWP | WTQ | HiTab | AIT-QA | TabMCQ | TabFact | InfoTabs | FeTaQA | QTSumm | | |
| *LLM* | | | | | | | | | | | | |
| Baichuan2-7B-Chat | - | 30.31 | 4.60 | 1.58 | 10.95 | 41.59 | 14.39 | 19.83 | 57.86 | 30.24 | 18.14 | 22.95 |
| GLM4-9B-Chat | - | 39.87 | 20.30 | 8.94 | 36.98 | 43.57 | 11.99 | 11.16 | 77.73 | 55.19 | 42.53 | 34.83 |
| DeepSeek-V2-Lite-16B-Chat | - | 49.01 | 15.65 | 7.67 | 29.94 | 63.45 | 29.75 | 37.11 | 64.20 | 35.81 | 29.36 | 36.19 |
| Phi3.5-mini-3.8B | - | 59.45 | 19.26 | 7.99 | 35.02 | 64.72 | 35.60 | 43.37 | 77.75 | 57.14 | 8.05 | 40.83 |
| MiniCPM3-4B | - | 50.53 | 34.06 | 20.93 | 55.34 | 72.98 | 28.09 | 42.33 | 68.55 | 42.39 | 40.79 | 45.60 |
| Mistral-7B-Instruct-v0.3 | - | 37.92 | 25.71 | 16.41 | 52.05 | 57.82 | 47.80 | 42.68 | 78.63 | 55.57 | 44.26 | 45.88 |
| InternLM2.5-7B-Chat | - | 50.22 | 32.59 | 13.51 | 51.46 | 36.25 | 45.07 | 47.33 | 81.43 | 62.86 | 39.52 | 46.02 |
| Yi-1.5-9B-Chat | - | 31.45 | 38.23 | 14.02 | 51.85 | 55.97 | 46.15 | 46.22 | 82.03 | 59.18 | 42.15 | 46.73 |
| Llama3.1-8B-Instruct | - | 53.39 | 36.53 | 11.35 | 43.63 | 75.31 | 53.87 | 48.94 | 78.98 | 66.98 | 21.68 | 49.07 |
| *General Instruction Tuning Data Synthesis Methods* | | | | | | | | | | | | |
| Self-Instruct | 100K | 46.68 | 28.98 | 13.77 | 48.92 | 80.27 | 52.92 | 45.07 | 81.13 | 53.48 | 43.13 | 49.44 |
| Dynasour | 132K | 49.71 | 28.59 | 20.11 | 43.44 | 59.66 | 50.70 | 41.01 | 57.56 | 42.57 | 13.40 | 40.67 |
| GenQA | 100K | 59.87 | 41.06 | 21.63 | **57.14** | 70.35 | 55.01 | 39.38 | 67.05 | 56.49 | 32.94 | 50.09 |
| Evol-Instruct | 100K | 54.61 | 31.83 | 12.37 | 45.20 | 73.27 | 54.12 | 45.61 | 83.02 | 62.77 | 42.55 | 50.54 |
| Magpie | 100K | 57.11 | 34.66 | 13.89 | 47.16 | 76.96 | 51.21 | 43.83 | 80.02 | **76.90** | 40.59 | 52.23 |
| *Table Instruction Tuning Data Synthesis Methods* | | | | | | | | | | | | |
| OmniTab | 100K | 17.53 | 22.67 | 18.84 | 35.02 | 50.63 | 16.37 | 3.14 | 5.04 | 4.82 | 18.38 | 19.24 |
| ReasTap | 100K | 11.22 | 19.54 | 9.96 | 20.54 | 48.49 | 15.66 | 5.70 | 7.14 | 4.92 | 20.67 | 16.38 |
| UCTR | 43K | 17.61 | 12.03 | 8.84 | 17.31 | 35.76 | 20.96 | 20.35 | 15.23 | 7.51 | 7.09 | 16.27 |
| TableGPT-syn-data | 66K | 25.21 | 16.13 | 9.13 | 24.26 | 47.52 | 19.70 | 25.29 | 46.03 | 36.64 | **47.23**$^{\dagger}$ | 29.71 |
| TableLLM-syn-data | 80K | 46.10 | 42.24$^{\dagger}$ | 13.92 | 39.72 | 25.46 | 29.24 | 31.31 | 79.08$^{\dagger}$ | 55.94 | 23.74 | 38.68 |
| *Tabular LLM* | | | | | | | | | | | | |
| TableBenchLLM (Llama3.1-8B) | 20K | 25.83 | 18.50$^{\dagger}$ | 12.31 | 29.74$^{\dagger}$ | 30.41 | 23.97$^{\dagger}$ | 17.33 | 48.27$^{\dagger}$ | 42.30 | 16.78 | 26.54 |
| TableLLM (CodeLlama-7B) | 80K | 43.11 | 37.86$^{\dagger}$ | 15.67 | 45.40 | 24.87 | 30.47 | 27.55 | 67.35$^{\dagger}$ | 37.66 | 15.14 | 34.51 |
| TableGPT2 (Qwen2.5-7B)$^{\ddagger}$ | 2.36M | 56.35 | 49.35 | 38.26 | 73.97 | 85.71 | 60.42 | 54.87 | 84.72 | 64.10 | 70.25 | 63.80 |
| *Ours* | | | | | | | | | | | | |
| **TableDreamer** (Llama3.1-70B-Instruct) | 27K | 60.57 | 42.47 | 17.25 | 56.75 | 82.99 | 57.32 | 49.98 | **84.67** | 75.12 | 33.03 | 56.02 |
| **TableDreamer** (GPT-4o) | 27K | **64.61** | **54.66** | **22.88** | 53.22 | **84.29** | **63.09** | **57.65** | 84.37 | 75.97 | 46.20 | **60.69** |

Table 1: Evaluation results on 10 tabular task benchmarks. † indicates that the model's fine-tuning data includes training samples from the corresponding dataset. ‡: we only list the performance of the TableGPT2 as its training data already contains these common benchmark datasets and the data volume also far exceeds others.

mance boost. For instance, fine-tuning with 100K Magpie synthetic data boosts the average accuracy from 49.07% to 52.23%. The traditional question generation approaches such as ReasTap obtain the worst performance because they can only generate simple table-related questions either through predefined question templates or by converting SQL queries. In comparison, although the LLM-based synthetic data from TableGPT and TableLLM can enhance the in-distribution model performance, e.g., fine-tuning with TableGPT synthetic data achieves the best result on the corresponding TableGPT benchmark, they still fail to improve the out-of-distribution table understanding capability on other benchmarks, which eventually yield a degenerated overall performance.

*Effectiveness of TableDreamer.* With Llama3.1-70B-instruct as the data synthesis LLM, TableDreamer improves the average accuracy of Llama3.1-8B-instruct by 6.95% (49.07% $\rightarrow$ 56.02%) ands surpasses other baselines without using any data from the public benchmarks, which validates the effectiveness of the proposed framework. The performance boost increases to 11.62% with the GPT-4o synthetic data due to better data quality. Notably, TableDreamer achieves a strong result (46.20%) on the TableGPT benchmark and is comparable to the model fine-tuned with TableGPT training data (47.23%), which showcases its effectiveness in improving the out-of-distribution table understanding capability. Moreover, TableDreamer obtains superior results with better data efficiency than data synthesis baselines, and is even competitive with the powerful TableGPT2 fine-tuned with 2.36M high-quality data.

**TableDreamer as Data Augmentation.** As shown in the Table 2, fine-tuning the model with very little labeled data offers limited improvement compared with the original performance, and adding TableDreamer synthetic data can bring a significant performance boost across various few-shot learning settings, which demonstrates its effectiveness in mitigating the scarcity of annotated table instruction tuning data.

**Ablation Study.** (1) *Ablation of synthetic tables.* We remove one type of tables and related instruction tuning data from the total data to analyze their influence, respectively. As presented in Table 3, removing each type of synthetic tables will cause negative effects due to the degenerated table diversity. We also observe the similar phenomenon in

| # Available Train Data of Each Dataset | TQA | | | TFV | | T2T | | Held-out | | Ave. Acc |
|---|---|---|---|---|---|---|---|---|---|---|
| | TABMWP | WTQ | HiTab | TabFact | InfoTabs | FeTaQA | QTSumm | AIT-QA | TabMCQ | |
| Llama3.1-8B-Instruct | 53.39 | 36.53 | 11.35 | 53.87 | 48.94 | 78.98 | 66.98 | 43.63 | 75.31 | 50.01 |
| 20 | 55.91 | 37.43 | 12.81 | 56.50 | 47.62 | 84.57 | 72.24 | 46.77 | 76.96 | 52.44 |
| w/ TableDreamer-27K | 64.88 | 56.23 | 24.17 | 60.46 | 53.38 | **83.87** | **76.62** | 53.42 | 83.28 | 59.94 |
| △ | 8.97 | 18.80 | 11.36 | 3.96 | 5.76 | -0.70 | 4.38 | 6.65 | 6.32 | 7.50 |
| 50 | 56.18 | 37.75 | 14.78 | 56.34 | 47.88 | 83.23 | 69.48 | 51.07 | 77.84 | 52.23 |
| w/ TableDreamer-27K | 70.89 | **56.37** | 26.90 | 60.68 | 47.22 | 83.37 | 74.95 | 61.64 | 83.86 | 60.05 |
| △ | 14.71 | 18.62 | 12.12 | 4.34 | -0.66 | 0.14 | 5.47 | 10.57 | 6.02 | 7.82 |
| 100 | 56.77 | 40.69 | 23.28 | 48.04 | 45.25 | 77.57 | 55.43 | 55.77 | 68.12 | 49.58 |
| w/ TableDreamer-27K | 70.96 | 54.37 | 36.04 | 57.07 | 46.00 | 81.38 | 73.28 | **64.18** | 84.15 | 59.87 |
| △ | 14.19 | 13.68 | 12.76 | 9.03 | 0.75 | 3.81 | 17.85 | 8.41 | 16.03 | 10.30 |
| 200 | 66.43 | 40.01 | 32.61 | 61.66 | 52.29 | 71.34 | 40.82 | 57.72 | 76.48 | 52.17 |
| w/ TableDreamer-27K | **76.59** | 50.59 | **41.94** | **63.33** | **57.44** | 78.43 | 72.26 | 59.29 | **84.64** | **62.94** |
| △ | 10.16 | 10.58 | 9.33 | 1.67 | 5.15 | 7.09 | 31.44 | 1.57 | 8.16 | 10.77 |

Table 2: Evaluation results under the few-shot learning setting, where only a limited number of training samples from 7 datasets (the first 7 columns) are available and TableDreamer data is used as additional training data.

| Mehtod | # IFT Data | TQA | TFV | T2T | TableGPT | Ave. Acc |
|---|---|---|---|---|---|---|
| Llama3.1-8B-Instruct | - | 44.04 | 51.41 | 72.98 | 21.68 | 49.07 |
| w/ TableDreamer | 27K | **55.93** | **60.37** | **80.17** | 46.20 | **60.69** |
| w/o Flat Tables | 17K | 51.41 | 52.02 | 74.85 | 40.59 | 55.13 |
| △ | | -4.53 | -8.36 | -5.33 | -5.61 | -5.56 |
| w/o Hier. Tables | 17K | 49.24 | 52.54 | 76.37 | **46.79** | 55.08 |
| △ | | -6.69 | -7.83 | -3.80 | +0.59 | -5.61 |
| w/o Hori. Tables | 18K | 54.58 | 51.40 | 78.07 | 45.38 | 57.72 |
| △ | | -1.35 | -8.98 | -2.11 | -0.82 | -2.97 |
| w/o Data Evolution | 3K | 47.71 | 49.50 | 71.28 | 38.68 | 51.88 |
| △ | | -8.22 | -10.87 | -8.90 | -7.52 | -8.82 |
| w/o Inst. Gene. | 18K | 52.32 | 51.77 | 78.26 | 40.89 | 56.26 |
| △ | | -3.61 | -8.60 | -1.91 | -5.31 | -4.44 |
| w/o Inst. Comp. | 18K | 50.83 | 51.25 | 73.95 | 39.82 | 54.44 |
| △ | | -5.10 | -9.12 | -6.22 | -6.38 | -6.26 |
| w/o Table Gene. | 19K | 50.20 | 54.29 | 76.19 | 42.35 | 55.43 |
| △ | | -5.73 | -6.09 | -3.98 | -3.85 | -5.26 |
| w/o Weakness Iden. | 34K | 53.12 | 51.72 | 75.82 | 42.12 | 56.28 |
| △ | | -2.81 | -8.65 | -4.35 | -4.08 | -4.41 |

Table 3: Ablation experiment results. We report average accuracy on four task types. △ stands for the performance gap between the Llama3.1-8B-Instruct finetuned with TableDreamer data and its variants. 'Hier.' and 'Hori.' stands for hierarchical and horizontal tables. 'Inst. Gene.', 'Inst. Comp.', 'Table. Gene.' and 'Weakness Iden.' represents three data evoluation directions and weakness data identification respectively.

the main experiments where the fine-tuning with TableGPT-syn-data (only including flat tables) results in poor performance on tables of different types (e.g., hierarchical tables from HiTab). Compared with others, removing horizontal tables leads to a lower performance decrease which may be because most benchmarks only contain flat or hierarchical tables. (2) *Ablation of data evolution.* We remove the data generated from different data evolution directions. We can find that all three data evolution directions make substantial contributions to the final model performance, and 'w/o Instruction Complication' causes a more significant performance decline than others, which highlights the importance of complex instructions in enhancing the model's table understanding ability. Unsurprisingly, 'w/o Data Evolution' causes the worst performance as we only fine-tuned the model with 3K seed data. This shows that, simply using LLMs to synthesize instructions of pre-defined types, which is the common practice of recent tabular LLMs, is insufficient to improve the model performance, and we need to thoroughly explore the vase input space for better data diversity. (3) *Ablation of weakness data identification.* We use all generated data from data evolution for fine-tuning rather than the selected weakness data. Despite using more synthetic data (34K), the model actually suffers a performance drop of 4.41, which suggests that choosing the weakness-exposing data is more conducive to model performance than blindly increasing the data volume. **More results and analysis are given in App. C due to space limitation**.

## 5 Conclusion

This papers introduces a novel data synthesis framework for table instruction tuning, which can generate diverse tables together with instructions spanning a wide range of tabular tasks, without relying on any public datasets. At the core of the proposed TableDreamer framework lies the iterative collaboration between input space exploration and weakness data identification. On the basis of TableDreamer, we construct and release 27K synthetic data, which can effectively enhance LLMs' table understanding ability and outperforms strong baselines. In conclusion, this paper promotes the research of data synthesis for the important table instruction tuning with the new method, dataset and thorough empirical study.

## 6 Limitations

Though this paper presents an effective framework as well as a systematic investigation within the scope of table instruction tuning data synthesis, there are certain limitations and promising directions that deserve future investigations. First, the proposed framework generates tables and instructions in the textual format. With the devolvement of multimodal large language models (MLLMs), considerable efforts have been dedicated to the multimodal or visual table understanding problem (Zheng et al., 2024; Deng et al., 2024; Zhao et al., 2024), which takes as input the table images rather than textual table sequence for visual understanding and also lacks the large amount of diverse instruction tuning data (i.e., triples of table image, instruction and response). One potential solution is to transform the TableDreamer synthetic textual tables into table images with automatic scripts, e.g., rendering the HTML tables into images with the python html2image package.

Second, there are three common paradigms for LLM-based data synthesis: Strong2Weak distillation (Huang et al., 2022), Weak2Strong Generalization (Burns et al., 2023), and self-improving or self-evolving (Tao et al., 2024). The proposed framework belongs to the Strong2Weak distillation where we use a strong LLM (Llama3.1-70B-instruct or GPT-4o) to synthesize input tables, instructions and responses to enhance the performance of a weaker LLM (e.g., Llama3.1-8B-instruct). The latter two paradigms worth more-depth future explorations, e.g., for the self-evolving paradigm, how can we continuously the table understanding ability of the most powerful LLMs like GPT-4o with its own synthetic data.

Third, current data synthesis methods for table understanding including this paper are restricted to synthesizing data for the supervised fine-tuning stage. It is worthwhile exploring the generation of table-related preference data to further improve the model performance with reinforce learning (Gallego, 2024; Wijaya et al., 2024). Particularly, we believe it is a very promising direction to explore incentivizing the table-based Deepseek-R1-style in-depth reasoning (DeepSeek-AI et al., 2025) of tabular LLM the with synthetic tabular task data that can provide correctness feedback using rule-based reward model.

## 7 Ethical Considerations

The main objective of the proposed TableDreamer framework is to develop a scalable data synthesis method for table instruction tuning to enhance the table understanding capabilities of LLMs. However, the data generated from the LLMs (Llama3.1-70B-instruct and GPT-4o) may contain harmful content in the synthesized tables, instruction and responses. To this end, we use the LLM-as-a-judge based on Llama3.1-70B-instruct to check for harmful content within the generated samples, and we also randomly sample 5K samples for manually checking. In our empirical evaluations, we do not observe such unsafe data but we still suggest adding relevant safety filtering strategies when using the proposed framework. The used benchmarks in the experiments are free and open datasets for research use, thus the authors foresee no ethical concerns.

## References


Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, Ilya Sutskever, and Jeff Wu. 2023. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. *Preprint*, arXiv:2312.09390.

Jiuhai Chen, Rifaa Qadri, Yuxin Wen, Neel Jain, John Kirchenbauer, Tianyi Zhou, and Tom Goldstein. 2024. Genqa: Generating millions of instructions from a handful of prompts. *Preprint*, arXiv:2406.10323.

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

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020. Tabfact: A large-scale dataset for table-based fact verification. *Preprint*, arXiv:1909.02164.

Zhoujun Cheng, Haoyu Dong, Zhiruo Wang, Ran Jia, Jiaqi Guo, Yan Gao, Shi Han, Jian-Guang Lou, and Dongmei Zhang. 2022. HiTab: A hierarchical table dataset for question answering and natural language generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1094–1110, Dublin, Ireland. Association for Computational Linguistics.

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, Ziyi Gao, Aixin Liu, Bing Xue,

Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, and et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Naihao Deng and Rada Mihalcea. 2025. Rethinking table instruction tuning. *Preprint*, arXiv:2501.14693.

Naihao Deng, Zhenjie Sun, Ruiqi He, Aman Sikka, Yulong Chen, Lin Ma, Yue Zhang, and Rada Mihalcea. 2024. Tables as texts or images: Evaluating the table reasoning ability of llms and mllms. *Preprint*, arXiv:2402.12424.

Víctor Gallego. 2024. Refined direct preference optimization with synthetic data for behavioral alignment of llms. *Preprint*, arXiv:2402.08005.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, and et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Vivek Gupta, Maitrey Mehta, Pegah Nokhiz, and Vivek Srikumar. 2020. INFOTABS: Inference on tables as semi-structured data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2309–2324, Online. Association for Computational Linguistics.

Tao Huang, Shan You, Fei Wang, Chen Qian, and Chang Xu. 2022. Knowledge distillation from a stronger teacher. *Advances in Neural Information Processing Systems*, 35:33716–33727.

Sujay Kumar Jauhar, Peter Turney, and Eduard Hovy. 2016. Tabmcq: A dataset of general knowledge tables and multiple-choice questions. *Preprint*, arXiv:1602.03960.

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. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Zhengbao Jiang, Yi Mao, Pengcheng He, Graham Neubig, and Weizhu Chen. 2022. OmniTab: Pretraining with natural and synthetic data for few-shot table-based question answering. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 932–942, Seattle, United States. Association for Computational Linguistics.

Yannis Katsis, Saneem Chemmengath, Vishwajeet Kumar, Samarth Bharadwaj, Mustafa Canim, Michael Glass, Alfio Gliozzo, Feifei Pan, Jaydeep Sen, Karthik Sankaranarayanan, and Soumen Chakrabarti. 2021. Ait-qa: Question answering dataset over complex tables in the airline industry. *Preprint*, arXiv:2106.12944.

Haoran Li, Qingxiu Dong, Zhengyang Tang, Chaojun Wang, Xingxing Zhang, Haoyang Huang, Shaohan Huang, Xiaolong Huang, Zeqiang Huang, Dongdong Zhang, Yuxian Gu, Xin Cheng, Xun Wang, Si-Qing Chen, Li Dong, Wei Lu, Zhifang Sui, Benyou Wang, Wai Lam, and Furu Wei. 2024a. Synthetic data (almost) from scratch: Generalized instruction tuning for language models. *Preprint*, arXiv:2402.13064.

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

Zhenyu Li, Xiuxing Li, Sunqi Fan, and Jianyong Wang. 2024b. Optimization techniques for unsupervised complex table reasoning via self-training framework. *Preprint*, arXiv:2212.10097.

Tianyang Liu, Fei Wang, and Muhao Chen. 2024. Rethinking tabular data understanding with large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 450–482.

Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. 2024. On LLMs-driven synthetic data generation, curation, and evaluation: A survey. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11065–11082, Bangkok, Thailand. Association for Computational Linguistics.

Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. 2023a. Chameleon: Plug-and-play compositional reasoning with large language models. In *The 37th Conference on Neural Information Processing Systems (NeurIPS)*.

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

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

Run Luo, Haonan Zhang, Longze Chen, Ting-En Lin, Xiong Liu, Yuchuan Wu, Min Yang, Minzheng Wang, Pengpeng Zeng, Lianli Gao, Heng Tao Shen, Yunshui

Li, Xiaobo Xia, Fei Huang, Jingkuan Song, and Yongbin Li. 2024. Mmevol: Empowering multimodal large language models with evol-instruct. *Preprint*, arXiv:2409.05840.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA. Association for Computational Linguistics.

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

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. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2024. Code llama: Open foundation models for code. *Preprint*, arXiv:2308.12950.

Yucheng Ruan, Xiang Lan, Jingying Ma, Yizhi Dong, Kai He, and Mengling Feng. 2024. Language modeling on tabular data: A survey of foundations, techniques and evolution. *Preprint*, arXiv:2408.10548.

Alexey Shigarov. 2023. Table understanding: Problem overview. *WIREs Data Mining and Knowledge Discovery*, 13(1):e1482.

Qingyi Si, Tong Wang, Zheng Lin, Xu Zhang, Yanan Cao, and Weiping Wang. 2023. An empirical study of instruction-tuning large language models in chinese. *Preprint*, arXiv:2310.07328.

Ananya Singha, José Cambronero, Sumit Gulwani, Vu Le, and Chris Parnin. 2023. Tabular representation, noisy operators, and impacts on table structure understanding tasks in llms. *Preprint*, arXiv:2310.10358.

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

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

Zhengwei Tao, Ting-En Lin, Xiancai Chen, Hangyu Li, Yuchuan Wu, Yongbin Li, Zhi Jin, Fei Huang, Dacheng Tao, and Jingren Zhou. 2024. A survey on self-evolution of large language models. *Preprint*, arXiv:2404.14387.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J. Zhang, Makesh Narsimhan Sreedhar, and Oleksii Kuchaiev. 2024a. Helpsteer2: Open-source dataset for training top-performing reward models. *Preprint*, arXiv:2406.08673.

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

Robert Wijaya, Ngoc-Bao Nguyen, and Ngai-Man Cheung. 2024. Multimodal preference data synthetic alignment with reward model. *Preprint*, arXiv:2412.17417.

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. 2024. Tablebench: A comprehensive and complex benchmark for table question answering. *Preprint*, arXiv:2408.09174.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. 2024. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing. *Preprint*, arXiv:2406.08464.

Da Yin, Xiao Liu, Fan Yin, Ming Zhong, Hritik Bansal, Jiawei Han, and Kai-Wei Chang. 2023. Dynosaur: A dynamic growth paradigm for instruction-tuning data curation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4031–4047, Singapore. Association for Computational Linguistics.

Tianshu Zhang, Xiang Yue, Yifei Li, and Huan Sun. 2024a. TableLlama: Towards open large generalist models for tables. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6024–6044, Mexico City, Mexico. Association for Computational Linguistics.

Xiaokang Zhang, Jing Zhang, Zeyao Ma, Yang Li, Bohan Zhang, Guanlin Li, Zijun Yao, Kangli Xu, Jinchang Zhou, Daniel Zhang-Li, Jifan Yu, Shu Zhao, Juanzi Li, and Jie Tang. 2024b. Tablellm: Enabling tabular data manipulation by llms in real office usage scenarios. *Preprint*, arXiv:2403.19318.

Weichao Zhao, Hao Feng, Qi Liu, Jingqun Tang, Shu Wei, Binghong Wu, Lei Liao, Yongjie Ye, Hao Liu, Wengang Zhou, Houqiang Li, and Can Huang. 2024. Tabpedia: Towards comprehensive visual table understanding with concept synergy. *Preprint*, arXiv:2406.01326.

Yilun Zhao, Linyong Nan, Zhenting Qi, Rui Zhang, and Dragomir Radev. 2022. ReasTAP: Injecting table reasoning skills during pre-training via synthetic reasoning examples. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9006–9018, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yilun Zhao, Zhenting Qi, Linyong Nan, Boyu Mi, Yixin Liu, Weijin Zou, Simeng Han, Ruizhe Chen, Xiangru Tang, Yumo Xu, Dragomir Radev, and Arman Cohan. 2023a. QTSumm: Query-focused summarization over tabular data. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1157–1172, Singapore. Association for Computational Linguistics.

Yilun Zhao, Haowei Zhang, Shengyun Si, Linyong Nan, Xiangru Tang, and Arman Cohan. 2023b. Investigating table-to-text generation capabilities of large language models in real-world information seeking scenarios. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 160–175, Singapore. Association for Computational Linguistics.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

Mingyu Zheng, Xinwei Feng, Qingyi Si, Qiaoqiao She, Zheng Lin, Wenbin Jiang, and Weiping Wang. 2024. Multimodal table understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9102–9124, Bangkok, Thailand. Association for Computational Linguistics.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. Lima: Less is more for alignment. *Preprint*, arXiv:2305.11206.

Wei Zhou, Mohsen Mesgar, Heike Adel, and Annemarie Friedrich. 2024. FREB-TQA: A fine-grained robustness evaluation benchmark for table question answering. In *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. Association for Computational Linguistics.

| Method | TQA | TFV | T2T | TableGPT | Ave. Acc |
|---|---|---|---|---|---|
| Llama3.1-8B-Instruct | 44.04 | 51.41 | 72.98 | 21.68 | 49.07 |
| w/ TableDreamer-27K | 55.93 | 60.37 | 80.17 | 46.20 | 60.69 |
| △ | 11.89 | 8.96 | 7.19 | 24.52 | 11.63 |
| Mistral-7B-Instruct-v0.3 | 37.98 | 45.24 | 67.10 | 44.26 | 45.88 |
| w/ TableDreamer-27K | 51.06 | 49.34 | 76.19 | 43.33 | 54.97 |
| △ | 13.07 | 4.10 | 9.09 | -0.93 | 9.08 |
| MiniCPM3-4B | 46.77 | 35.21 | 55.47 | 40.79 | 45.60 |
| w/ TableDreamer-27K | 53.03 | 40.50 | 64.51 | 42.85 | 51.80 |
| △ | 6.27 | 5.29 | 9.04 | 2.06 | 6.20 |
| InternLM2.5-7B-Chat | 36.81 | 46.20 | 72.15 | 39.52 | 46.02 |
| w/ TableDreamer-27K | 54.99 | 51.99 | 73.08 | 40.14 | 56.52 |
| △ | 18.18 | 5.79 | 0.93 | 0.62 | 10.50 |

Table 4: Comparison of average performance of different LLMs fine-tuned with the TableDreamer data.

| Characteristic | Value |
|---|---|
| Avg. instruction number per table | 3.4 |
| Row number per table (median/mean/min/max) | 15/16.8/4/43 |
| Column number per table (median/mean/min/max) | 13/14.8/4/45 |
| Cell number per table (median/mean/min/max) | 200/237/28/1008 |
| Instruction length by word (median/mean/min/max) | 29/36.9/6/900 |
| Output length by word (median/mean/min/max) | 288/412.9/3/11513 |

Table 5: Basic statistics of the TableDreamer-27K synthetic data.

# A Dataset Statistics and Examples

Table 5 shows the basic statistics of 27K synthetic data from GPT-4o, such as the average instruction number per table, instruction length (we use whitespace to split instruction and compute word number for simplicity) and so on. Figure 4 illustrates the distribution of row number and column number in the 1,541 synthetic tables from GPT-4o. Figure 6 illustrates more dataset examples.

# B Implementation Details

## B.1 TableDreamer Implementations.

The prompt for table generation is shown in the Figure 7, and the prompts and strategies for data evolution in three directions are given in Fig. 8, Fig. 10 , Fig. 11 and Table 8. The LLM-as-a-judge prompt used for weakness data identification is shown in Figure 9, which is modified from the correctness judging standard from HelpSteer2 (Wang et al., 2024a). The 20 seed tasks and their descriptions are given in Table 7, which are used by an LLM (Llama3.1-70B-instruct or GPT-4o) to generate seed instructions based on synthetic tables. Multiple instruction templates are used to combine the input table, table title and instruction to form the final input prompt in the training data and we adopt the recommended hyper-parameters from Deng and Mihalcea (2025) and perform the standard supervised fine-tuning with a learning rate of 1e-6 and a batch size of 128 for 2 epochs. During inference, we set the temperature to 0.01 for reproducible evaluation results.

## B.2 Baseline Implementations.

For general data synthesis baselines, we reimplement them with Llama3.1-70B-instruct to generate table instruction tuning data. For table instruction tuning data generation baselines, we directly use the released synthetic data as the training data. To reimplement self-instruct (Wang et al., 2023), we construct 175 general tabular task request with the help of GPT-4o and use them as seed data to generate more tabular tasks with the original self-instruct procedure. Then, the filtered tabular tasks are used to synthesize task-inputs which include input table and instructions. For Magpie (Xu et al., 2024) reimplementation, we modify the system prompt to ask the LLM to act as a table understanding expert that fulfills table-related tasks. Then, a pre-query template with the modified system prompt is input to the LLM and it will autonomously generate the input table and related instructions autoregressively, which are further filtered with the methods from the original paper. For GenQA (Chen et al., 2024), we modify the Generator-Conditional data synthesis prompt to generate input table and instructions based on the diverse topics from the original paper. For evol-instruct (Xu et al., 2023), we select 40K samples generated from Magpie and Self-instruct synthetic data as seed data for synthesizing new samples with the evol-instruct prompts. For Dynosaur (Yin et al., 2023) which synthesize instruction-tuning data by converting existing datasets, we collect 5 table understanding datasets including FinQA, SQA, WikiSQL, TAT-QA and PubHealthTab as the basic data source and carefully construct dataset metadata, which are used by LLM to design tabular tasks and instructions. Please refer to the original papers of baseline methods for more details. All experiments were conducted on one machine with 8 80GB A100.

# C Additional Experimental Results and Analysis

**Effect of Increasing Data Size.** We investigate the performance improvement resulting from the accumulation of TableDreamer synthetic data. To this end, we fine-tuned the Llama3.1-8B-Instruct

| Task Category | Benchmark | # Test samples | Ave. Input Length | Task Description | Metric |
|---|---|---|---|---|---|
| Table Question Answering (TQA) | WTQ | 4344 | 496.3 | TQA based on tables which usually possesses a flat structure with the first row as the sole column header. | Accuracy |
| | HiTab | 1576 | 399.4 | TQA based on tables which usually possesses hierachical headers and merged cells. | Accuracy |
| | AIT-QA | 511 | 275.2 | TQA based on hierarchical tables from the airline industry. | Accuracy |
| | TabMCQ | 1029 | 311.8 | TQA with multi-choice questions. | Accuracy |
| | TABMWP | 7686 | 89.6 | TQA requiring mathematical reasoning operations such as finding the largest number or do math computations. | Accuracy |
| Table Fact Verification (TFV) | TabFact | 6845 | 303.7 | Given a table as evidence and a statement, the task is to distinguish whether the given statement is entailed or refuted by the table. | Accuracy |
| | InfoTabs | 5400 | 155.1 | Given a infobox table as evidence and a statement, the task is to distinguish whether the givenstatement is entailed or refuted by the table. | Accuracy |
| Table to Text Generation (T2T) | QTSumm | 1078 | 242.8 | Given a table and a query, models must perform human-like reasoning and analysis over the given table to generate a tailored summary. | LLM-as-a-judge Acc. |
| | FeTaQA | 2003 | 263 | TQA with a free-form text answer rather than a short text span copied from the table. | LLM-as-a-judge Acc. |
| TableGPT | Column Finding | 1682 | 106.3 | Identify the column-name of a specific value that appears only once in a given table | Accuracy |
| | Data Imputation | 2000 | 147.8 | Predict the missing values in a cell based on the table context | Accuracy |
| | Row2Row Transformation | 570 | 101.7 | Transform table data based on input/output examples | Accuracy |
| | Missing Value Identification | 8000 | 107.1 | Identify the row and column position of the only missing cell in a given table | Accuracy |
| | TQA (SQA,WTQ) | 9048 | 229.5 | Answer a natural-language question based on the content of a table | Accuracy |

Table 6: Detailed description and statistics of used benchmarks. The average input length is computed by whitespace-spited word number.

with the initial seed data (3K), the synthetic data after the first round (10K, including seed data) and the total data after the second round (27K), respectively. From the results in Figure 5, we can observe that the model performance continues to improve with the growth of synthetic data. This demonstrates the value of the proposed progressive framework, which continuously explores the vast input space to improve the data diversity.

**Improvement to Different LLMs.** In addition to the Llama3.1-8B-Instruct, we also validated the performance gains of TableDreamer data for other LLMs including Mistral-7B-Instruct-v0.3, InternLM2.5-7B-Chat and MiniCPM3-4B. As shown in Table 4, all three LLMs can benefit from fine-tuning with TableDreamer data, which indicates the transferability of TableDreamer framework towards other LLMs. Compared with the Llama3.1-8B-Instruct, the performance gains of three LLMs are relatively smaller, which may be because we used the Llama3.1-8B-Instruct as the target LLM to identify vulnerability data in order to achieve targeted performance enhancement.

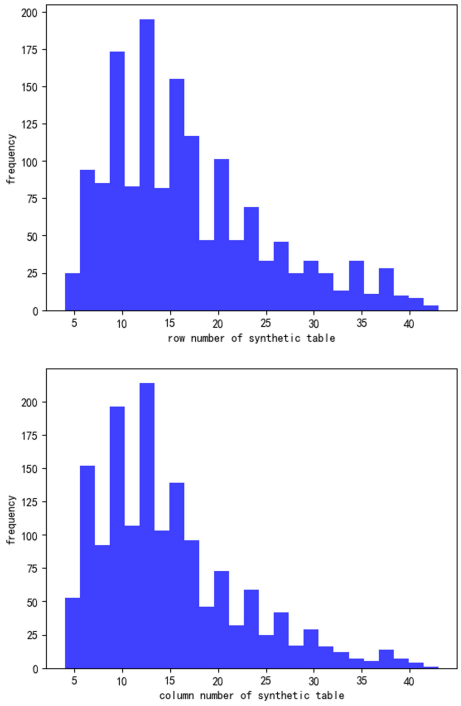


Figure 4: The distribution of row number and column number in the synthetic tables from GPT-4o.

| Task Category | Task Name | Task Description |
|---|---|---|
| Table Question Answering | Numerical reasoning problem | Given a table and a problem, the model needs to perform mathematical calculations based on numerical values in the table and the problem, such as addition, subtraction, averaging, calculation of growth rates, etc. |
| | Information seeking problem | Given a table and a related problem, the model needs to conduct information seeking from table cells based on the requirements of problem. |
| | Multihop reasoning problem | Given a table and a related problem, the model needs to conduct multi-hop reasoning according to the requirements of the problem to get the final answer. |
| | Time calculation problem | Given a table and a problem, the model needs to perform temporal calculations or comparison based on the time information, such as calculating the time difference between the release time of two movies. |
| | Table-based fact verification | Given a table and a statement, the model needs to determine whether the statement is true based on the table information. |
| Table-to-text generation | Table description | Given a table, the model needs to describe the table contents in detail. |
| | Table summarization | Given a table, the model is asked to summarize the key information in the table and generate a summary. |
| | Table analysis | Given a table, the model is asked to act as a professional data analyst, analyzing the key trends and phenomena in the table data, such as analyzing the sales of products in each quarter against the sales report. |
| Table Structure Understanding | Table size detection | Given a table, the model is asked to determine how many rows and columns the table has. |
| | Table cell extraction | Given a table and some cell locations (represented by row and column numbers), the model is asked to extract the cell text for the corresponding location. |
| | Table cell location | Given a table and some cell text, the model is asked to find the position of those cells in the table (represented by row and column numbers). |
| | Row&Column extraction | Given a table and some row or column numbers, the model is asked to extract all the text for the corresponding row or column. |
| | Merged cell detection | Given a table, the model is asked to determine whether the table contains merged cells and give the location of all the merged cells (represented by row and column numbers) if so. |
| Data Manipulation | Data formating | Given a table and user requirements, the model needs to modify the formats of some table data according to user requirements. |
| | Data cleaning | Given a table that may contain noise or errors, the model needs to identify and correct errors in the table based on the user requirements, such as typos, duplicate values, or illegal characters and so on. |
| | Data filtering | Given a table and some filter criteria, the model is asked to filter some rows and columns in the table based on the given criteria. For example, only reserving rows that meet certain criteria. |
| | Data classification | Given a table and user requests, the model needs to classify table data into pre-defined categories. For example, classifying movie reviews in the given table into positive reviews or negative reviews. |
| | Data sorting | The model needs to sort the data in the table according to the user's requirements and return the sorted data, which can be sorted in the ascending or descending order. |
| Table Processing | Table modification | Given the table and modification requirements, the model is asked to modify the overall table according to the user's requirements and returns the processed table. |
| | Format transformation | The model needs to convert the original table to the desired format based on user requirements, such as from Markdown format to Latex format. |

Table 7: Description of 20 seed tasks which are used to synthesize seed instructions based on synthetic tables.

| **Evolution Direction** | **Evolution Strategy** | **Description** |
|---|---|---|
| Instruction Complication | Add Constrains | adding one more constraints/requirements/conditions to the original instruction. |
| | Increase Depth | increasing the depth of the questions or requests in the original instruction. For instance, rewriting a simple question into a more profound question, or proposing a complex math problem instead of a simple calculation. |
| | Add Reasoning Steps | increasing the required reasoning steps of the original instruction. For instance, if the original task can be solved with a few simple steps, you should rewrite it into more complex problems that request multi-step reasoning. |
| | Add Task Number | adding more tasks/demands to the original instruction so that models need to perform multiple tasks. For instance, if the original instruction only contains one task, you can propose more tasks in the instruction and organize them in a Markdown list. |
| | Add Details | replacing general concepts in the original instruction with more specific concepts. |
| | Increase Length | writing long and multi-line instructions. Each instruction consists of multiple lines or paragraphs of text to create complex tasks. |
| | Add Context | designing more complex tasks which require not only the original input table but also additional input data (e.g., related contexts, code, background information or task examples, etc). |
| Instruction Generalization | New Instruction | draw inspiration from the example tabular instruction and come up brand new instructions about the provided table. New instructions require performing tasks that are different from example instructions. |
| | Similar Instruction | come up with task instructions about the given table, which are similar with the original instruction. The new instructions SHOULD belong to the same task type or the same demand as the example instruction. |
| Table Generalization | Change Format | convert the original table into a table in the new format |
| | Modify Header | paraphrasing some row headers or column headers into new headers with the same meaning. For instance, replacing original headers with synonyms. |
| | Modify Data | replacing the data in the original table with new data. Make new data as diverse as possible. You can also replace some data with null values. |
| | Order Permutation | randomly changing the order of rows and columns in the original table. |
| | Insert/Remove Data | randomly inserting or removing some new rows and columns. |

Table 8: Description of 14 detailed data evolution strategies. In the evolution of each direction, one strategy is randomly sampled to fill in the corresponding data evolution prompt.

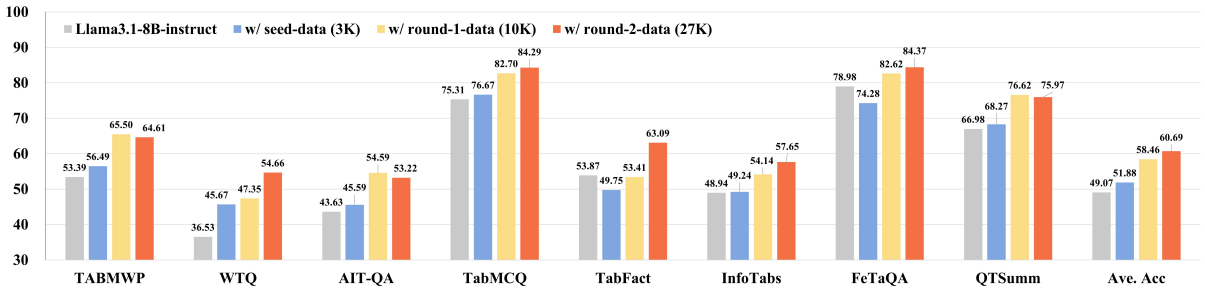


Figure 5: The performance improvement as the TableDreamer synthetic data continues to accumulate.

**Instruction**

First, identify the last row of the table, focusing on the details starting from the third column. Your goal is to extract information from the fields that represent age, aphasia severity, peer support group attendance, sessions attended, improvement……

**Table title**

Evaluating the Impact of Peer Support in Aphasia Rehabilitation Frameworks: Insights from Patient Surveys

**Flat Table**

| Survey ID | Participant Name | Age | Gender | Time Since Stroke (months) | Aphasia Severity |
|---|---|---|---|---|---|
| 101 | John Smith | 67 | Male | 18 | Moderate |
| 102 | Anna Brown | 58 | Female | 14 | Severe |
| 103 | Sarah Johnson | 72 | Female | 24 | Mild |
| 104 | David Lee | 63 | Male | 30 | Moderate |
| 105 | Emily White | 52 | Female | 16 | Severe |
| 106 | Michael Green | 70 | Male | 22 | Moderate |
| 107 | Linda Carter | 65 | Female | 20 | Mild |
| 108 | Chris Brown | 74 | Male | 19 | Severe |
| 109 | Megan Scott | 60 | Female | 12 | Moderate |
| 110 | Richard King | 68 | Male | 11 | Mild |

**Instruction**

Calculate the average subsidy amount for the 'Roads' category across all regions. What is the percentage difference between the highest and lowest values?

**Table title**

Rural Infrastructure Subsidies

**Hierachical Table**

| Region | Transport Infrastructure | | | |
|---|---|---|---|---|
| | Roads | | Railways | |
| | Subsidy Amount ($ million) | Projects | Subsidy Amount ($ million) | Projects |
| Region A | 45 | 12 | 30 | 8 |
| Region B | 50 | 14 | 28 | 7 |
| Region C | 48 | 13 | 35 | 9 |
| Region D | 55 | 16 | 32 | 10 |
| Region E | 52 | 15 | 20 | 6 |
| Region F | 60 | 18 | 31 | 9 |
| Region G | 58 | 17 | 29 | 8 |

**Instruction**

Based on the table, compare the revenue growth of enterprises that employ organic farming with those that implement renewable energy use, integrated pest management, and agroforestry ……

**Table title**

Sustainability Practices Adopted by Rural Enterprises and Their Correlation with Business Growth

**Horizontal Table**

| Enterprise Name | Green Fields Co. | Farm Fresh Ltd. | EcoFarmers SA | AgriLife Pvt. |
|---|---|---|---|---|
| Country | USA | UK | South Africa | India |
| Sustainability Practice | Organic Farming | Renewable Energy Use | Rainwater Harvesting | No-till Farming |
| Current Revenue (USD) | 123000 | 76000 | 145000 | 134000 |
| Revenue Growth (%) | 44.706 | 16.923 | 55.914 | 31.373 |

**Instruction**

Select the retailers from the dataset with a 'Brick-and-Mortar Sales Growth (%)' of under 5% and return their 'Total Sales' figures in descending order.

**Table title**

Comparison of Online vs Brick-and-Mortar Sales in the Fashion Industry, Q1 2023

**Flat Table**

| Retailer | Country | Online Sales (in $) | Brick-and-Mortar Sales (in $) | Total Sales (in $) | Online Sales Growth (%) | Brick-and-Mortar Sales Growth (%) |
|---|---|---|---|---|---|---|
| Zara | USA | 450000 | 650000 | 1100000 | 8.5 | 5 |
| H&M | UK | 300000 | 380000 | 680000 | 7.5 | 4.2 |
| Uniqlo | Japan | 520000 | 590000 | 1110000 | 10 | 6 |
| Primark | Ireland | 270000 | 750000 | 1020000 | 9.1 | 4.9 |
| ASOS | UK | 500000 | 0 | 500000 | 12 | 0 |
| Next | UK | 285000 | 415000 | 700000 | 7.9 | 3.7 |
| Nordstrom | USA | 610000 | 790000 | 1400000 | 9.5 | 6.5 |
| Urban Outfitters | USA | 315000 | 435000 | 750000 | 5.5 | 4 |

**Instruction**

Examine the relationship between personalization trends and convenience across all segments. Which segment shows the strongest impact from these customer preferences, and what operational strategies are influenced as a result?

**Table title**

Emerging Trends in E-commerce Supply Chain and Their Impact on Operational Strategies

**Hierachical Table**

| E-commerce Segments | Emerging Trends | | | | | |
|---|---|---|---|---|---|---|
| | Technology Integration | | Sustainability | | Customer Preferences | |
| | IoT | AI Driven | Green Packaging | Carbon Footprint | Personalization | Convenience |
| Online Retail | High | Medium | Low | Medium | High | Medium |
| | Medium | High | Medium | Low | High | High |
| | Low | Medium | High | Medium | Medium | Medium |
| | Medium | Low | Medium | High | Low | High |

**Instruction**

Determine the average emissions for each type of packaging, considering manufacturing, transportation, and recycling emissions. Provide the most and least emission-intensive packaging types.

**Table title**

Food Industry Packaging Lifecycle Emissions

**Horizontal Table**

| Plastic Bottles | Jute Bags | Bioplastic Containers | Foil Paper | Wax Paper |
|---|---|---|---|---|
| Material | Plastic | Glass | Composite | Aluminium |
| Manufacturing Emissions (kg CO2) | 8.2 | 5.5 | 4.3 | 7 |
| Transportation Emissions (kg CO2) | 1.9 | 2.1 | 1.2 | 1.4 |
| Recycling Emissions (kg CO2) | 0.7 | 0.9 | 0.5 | 0.8 |
| Total Emissions (kg CO2) | 10.8 | 8.5 | 6 | 9.2 |

**Instruction**

Examine the relationship between work stress level and sleep disruption events. Which subjects with a work stress level above 5 report the most sleep disruption events? Explain any patterns or anomalies you find.

**Table title**

Impact of Sleep Quality on Memory

**Flat Table**

| Subject ID | Age | Gender | Sleep Quality | Memory Test 1 | Memory Test 2 | Reaction Time (ms) | Sleep Duration (hrs) |
|---|---|---|---|---|---|---|---|
| 1001 | 25 | M | Good | 85 | 88 | 320 | 7.5 |
| 1002 | 30 | F | Fair | 72 | 75 | 340 | 6 |
| 1003 | 22 | F | Excellent | 92 | 95 | 280 | 8 |
| 1004 | 35 | M | Poor | 60 | 65 | 350 | 5 |
| 1005 | 28 | F | Good | 87 | 89 | 300 | 7 |
| 1006 | 45 | M | Fair | 74 | 77 | 330 | 6.5 |
| 1007 | 32 | F | Excellent | 93 | 94 | 275 | 8.5 |
| 1008 | 27 | M | Poor | 63 | 68 | 345 | 5.5 |
| 1009 | 40 | F | Good | 88 | 90 | 310 | 7.2 |

**Instruction**

Considering the data from 2015 to 2020, analyze the relative growth patterns in consumer spending across North America, Europe, Asia, and Other Regions. Which region's consumer spending shows the most consistent increase year over year? ……

**Table title**

Annual Increase in Consumer Spending on Sustainable Household Products from 2015 to 2025 by Region

**Hierachical Table**

| Year | Region | | | | | | Overall Growth Rate |
|---|---|---|---|---|---|---|---|
| | North America | | | Europe | | | |
| | USA | Canada | Mexico | UK | Germany | France | |
| 2015 | $200 | $50 | $30 | $40 | $60 | $55 | $700 |
| 2016 | $260 | $70 | $35 | $45 | $80 | $60 | $893 |
| 2017 | $310 | $85 | $40 | $65 | $100 | $75 | $1,078 |
| 2018 | $350 | $95 | $45 | $70 | $120 | $90 | $1,232 |
| 2019 | $390 | $110 | $50 | $75 | $140 | $95 | $1,425 |
| 2020 | $440 | $130 | $55 | $80 | $160 | $100 | $1,555 |
| 2021 | $480 | $145 | $60 | $90 | $180 | $110 | $1,725 |
| 2022 | $520 | $160 | $65 | $100 | $200 | $120 | $1,935 |
| 2023 | $570 | $180 | $70 | $110 | $220 | $130 | $2,135 |
| 2024 | $620 | $200 | $75 | $120 | $240 | $140 | $2,415 |
| 2025 | $680 | $220 | $80 | $130 | $260 | $150 | $2,685 |

**Instruction**

Based on the data, provide a general profile for users who prefer 'Tool A'. Include the average age, average satisfaction rating, and the most common use case for these users.

**Table title**

Blockchain Analytics User Preferences

**Horizontal Table**

| User ID | U001 | U002 | U003 | U004 | U005 | U006 | U007 |
|---|---|---|---|---|---|---|---|
| Age | 34 | 29 | 41 | 37 | 25 | 30 | 36 |
| Country | USA | UK | Germany | Canada | Brazil | Australia | India |
| Preferred Tool | Tool A | Tool B | Tool C | Tool D | Tool B | Tool A | Tool C |
| Use Case | Transaction Tracking | Market Analysis | Compliance | Asset Management | Transaction Tracking | Market Analysis | Compliance |
| Satisfaction (1-10) | 8 | 7 | 6 | 9 | 5 | 7 | 8 |
| Tool Customization | Yes | No | Yes | Yes | No | Yes | Yes |
| Subscription Type | Premium | Basic | Business | Enterprise | Basic | Premium | Enterprise |

Figure 6: More examples of TableDreamer synthetic data. Tables and instructions are clipped due to space limitation. We render tables into images for better visualization, and real tables could have various formats such as HTML, CSV, Markdown and et al.

**Table Synthesis Prompt**

Given a topic, a subtopic and a table title related to these topics, please design a <Table Type> table based on the table title and the following requirements.

## Table Design Requirements
1. **Table Content**: The table header and table data should match the given table title, i.e., the table title can describe the main content of the table. In addition, make the table content as realistic and diverse as possible.
2. **Table Header Structure**: <Header Structure Description>, e.g., the expected table has a 3-level hierarchical column header.
3. **Table Size**: <Table Size Description>, e.g., the expected table has 5 rows and 3 columns.
4. **Table Format**: <Table Format Description>, e.g., the expected table is represented in the HTML format.>
5. **Table Cell Dependencies**: When designing the table, there could be dependencies between different table cells. For instance, in a table titled 'Details of Company Net Profit', the cell values in the 'Profit' column should be equal to the difference between 'Revenue' cell values and 'Cost' cell values. In such cases, please represent the target cell's value using the corresponding calculation formula, formatted as a Markdown inline formula, e.g., '$(30-20)\\times 2.1$' and '$240+200+150$'. After generating the table, I will extract and compute these formulas, and fill in original cells with the computed results. Note that table cell dependencies are optional and are not necessary for every table.

6. **Output Format**: Output the designed table in the following JSON format.
```json
{
    "table_string": "<The string representation of the designed table>"
}
```

## Input
**Topic**: <Topic>
**Subtopic**: <Subtopic>
**Table Title**: <Table title>

## Output

Figure 7: The prompt used for synthesize diverse tables. The string in red color will be replaced with correlative content in implementation.

**Instruction Complication Prompt**

I want you act as an Instruction Creator.
Given a table, its title and a tabular task instruction, your goal is to generate {New Instruction Number} more complex instructions based on the original instruction, which makes it harder for language models like GPT-4 to handle.

Requirements:
1. You SHOULD generate new instructions with the following strategy: <Evolution Strategy Description>
2. New instructions are more difficult than the original instruction but SHOULD still be reasonable instructions about the given table.
3. The language for new instructions SHOULD be diverse and fluent.
4. The new instructions SHOULD belong to text-only tasks. Do not ask the model to create any visual or audio output.
5. Output new instructions in the following JSON format:
```json
{
  "new_instruction_list" : [ <instruction_1>, ..., <instruction_N> ]
}
```

## Table Title:
<Table Title>

## Table:
<String Representation of Input Table>

## The Original Instruction:
<The Original Tabular Task Instruction>

## Generated New Instructions:

Figure 8: The prompt used for data evolution in the instruction complication direction.

**LLM-as-a-Judge Prompt**

You will be given a prompt of a table-related task, a reference response and a response from a language model (LM).
Your task is to rate the correctness of the LM's response on a 5 point likert scale.
The Detailed Rating Breakdown is as follows.
- 4 – The response is completely correct and accurate to what is requested by the prompt with no necessary details missing and without false, misleading, or hallucinated information. If the prompt asks the LM to do a task, the task is completely done and addressed in the response.
- 3 – The response is mostly correct and accurate with a small amount of missing information. It contains no misleading information or hallucinations. If the prompt asks the LM to perform a task, the task is mostly successfully attempted.
- 2 – The response contains a mix of correct and incorrect information. The response may miss some information, contain misleading information, or minor hallucinations, but is more or less aligned with what the prompt asks for. If the prompt asks the LM to perform a task, the task is attempted with moderate success but still has clear room for improvement.
- 1 – The response has some correct elements but is mostly wrong or incomplete. The response may contain multiple instances of hallucinations, false information, misleading information, or irrelevant information. If the prompt asks the LM to do a task, the task was attempted with a small amount of success.
- 0 – The response is completely incorrect. All information provided is wrong, false or hallucinated. If the prompt asks the LM to do a task, the task is not at all attempted, or the wrong task was attempted in the response. The response is completely irrelevant to the prompt.

Additionally, flag the rating result as 'Invalid Prompt' in the following cases:
1. The input prompt requires the LM to perform a multimodal task (e.g., draw a picture or book a restaurant).
2. The input prompt is not about table-related tasks.
3. The input prompt is incomplete and illogical and contains lots of grammar mistakes.

Format your output in the JSON format:
```json
{
   'rating': <final rating result, string format>
   'explanation': <rating explanation, string format>
}
```

Figure 9: The LLM-as-a-judge prompt used for weakness data identification, which is modified from the correctness judging standard from HelpSteer2 (Wang et al., 2024a).

**New Instruction Generation Prompt**

I want you act as an Instruction Creator.
Given a table and its title, your goal is to draw inspiration from the example tabular instruction and to come up with a set of {New Instruction Number} brand new instructions about the provided table.

Requirements:
1. New instructions require performing tasks that are different from example instructions. You could include various types of tabular tasks like open-ended text generation, question answering, table editing, etc. You can also design any creative table-related tasks or demands that can be completed based on the given table.
2. Make new instructions as diverse as possible. For example, you could use diverse language style, combine questions with imperative instructions or necessary background contexts and so on.
3. New instructions should belong to text-only tasks. Do not ask the model to create any visual or audio output.
4. Output new instructions in the following JSON format:
```json
{
  'new_instruction_list': [ <instruction_1>, ..., <instruction_N> ]
}
```

## Table Title:
<Table Title>

## Table:
<String Representation of Input Table>

## The Original Instruction:
<The Original Tabular Task Instruction>

## Generated New Instructions:

**Similar Instruction Generation Prompt**

I want you act as an Instruction Creator.
Given a table, its title and an example instruction, your goal is to come up with a set of {New Instruction Number} similar task instructions about the given table.

Here are the requirements:
1. The new instructions SHOULD belong to the same task type or the same demand as the example instruction.
2. The difficulty of new instructions SHOULD be similar with the example instruction.
3. The language expression of new instructions SHOULD be diverse. For instance, you can paraphrase the original instruction, add colloquial expressions, change instruction format (e.g., convert open-ended questions to multi-choice questions), change word order and verb patterns, or directly write new instructions.
4. Output the new instructions in the following JSON format:
```json
{
  'new_instruction_list': [ <instruction_1>, ..., <instruction_N> ]
}
```

## Table Title:
<Table Title>

## Table:
<String Representation of Input Table>

## The Original Instruction:
<The Original Tabular Task Instruction>

## Generated New Instructions:

Figure 10: The prompt used for data evolution in the instruction generalization direction.

**Table Generalization Prompt**

I want you act as a Table Creator.
Given a table and its title and a tabular task instruction , your goal is to modify the original table to create a new table based on the following requirements.

Requirements:
1. You SHOULD create a new table with the following strategy: <Evolution Strategy Description>
2. The new table is still compatible with the given tabular task instruction.
3. Output the resulting table in the following JSON format:
```json
{
  'new_table': <The string representation of the new table>
}
```
## Table Title:
<Table Title>

## Table:
<String Representation of Input Table>

## The Tabular Task Instruction:
<The Tabular Task Instruction>

## The Output Result:

Figure 11: The prompt used for data evolution in the table generalization direction.