Large Language Models are few(1)-shot Table Reasoners

Shahnoza Yadgarova, KAIST Guldana Abduali, KAIST Batyrkhan Mukhlis, KAIST

Botagoz Issayeva, KAIST Yerbolat Uzakbay, City University of Hong Kong

Abstract

The advancement in large language models (LLMs) has previously shown promising results in reasoning tasks. The baseline paper explores the capabilities of LLMs in table-related tasks using few-shot in-context learning. Specifically, it examines the performance of LLMs on popular table QA datasets such as WikiTableQuestion, FetaQA, and TabFact, demonstrating promising results compared to models trained on table-specific corpora. Despite these promising results, the baseline paper highlights certain limitations, particularly the struggle of LLMs with interpreting large tables due to token limitations and the approach not being state-of-the-art. In this paper, we replicated the baseline results by testing direct and chain-of-thought prompting on table-related tasks. Additionally, we investigated how table decomposition approaches could enhance the results reported in the baseline paper. Our findings indicate promising implications for this approach. Furthermore, we explored how different existing LLMs handle various table reasoning tasks. The code and data are available at https://github.com/bachaquer/CS470_2024_Team9

1. Introduction

Tabular data is a crucial complement to textual data, which is pervasive and informative in our daily lives. Reasoning about combined tabular and textual information is a fundamental challenge in natural language understanding (NLU) and information retrieval (IR) (Wang et al., 2021)[1]. This capability is essential for a variety of downstream applications, such as table-based fact verification (FV) (Chen et al., 2020; Aly et al., 2021; Gupta et al., 2020)[2,3,4] and table-based question answering (QA) (Pasupat and Liang, 2015; Zhong et al., 2017; Nan et al., 2022; Cho et al., 2019)[5,6,7,8]. Table-based reasoning is inherently complex, requiring sophisticated textual, numerical, and logical reasoning across both unstructured text and (semi-)structured tables.

Traditional approaches to table-based reasoning have relied on synthesizing executable languages (e.g., SQL and SPARQL) to interact with tables (Date, 1989; Abdelaziz et al., 2017; Hui et al., 2021, 2022)[9,10,11]. However, these methods often fail to capture the semantics of text within the tables, which can limit their effectiveness in handling web tables with unnormalized, free-form text.

Recently, pre-trained models designed for table-based tasks (Herzig et al., 2020; Liu et al., 2021; Jiang et al., 2022; Gu et al., 2022; Cai et al., 2022)[12,13,14,15,16] have demonstrated significant improvements in reasoning over textual and tabular data. For instance, TaPas (Herzig et al., 2020)[12] improved the understanding of tabular data by predicting masked cell content in tables. Despite their advancements, these models typically require extensive fine-tuning on task-specific datasets, which can hinder their performance on new datasets with unseen reasoning types. Furthermore, fine-tuning can degrade a model's in-context learning capabilities (Wang et al., 2022c)[17].

In-context learning with large language models (LLMs) (Brown et al., 2020; Wei et al., 2022; Kojima et al., 2022)[18,19,20] has gained traction as a method to leverage the reasoning abilities of LLMs without fine-tuning. This approach involves providing input-output examples as prompts, enabling models to perform complex reasoning tasks by generating intermediate reasoning steps (Wei et al., 2022)[19]. Prior research (Wei et al., 2022; Kojima et al., 2022)[19,20] has shown that LLMs can achieve impressive performance on text reasoning tasks without the need for task-specific model designs and training data. However, the application of LLMs to table-based reasoning tasks remains underexplored.

Several challenges arise when applying LLMs to table-based reasoning, especially with large tables and complex questions. Encoding all content from a large table can be computationally prohibitive and introduce irrelevant information. LLMs struggle with "huge" tables exceeding 30 rows due to token limitations (Chen, 2022)[21]. Existing methods that retrieve sub-evidence for reasoning (Yin et al., 2020; Chen et al., 2020)[22,2] often require large amounts of domain-specific training data. Moreover, decomposing complex questions into simpler sub-questions can improve multi-step reasoning (Dua et al., 2022; Chen et al., 2022)[23,21], but direct decomposition using chain-of-thought prompting (Wei et al., 2022)[19] can lead to hallucinations, generating misleading sub-questions (Ji et al., 2022)[24]. Additionally, LLMs can make simple mistakes in symbolic operations, affecting subsequent reasoning steps (Chen, 2022)[21].

To address these challenges, we propose methods that decompose tables and extract key information to facilitate reasoning:

Evidence Decomposition: We reduce large tables into smaller, more manageable sub-tables by predicting relevant row and column indexes using a powerful LLM and a few prompting examples. This method retains essential evidence while excluding irrelevant information, improving focus and interpretability.

Evaluation of Open-Source LLMs: We assess the performance of different, up-to-date open-source LLMs on table reasoning tasks, identifying models that perform well without extensive fine-tuning.

The main contributions of this paper are:

- We develop a method to decompose large tables into smaller sub-tables by predicting the relevant row and column indexes, enhancing focus and interpretability.
- We evaluate the performance of various open-source LLMs on table reasoning tasks, identifying models that perform well without extensive fine-tuning.
- We improve accuracy scores across multiple datasets, particularly for tasks requiring factual extraction from tables.

2. Related Work

2.1. Table-based Reasoning

Historically, table-based reasoning has relied heavily on semantic parsing techniques to execute commands over tables. Early work, such as WikiTableQuestions (Pasupat and Liang, 2015)[5] and WikiSQL (Zhong et al., 2017)[6], involved generating SQL queries to extract and manipulate table data. These methods required strict adherence to table schemas and data types, often leading to limitations in handling unstructured or semi-structured data commonly found in web tables. Subsequent models, like Spider (Yu et al., 2018)[8], advanced this approach by tackling more complex SQL queries, yet still faced challenges in semantic understanding and flexibility.

To address these limitations, researchers proposed pre-training models on large table-related corpora. TAPAS (Herzig et al., 2020)[12], TaBERT (Yin et al., 2020)[22], and TAPEX (Liu et al., 2021)[13] exemplify this line of work. These models leverage joint representation learning of tables and text, enabling more nuanced reasoning without explicit query generation. Despite their success, these models typically require extensive fine-tuning on specific downstream tasks, highlighting a gap in generalizability and ease of adaptation.

2.2. In-context Learning with Large Language Models

Firstly, the advent of large language models (LLMs) like GPT-3 (Brown et al., 2020)[18] has revolutionized the approach to few-shot learning, including table-based reasoning. These models demonstrate remarkable capabilities in few-shot prediction by leveraging natural language prompts. Recent studies have shown that LLMs can perform various reasoning tasks, including mathematical and commonsense reasoning, with minimal examples (Rae et al., 2021; Smith et al., 2022)[25,26].

Specifically, "Large Language Models are few(1)-shot Table Reasoners" (Chen, 2022)[22] explored the potential of LLMs in table-related tasks using few-shot in-context learning. This work evaluated LLMs on datasets like WikiTableQuestions, FetaQA, and TabFact, demonstrating that these models, despite not being pre-trained on table-specific data, could achieve competitive results. This study underscored the versatility and robustness of LLMs, positioning them as a simple yet powerful baseline for future research in table reasoning.

2.3. Chain-of-thought Prompting

A significant enhancement in leveraging LLMs for reasoning tasks is the use of chain-of-thought (CoT) prompting. Introduced by Wei et al. (2022)[19], CoT prompting involves providing the model with examples that include intermediate reasoning steps, thereby guiding the model to perform complex reasoning tasks more effectively. This technique has shown substantial improvements in model performance across various reasoning tasks.

In the context of table-based reasoning, Chen (2022)[22] demonstrated that CoT prompting could significantly enhance the performance of LLMs. By incorporating reasoning chains, LLMs could achieve higher accuracy and produce more interpretable results. This approach was further refined by integrating self-consistency mechanisms (Wang et al., 2022)[17], which involve generating multiple reasoning paths and selecting the most consistent output, thereby improving reliability and robustness.

3. Recreational Results

3.1. Baseline

Firstly, we have recreated the baseline paper results to check if we are implementing the right code so we can add our improvements to make it better. This step was crucial to ensure that our setup and approach were aligned with established benchmarks before introducing our improvements. Despite some differences, we successfully achieved results close to the baseline, albeit with minor deviations. These variations are likely due to the use of a

different model and restrictions on the number of questions processed per time. For each dataset, we compared the recreated results with the baseline and analyzed the differences.

3.2. Datasets

We evaluated our recreational method on three table-based reasoning datasets, including TabFact (Chen et al., 2020)[2], WikiTableQuestion (Pasupat and Liang, 2015)[5], and FetaQA (Nan et al., 2022)[7]. The details of these three datasets are provided as follows:

- ***TabFact (Chen et al., 2020)[2]:*** This dataset focuses on table-based fact verification and comprises both simple and complex claims created by crowd workers using Wikipedia tables. Simple claims do not involve higher-order operations like max/min/count, while complex claims do. For example, given the statement, "The industrial and commercial panel has four more members than the cultural and educational panel," the task is to determine whether it is "True" or "False" based on the provided table. We evaluate 500 examples from the original test set containing 12,779 examples.
- ***WikiTableQuestion (Pasupat and Liang, 2015)[5]:*** This dataset contains complex questions also derived from Wikipedia tables. Crowd workers are tasked with formulating questions that require various operations such as comparisons, aggregations, and arithmetic calculations, necessitating compositional reasoning over multiple table entries. In our experiments, we use the unseen test set for evaluation. We evaluate 500 examples from the standard test set with roughly 4000 questions. In this dataset, we adopt the answer exact match as our evaluation metric.
- ***FetaQA (Nan et al., 2022)[7]:*** This dataset features free-form table questions demanding deep reasoning and understanding, often involving information from non-contiguous table chunks. Instead of short answers, FetaQA provides long, free-form answers, necessitating a high-level comprehension of the data.

3.3. Implementation Details

For our recreation, we utilized the GPT-3.5-turbo model due to the deprecation of the Text-Davinci-002 model. Our implementation was constrained by the high cost of OpenAI API calls, leading us to process only 500 questions at a time. We used 500 test cases from each of the following datasets for our evaluation: WikiTableQuestion, TabFact, and FetaQA. This selective approach allows us to efficiently show the performance of the baseline paper.

3.4. Evaluation Metrics

TabFact assesses the verification of table-based facts by determining if a statement is true according to table data. We use binary classification accuracy to evaluate the TabFact dataset. For WikiTableQuestion, denotation accuracy is our metric, which checks if the predicted answers match the correct ones. Unlike TabFact and WikiTableQuestion, which generate short phrase answers, FetaQA aims to produce comprehensive long-form answers. Thus, we evaluate FetaQA using the BLEU metric.

3.5. Results and Comparison

Here we show our recreated results for different datasets as follows.

WikiTableQuestions Based on baseline paper results, directly asking GPT-3 to generate answers leads to a 24% EM score. However, using CoT demonstrations, GPT-3's EM score improves to roughly 44.2%.

As can be seen from Table 1, GPT-3.5-turbo yielded different results in our recreation. The difference between the baseline results and our recreated results are:

- Direct Method: Difference of 19.8% (43.8% - 24%)
- CoT Method: Difference of 0.6% (43.6% - 44.2%)

FetaQA Based on baseline paper results, various fine-tuned models from Nan et al. (2022) are compared with GPT-3's performance. FetaQA aims to generate comprehensive long-form answers. The sacreBLEU score indicates GPT-3 trails behind T5-large, but human evaluation suggests it excels in correctness, adequacy, and faithfulness.

As can be seen from Table 2, GPT-3.5-turbo yielded different results in our recreation. The difference between the baseline results and our recreated results are:

- 1-shot GPT-3 Direct Method: Difference of 2.4% (24.48% - 26.88%).

TabFact As can be seen from Table 3, For the 1-shot (simple) Direct Method, we achieved an accuracy of 77.4%. The 1-shot (complex) Direct Method resulted in an accuracy of 64.6%. When using the 1-shot (simple) Chain of Thought (CoT) Method, the accuracy was 76.8%, and for the 1-shot (complex) CoT Method, it was 66.4%.

The differences between the baseline results and our recreated results are minimal, indicating that GPT-3.5-turbo performs similarly to the baseline GPT-3 in terms of

accuracy on TabFact. Our recreated results were almost close to the baseline paper, so we proceeded further with the improvements.

Type	Model	Test EM
1-shot	GPT-3.5-turbo Direct	43.8
1-shot	GPT-3.5-turbo Cot	43.6

Table 1: Recreational Results on WikiTableQuestions.

Type	Model	sacreBLEU
1-shot	GPT-3.5-turbo Direct	24.48

Table 2: Recreational Results on FetaQA.

Type	Model	Overall
1-shot (simple)	GPT-3.5-turbo Direct	77.4
1-shot (complex)	GPT-3.5-turbo Direct	64.6
1-shot (simple)	GPT-3.5-turbo Cot	76.8
1-shot (complex)	GPT-3.5-turbo Cot	66.4

Table 3: Recreational Results on TabFact.

Several factors influenced the discrepancies between our recreated results and those reported in the baseline paper. Firstly, the deprecation of the Text-Davinci-002 model required us to utilize the GPT-3.5-turbo model instead. Additionally, due to the high costs associated with OpenAI API usage, we limited our evaluations to 500 questions at a time, which may not fully capture the model's performance across the entire dataset. Moreover, the baseline paper's methods involving 2-shot prompting were only partially replicated because of missing details on the second shot, impacting our ability to fully recreate those specific results. Despite these challenges, our recreated results closely matched the baseline with some minor deviations.

Our findings offer several insights into the performance and limitations of GPT-3.5-turbo compared to the baseline results. Contrary to the baseline paper, Chain-of-Thought (CoT) prompting did not yield better accuracy for complex tasks like WikiTableQuestions compared to the Direct method. Furthermore, the model's performance significantly degrades when handling tables that exceed a certain size, leading to results that approximate random guesses. Large Language Model (LLM) prompting exhibits unpredictable randomness, indicating that current methods struggle to generalize well to large tables. There is a clear need for improved methods that can maintain consistent performance across various table sizes and complexity levels.

4. Methods

After replicating the baseline paper, we sought to implement new methods to identify potential improvements in LLM performance on table-reasoning tasks. As the baseline paper highlighted the main challenge of dealing with large tables, we addressed this issue by decomposing large tables and accessing key information first to generate responses. Additionally, our second method aimed to explore how different, up-to-date large language models perform on table reasoning tasks.

4.1. Table Decomposition

Our first approach to handling large tables involves decomposing the tables based on a given question before finding the correct answer. This intuitive method deals with large tables using an LLM-based pipeline. The Table Decomposition approach includes two different methods for decomposing tables. The first method is direct column-based table decomposition. Upon receiving a question, relevant columns from the large tables are extracted using the LLM. The large table is then decomposed into a subtable with pertinent columns and information to find the correct answer [Figure 1].

The second method takes a preliminary step to determine whether the table is organized by rows or columns before decomposition. Once this is established, the large table is decomposed accordingly. Additional instructions are provided for decomposing row-based tables. [Figure 2]

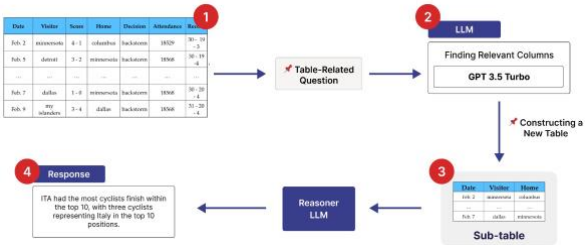

Figure 1. Table Decomposition by Columns

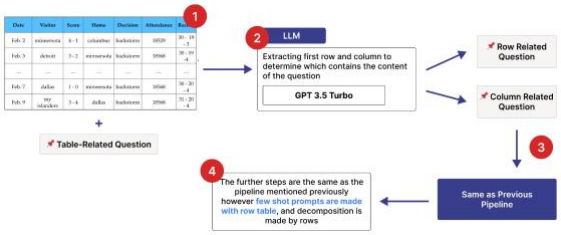

Figure 2. Table Decomposition by Rows and Columns

4.2. Various Large Language Models on Table Reasoning Tasks

We also experimented with different open-source large language models to evaluate their performance on table

reasoning tasks. For this study, we selected three LLMs: Llama3-70B, Gemma-7B, and Mixtral-8X7B Instruct. Llama3-70B, trained on 70 billion parameters, is a sophisticated model suitable for our approach. Gemma-7B was chosen for its lightweight nature, high performance, and versatility in handling a wide range of text generation tasks. Lastly, Mixtral-8X7B outperforms models like GPT-3.5 in benchmarks, offering faster inference and strong performance.

5. Experimental Results

5.1. Table Decomposition

We conducted experiments on three datasets: WikitableQA, FetaQA, and TabFact, using the GPT-3.5 instruct model on 500 questions. To ensure fair comparisons, we maintained the original prompts. The experimental results for the table decomposition approach are shown in Figures 3,4,5.

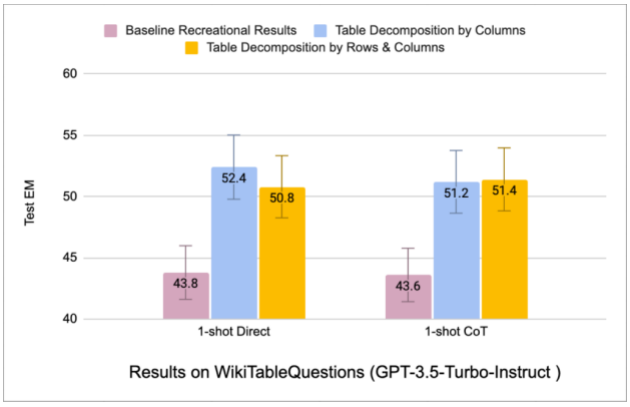

Figure 3. Results on Table decomposition for WikitableQA

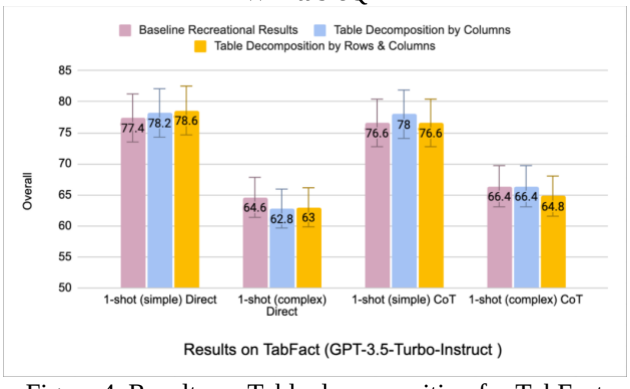

Figure 4. Results on Table decomposition for TabFact

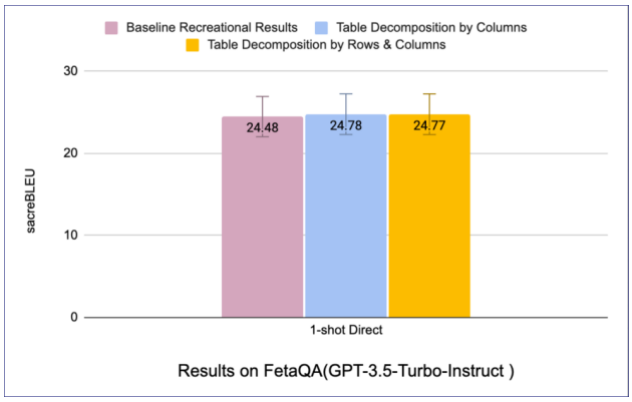

Figure 5. Results on Table decomposition for FetaQA

5.2. Various Large Language Models on Table Reasoning Tasks

We also obtained results on testing how different large language models perform on table reasoning tasks, especially with large tables. We used a platform called Replicate to make API calls and access various language models. Due to the high cost, time consumption, and the need for fair comparison, we decided to test on 500 questions. The figures below (Figures 6, 7, 8) present the results we obtained for this approach.

Although, GPR-4o is not an open source large language model we put it in this section to compare the performance.

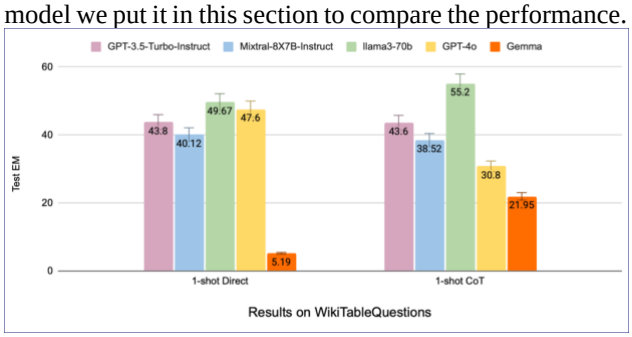

Figure 6. Results on testing different LLMs on WikiTableQA

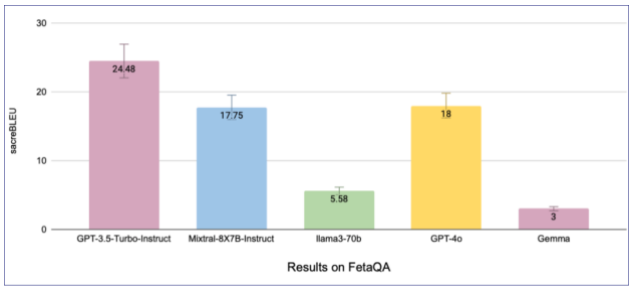

Figure 7. Results on testing different LLMs on FetaQA

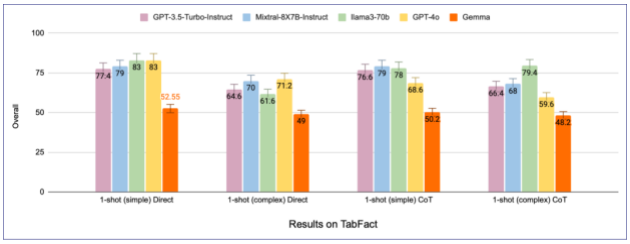

Figure 8. Results on testing different LLMs on TabFact

6. Discussion

6.1. On Table Decomposition approach

The implementation of table decomposition functions led to a noticeable increase in accuracy scores, indicating that this approach aligns well with our problem statement. For WikitableQA, accuracy improved by approximately **10%** for both direct and chain-of-thought prompting. In the case of TabFact, the approach generally enhanced accuracy by 1-2% or maintained it. However, for FetaQA, the accuracy improvement was insignificant. Additionally, there was minimal difference between decomposing tables by columns alone and by both rows and columns.

Overall, we consider table decomposition a promising approach that warrants further detailed investigation and development, especially to achieve better results with the TabFact and FetaQA datasets. Given that WikitableQA and TabFact are heavily based on extracting facts from tables, it is reasonable to expect that table decomposition facilitates the faster and more accurate retrieval of relevant information based on the questions asked. On the other hand, FetaQA involves questions that require free-form reasoning, which does not benefit as much from the table decomposition approach. The small difference observed between the two decomposition methods could be attributed to the relatively small number of horizontal tables categorized by rows, which we found to be less than 5% in WikitableQA through manual inspection.

This method involves accessing the language model two or three times, where mistakes made in intermediate steps can affect subsequent ones (Figure 9). Misunderstanding questions by the language model remains an issue. (Figure 10) Therefore, while table decomposition shows promise, especially for datasets focused on factual extraction, its efficacy is limited for those requiring more nuanced reasoning, necessitating further refinement and development.

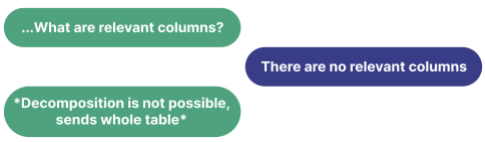

Figure 9. LLM dependency example

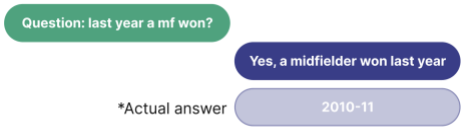

Figure 10. Actual example of LLM misunderstanding

6.2. On Open Source Large Language Models

For the WikitableQA dataset, Llama 3 outperformed the GPT-3.5 base model, while Mixtral performed nearly the same as the base model, though with a 3-5% lower accuracy. GPT-4o excelled in direct prompting but showed worse results with chain-of-thought prompting. Gemma had the lowest performance among all the models. In the FetaQA dataset, all models, particularly Llama and Gemma, performed relatively poorly. For TabFact, Llama and Mixtral generally outperformed the other models, while the remaining models exhibited lower performance.

Overall, Llama and Mixtral demonstrated better or equivalent results compared to the baseline recreation accuracy numbers. In contrast, Gemma and GPT-4o did not show significant improvement and even reduced accuracy results. This can likely be attributed to Gemma's smaller training data and GPT-4o's tendency to provide instructions rather than directly answering the questions, as observed in Figure 11. Models with higher accuracies were found to adhere more closely to the expected output format compared to others. This observation highlights the importance of output format consistency in achieving better performance.

Figure 11. GPT-4o example

6.3. On baseline recreation

While attempting to replicate the baseline results, we encountered difficulties with the chosen model. The initial model, text-davinci-002, seemed to be deprecated. Consequently, we adjusted our approach to use the GPT-3.5-turbo-instruct model. We believe that the deviations from the original baseline results are due to the intricate modifications in the newer model. Therefore, we concluded that implementing chain-of-thought reasoning may offer minimal or no improvement in processing.

6.4. On Datasets

Overall, each dataset showed different types of questions and tables. Below, we described our findings on each of them:

WikiTableQuestions: It includes a dataset characterized by large tables with varying structures, such as differences in column and row titles. The queries within this dataset demand logical reasoning and quantitative analysis. For instance, determining the cyclist with the highest rank among the top 10 requires an examination of the entire table.

FetaQA: Evaluating the FetaQA dataset is notably more complex compared to others. Understanding and implementing evaluation metrics such as sacreBleu and bleury presents a significant challenge, making it the most difficult dataset to improve accuracy on.

Tabfact: Tabfact contains both simple and complex tables, with accuracy variations of no more than 10%. It is the easiest dataset to verify responses.

Out of those, Tabfact showed highest results in accuracy and FetaQA showed the lowest. We believe this is due to the complexity of questions that appear in each dataset.

6.5. On Models

Here are some of our conclusions about each separate model:

GPT-3.5-Turbo-Instruct: This model proved to be the most stable and cost-effective model, which we utilized for the majority of our experiments.

Mixtral-8x7B-Instruct: Despite efforts to enforce consistency through direct prompts and settings, the model occasionally fails to adhere to specified formats. Additionally, it tends to produce overly lengthy outputs when given a larger token allocation, complicating information extraction. Compared to other models, it also exhibits longer processing times, potentially limiting its practical use in time-sensitive environments. Addressing these issues is crucial to improving the model's efficiency and applicability across different tasks.

Llama3-70B: Llama 3-70B showed the best performance overall, except for FetaQA. This discrepancy might be due to the complexity of the questions in FetaQA. We believe Llama 3-70B's performance is directly correlated with the extensive amount of data it has been trained on.

GPT-4o: Surprisingly, GPT-4o, the newest model, barely showed an increase in accuracy compared to the GPT-3.5-Turbo-Instruct model. Chain-of-thought reasoning actually decreased the accuracy for this model. Additionally, GPT-4o generally produced larger and more vague responses, often providing instructions instead of direct answers and not adhering to the expected output format.

Gemma: Gemma, having the lowest training data, showed the least performance among all the models. However, we observed almost a fourfold improvement in performance with the chain-of-thoughts method, indicating some potential for this approach despite its limited data.

6.6. Additional Findings

Throughout our investigation, we encountered tables with varying structures, where titles could appear in both the first row and the first column. Moreover, upon manually inspecting the outputs and their evaluation, we identified a flaw in the original code used for post-processing and scoring. The results were required to match almost exactly, which led to instances where answers that should have been correct **were not** counted as such (Figure 12).

Figure 12. False positive examples

Specifically for the WikitableQA dataset using Chain-of-Thought (CoT) prompting, after manually inspecting "Table Decomposition by Rows and Columns," we observed an actual increase in accuracy from 51.4% to 54.2%, representing an improvement of almost 3%. However, we remain uncertain whether this discrepancy was intentional or a mistake. Consequently, we opted to use their code to derive the results presented earlier in our findings.

7. Conclusion

Our study has reproduced the baseline results and extended the exploration of LLMs for table-based reasoning tasks, identifying limitations with the handling of complex, large datasets with current prompting techniques. These findings underline the need for further studies in developing fine-tuned approaches that can exploit the true powers of LLMs in table related tasks.

There is a greater need for improved table decomposition methods to operate on large tables. Experimental evaluation of the newer LLMs is under growing pressure as the models continue to evolve rapidly for better performance. Also, given the heavy costs levied on the computational side, particularly when financial pressure limits testing to dataset samples, the need for innovative training and deployment strategies that are cheap is obvious and urgent. Indeed, such strategies are essential for economical viability and scalability in the deployment of sophisticated LLMs across many domains due to the demands that increasingly complex data analysis tasks place on them. Furthermore, to enhance the accuracy of our findings, there is a pressing need to develop more precise post-processing and evaluation codes. These improvements will better account for instances that are actually correct but were previously marked as errors, ensuring a more accurate assessment of LLM performance.

References

[1] Fei Wang, Kexuan Sun, Muhao Chen, Jay Pujara, and Pedro Szekely. 2021. Retrieving complex tables with multi-granular graph representation learning. *ArXiv preprint*, abs/2105.01736.

[2] 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. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

[3] Rami Aly, Zhijiang Guo, Michael Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. Feverous: Fact extraction and verification over unstructured and structured information. *ArXiv preprint*, abs/2106.05707.

[4] 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.

[5] 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.

[6] Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103.

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

[8] Minseok Cho, Gyeongbok Lee, and Seung-won Hwang. 2019. Explanatory and actionable debugging for machine learning: A tableqa demonstration. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 1333–1336. ACM.

[9] Chris J Date. 1989. *A Guide to the SQL Standard*. Addison-Wesley Longman Publishing Co., Inc.

[10] Ibrahim Abdelaziz, Razen Harbi, Zuhair Khayyat, and Panos Kalnis. 2017. A survey and experimental comparison of distributed sparql engines for very large rdf data. *Proceedings of the VLDB Endowment*, 10(13):2049–2060.

[11] Binyuan Hui, Ruiying Geng, Qiyu Ren, Binhua Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, Pengfei Zhu, and Xiaodan Zhu. 2021. Dynamic hybrid relation exploration network for cross-domain context-dependent semantic parsing. In *AAAI Conference on Artificial Intelligence*.

[12] Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisen- schlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online. Association for Computational Linguistics.

[13] Qian Liu, Bei Chen, Jiaqi Guo, Morteza Ziyadi, Zeqi Lin, Weizhu Chen, and Jian-Guang Lou. 2021. Tapex: Table pre-training via learning a neural sql executor. *ArXiv preprint*, abs/2107.07653.

[14] 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 Com- putational Linguistics: Human Language Technologies*, pages 932–942, Seattle, United States. Association for Computational Linguistics.

[15] Zihui Gu, Ju Fan, Nan Tang, Preslav Nakov, Xiaoman Zhao, and Xiaoyong Du. 2022. Pasta: Table-operations aware fact verification via sentence-table cloze pre-training. *ArXiv preprint*, abs/2211.02816.

[16] Zefeng Cai, Xiangyu Li, Binyuan Hui, Min Yang, Bowen Li, Binhua Li, Zhen Cao, Weijie Li, Fei Huang, Luo Si, and Yongbin Li. 2022. Star: Sql guided pre-training for context-dependent text-to-sql parsing. In *EMNLP*.

[17] Yihan Wang, Si Si, Daliang Li, Michal Lukasik, Felix Yu, Cho-Jui Hsieh, Inderjit S Dhillon, and Sanjiv Kumar. 2022c. Preserving in-context learning ability in large language model fine-tuning. *ArXiv preprint*, abs/2211.00635.

[18] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

[19] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. *ArXiv preprint*, abs/2201.11903.

[20] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *ArXiv preprint*, abs/2205.11916.

[21] Wenhu Chen. 2022. Large language models are few (1)-shot table reasoners. *ArXiv preprint*, abs/2210.06710.

[22] Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426, Online. Association for Computational Linguistics.

[23] Dheeru Dua, Shivanshu Gupta, Sameer Singh, and Matt Gardner. 2022. Successive prompting for decomposing complex questions. *ArXiv preprint*, abs/2212.04092.

[24] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2022. Survey of hallucination in natural language generation. *ACM Computing Surveys*.

[25] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susan- nah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

[26] Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model.*arXiv preprint arXiv:2201.11990*.