TableCritic: Refine Table Reasoning via Self-Criticism and Tool Library

Ruochun Jin, Dong Wang, Xiyue Wang† and Haoqi Zheng†
{jinrc,wangdong19a,wangxiyue,zhenghaoqi}@nudt.edu.cn
College of Computer Science and Technology, National University of Defense Technology,
No. 137 Yanwachi Street, Changsha, Hunan, 410073, P.R.China

Abstract

The rapid development of large language models (LLMs) has spurred their applications to tabular data. Prior research has investigated prompt engineering and external tool integration (e.g., code interpreters) to enhance LLMs' table comprehension, yet existing approaches struggle to generalize across diverse table-based reasoning tasks due to task-specific fixed workflows. Additionally, LLMs' inherent limitations, including hallucination and unreliable reasoning, necessitate iterative refinement mechanisms for robust performance. Inspired by cognitive-inspired problem-solving strategies, where iterative reflection and tool augmentation improve decision-making, we propose TableCritic, a functionally grounded computational framework for adaptive table reasoning. Following the functional-structural model taxonomy, our framework operates at the algorithmic level, implementing a dynamic feedback loop: (1) LLM-driven self-critique evaluates intermediate outputs and (2) tool-guided error correction. This task-agnostic architecture achieves performance transparency through modular tool composition while avoiding neurobiological implementation constraints. Experiments show that TableCritic significantly improves accuracy and outperforms baselines across various table-based tasks.

Keywords: Cognitively-Inspired AI, Table-based Reasoning, Large Language Model

Introduction

Table-based reasoning, such as Table Question Answering (TQA) (W. Chen, Chang, Schlinger, Wang, & Cohen, 2020; Z. Chen et al., 2021; Pasupat & Liang, 2015; Zhu et al., 2021) and Table Fact Verification (TFV) (Aly et al., 2021; W. Chen et al., 2019), requires a model to answer queries based on information extracted from the tables, where current solutions typically rely on prompting large language models (LLMs) with input tables flattened into a textual sequence (W. Chen, 2022; Cheng et al., 2023; J. Jiang et al., 2023; Z. Wang et al., 2024; Ye et al., 2023). To address the challenges of complex reasoning (X. Lu, Pan, Ma, Nakov, & Kan, 2024), limited context window (Mashaabi, Al-Khalifa, & Al-Khalifa, 2024) and hallucinations (Ji et al., 2023) in LLMs during table-based reasoning, researchers have proposed effective prompt engineering methods such as question decomposition (Ye et al., 2023), chain-of-thoughts (CoT) (Wei et al., 2022), and program-of-thoughts (PoT) (W. Chen, Ma, Wang, & Cohen, 2022). Moreover, LLMs have been integrated with external tools to enhance their performance on table-based reasoning tasks (P. Lu et al., 2024).

However, there remain the following three drawbacks in previous reasoning solutions. First, most existing work supports only one external tool, which is typically a code interpreter (Cheng et al., 2023; J. Jiang et al., 2023; Y. Zhang et al., 2023). However, the varied distributions across a table-based task pose a challenge that cannot be easily addressed by a single tool. For instance, about 20% of WikiTableQuestions (Pasupat & Liang, 2015) are not answerable by pure SQL with a code interpreter, mostly due to the limited coverage of the SQL grammar (Shi, Zhao, Boyd-Graber, Daumé III, & Lee, 2020). Second, previous studies can hardly simultaneously handle different table-based reasoning tasks, which are often designed as fixed workflows for a specific task. For instance, research on TFV tasks (Cheng et al., 2023; Y. Zhang et al., 2023) primarily focuses on precise information extraction and numerical reasoning. In contrast, research on TQA tasks (Zhao, Chen, Cohan, & Zhao, 2024) generally emphasizes leveraging the reasoning capabilities of LLMs, particularly for interpreting large tables and answering complex questions. Third, current solutions generally rely on single-round outputs only, which occasionally suffer from the misbehavior of LLMs, such as hallucination, faulty code, and unfaithful reasoning (M. Chen et al., 2021; Gehman, Gururangan, Sap, Choi, & Smith, 2020; Pourreza & Rafiei, 2024).

To address these challenges, we propose TableCritic, a functionally-grounded framework that orchestrates LLMs' self-critical capabilities and a modular toolset to iteratively refine table-based reasoning. Drawing on the computational model taxonomy proposed by Lieto (Lieto, 2021), our design operates at the algorithmic level, prioritizing task-agnostic adaptability over biologically constrained architectures. Specifically, the toolset can be selected from a tool library based on the task's characteristics, which includes current mainstream reasoning methods suitable for various tasks and is extensible. As shown in Fig.1, the TableCritic framework mainly consists of the following four steps: (1) TableCritic obtains the model's initial answer. (2) It then prompts the model to evaluate whether the answer requires refinement and generates corresponding suggestions. (3) If refinement is deemed necessary, TableCritic adaptively selects a set of tools from the tool library based on the current table task. Following the multi-tool consistency mechanism, which draws on the principle of mix self-consistency (T. Liu, Wang, & Chen, 2023), the optimal final answer is determined through ma-

† Corresponding author

In D. Barner, N.R. Bramley, A. Ruggeri and C.M. Walker (Eds.), *Proceedings of the 47th Annual Conference of the Cognitive Science Society*

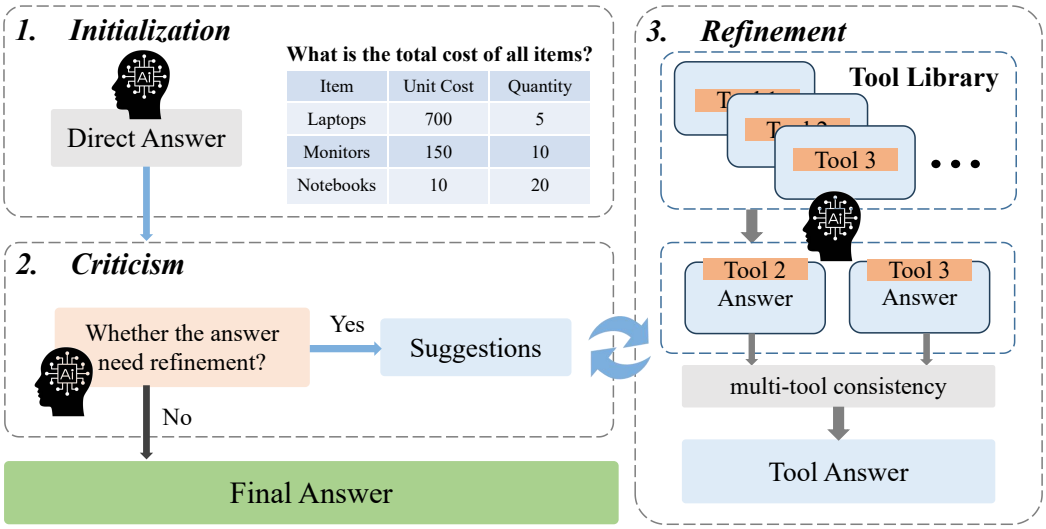


Figure 1: Overview of TableCritic. (1) Given a table with questions, TableCritic begins by generating an initial output. (2) Then, the output is passed to the criticism module to assess whether refinement is needed. (3) If refinement is necessary, TableCritic regenerates the answer using selected tools from the tool library by multi-tool consistency, and the refined answer is then sent back to the criticism module. Steps (2) and (3) iterate until (2) determines that no further refinement is needed.

jority voting among candidate answers generated by multiple tools. (4) Step (3) and (2) iterate until the self-criticism module in step (2) determines that no further refinement is needed, which effectively enhances model responses to table-based tasks.

We evaluated our approach in a variety of LLMs, including GPT-4o(Achiam et al., 2023), GPT-3.5 (Brown, 2020) and open source LLaMA-3 (Dubey et al., 2024) variants (7B and 70B), spanning two distinct tasks: Table Question Answering and Table Fact Verification. Our findings demonstrate the effectiveness of the TableCritic framework, with balanced performance across both tasks. When applied to GPT-3.5, TableCritic surpasses the initial outputs, achieving an average accuracy improvement of 6.0% across various tasks and reaching the best average performance compared to all baselines, with an accuracy of 82.5%. Specifically, the highest performance improvement can reach up to 25.4% on Llama3-70B. In summary, our primary contributions are as follows:

- We introduce the TableCritic framework that is, to the best of our knowledge, the first to enhance table processing capabilities through human-like self-criticism.
- We experimentally verify that by integrating multiple tools with multi-tool consistency, TableCritic demonstrates balanced performance and great scalability across various table-based reasoning tasks.
- We conducted comprehensive experiments across various models and baseline approaches, demonstrating that TableCritic brings significant improvements to all foundational models and outperforms all of the baseline methods employed.

Related Work

Recent research has primarily focused on designing prompts and pipelines that leverage the remarkable capabilities of LLMs to address specific types of datasets. Building on the concepts of chain of thought and question decomposition, Chain-of-Table (Z. Wang et al., 2024) and StructGPT (J. Jiang et al., 2023) leverage LLMs to gradually gather relevant evidence to infer answers. Dater (Ye et al., 2023) and TaPERA (Zhao et al., 2024) decompose the input question into sub-questions and generate answers based on each sub-question. Additionally, research demonstrates that when integrated with external tools, LLMs can significantly enhance their capabilities in tool-related areas (Schick et al., 2024; Zhuang, Yu, Wang, Sun, & Zhang, 2023). Binder (Cheng et al., 2023) and ReAcTable (Y. Zhang et al., 2023) leverage the symbolic code generation capabilities of LLM to parse text into SQL or Python code and execute it using a code interpreter. Besides, Mix-SC (T. Liu et al., 2023) directly combines the LLMs' direct reasoning and symbolic reasoning to produce the final answer, utilizing a mixed self-consistency mechanism that aggregates outputs from the two reasoning methods through majority voting. Unlike previous approaches, we propose a tool library that features multiple tools, which is capable of adaptively selecting a set of tools based on the characteristics of the current table-based task. Furthermore, to better utilize multiple tools in multi-tool scenarios, we propose the multi-tool consistency mechanism, which conducts a majority vote across all candidate answers from different tools to determine the final answer.

Like humans, large language models (LLMs) do not always generate the best output on their first try. Self-refine (Madaan et al., 2024) improved initial outputs from LLMs

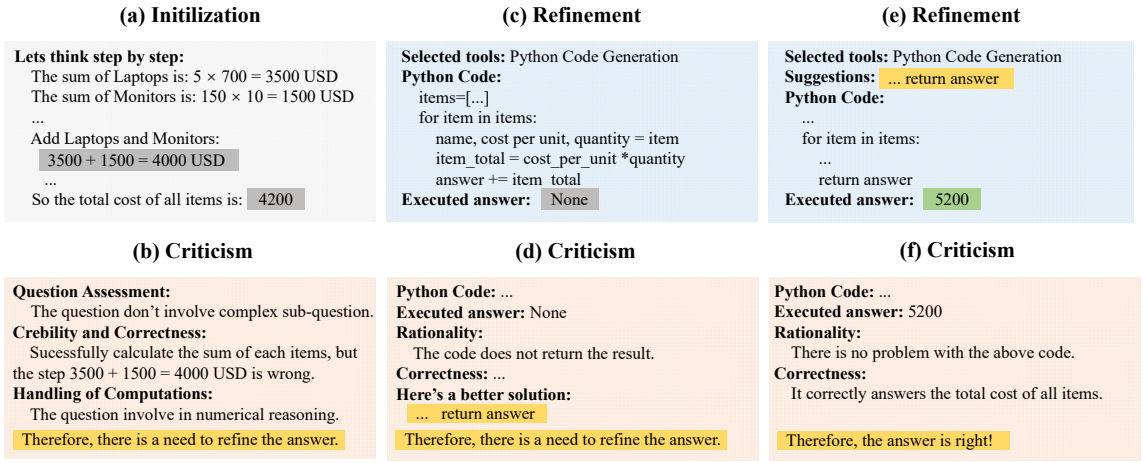


Figure 2: Example of a TQA task. The first row represents the table processing workflow, with (a) indicating direct answers generated by the initialization module and blue indicating answers generated by the refinement module using tools, which sends its output to the criticism module shown in the second row. The criticism module continues its evaluation until no further refinement is required. Here, gray indicates errors, yellow represents decisions and suggestions, and green denotes the correct answer.

through iterative feedback and refinement. Critic (Gou et al., 2023) amend the outputs like human interaction with tools. However, to the best of our knowledge, no human-like mechanism has been applied in the table-based tasks domain. Thus, inspired by human cognitive processes where iterative reflection and tool-augmented reasoning enhance decision-making, our proposed framework leverages feedback from the LLM's self-criticism and external tools to evaluate the quality of answers and suggest corrections.

Methods

In this section, we first define the tasks of TQA and TFV and then present main pipeline of TableCritic. As outlined in Fig.1, the TableCritic framework comprises three main components: the initialization module, the criticism module, and the refinement module.

Problem Definition.

Each instance in table-based reasoning can be represented as a triplet (T, Q, A), where T is the table, Q denotes a question or statement related to the table, and A is the ground-truth answer. Table-based reasoning aims to predict the answer A given the question Q and the table T as inputs. Specifically, for table-based fact verification (TFV), the answer $A \in \{0,1\}$ is a boolean value determining whether the input statement Q is true or false. As for table-based question answering (TQA), the answer is a natural language sequence $A =< a_1, \cdots, a_n >$ to the question Q. To conform with the model training paradigm, we follow (Ye et al., 2023; Y. Zhang et al., 2023) and convert the table into textual representations.

Initialization Module.

Given a table T and a question (or statement) Q as inputs, an LLM $\mathcal{M}$ is employed to generate the initial answer y_0 with a chain-of-thought (Wei et al., 2022) thinking step $\mathcal{C}_{y_0}$. Specifically, we use a step-by-step prompt P_{init} to instruct the model to answer the question.

Criticism Module.

We employ the same LLM $\mathcal{M}$ in the initialization stage to decide whether the initialization and refinement modules' outputs require further improvements and then provide refinement suggestions.

Given outputs y_0 and its thinking step $\mathcal{C}_{y_0}$ from the initialization module as inputs in this stage, we design a prompt $P^{y_0}_{critic}$ that consists of both rules and instructions to guide the model $\mathcal{M}$ in determining whether further refinement is needed. For example, based on the observed low accuracy of the LLM in numerical computation, we establish a rule that if the problem Q involves numerical reasoning, further refinement is required. As shown in Fig.2(b), TableCritic first identifies that the reasoning process behind $3500 + 1500 = 4000$ is incorrect, noting that the question involves numerical reasoning and therefore necessitates tool assistance for modification. Alternatively, if no further refinement is necessary, TableCritic will return y_0 as the final answer.

As for inputs from the refinement module, since the answers are obtained with the assistance of tools, we employ specialized prompts $P^{t_i}_{critic}$ specifically designed for tool t_i to guide the judgments of $\mathcal{M}$.Answers that require no further refinements are returned directly. In contrast, those in need of refinement are attached with tool suggestions $\mathcal{S}_{t_i}$ from $\mathcal{M}$ and returned to the refinement module for improvements. From Fig.2(d), the input is based on Python code, TableCritic verifies the code's accuracy—ensuring it is free from errors—and assesses its completeness to determine if it effectively addresses the table problem. Upon completion of the criticism,

it provides suggestions for refinement, targeting any errors or gaps in information.

Refinement Module

To effectively address table-based tasks, we propose a comprehensive and scalable tool library $\mathcal{T}_p$. Any tools or methods beneficial for table processing can be incorporated into the library, allowing for more flexible utilization. Specifically, this paper introduces two tools: a sub-table extraction tool for handling long and noisy contents and a Python code generation tool for numerical computations or key identification.

With the tool library at hand, we can utilize appropriate tools $\mathcal{T}_s \subset \mathcal{T}_p$ that are task-dependent, heuristically selected, or automatically chosen for effective table processing. In our implementation, we prespecify tools for different tasks to facilitate evaluation and experimentation. As shown in Fig.2(c), when the tool t_i is being used for the first time, TableCritic generates answers directly with its processing steps $\mathcal{C}_{t_i}$; otherwise, it integrates suggestions $\mathcal{S}_{t_i}$ from the criticism module to prevent the recurrence of similar errors as in Fig.2(e).

Specifically, leveraging the tabular data processing tools, each tool generates n candidate responses. Following the completion of candidate generation across all tools, TableCritic employs a multi-tool consistency mechanism to determine the final answer, which is subsequently fed into the criticism module along with its corresponding tool. Formally, let $a_{t_i}^j$ denote the j^{th} candidate output from tool t_i, where $i \in \{1,2,...,|\mathcal{T}_s|\}$ indexes the tool set $\mathcal{T}_s$ and $j \in \{1,2,...,n\}$ enumerates the candidate outputs per tool. The multi-tool consistency mechanism executes a majority voting protocol across the complete set of candidate outputs $\{a_{t_i}^j \mid \forall i,j\}$, considering all tools and their respective candidates. This procedure enables TableCritic to derive the optimized output $y_{\sqcup+1}$ through systematic aggregation of multi-tool evidence, ensuring robust decision-making through cross-verification of tool outputs.

Algorithm 1 encapsulates the essence of the TableCritic framework, highlighting its iterative approach.

Algorithm 1 TableCritic algorithm

Require: model $\mathcal{M}$, table T, question or statement Q, prompts, table tools pool $\mathcal{T}_p$

```
select T_s ⊂ T_p
C_{y0}, y0 = M(P_init, T, Q)                    ▷ Initialization Module
needRefinement = M(P^{y0}_critic, T, Q, y0, C_{y0})
                                                ▷ Criticism Module
if not needRefinement then
    return y0
end if
for iteration ⊔ ∈ 0, 1, ..., iter_max − 1 do
    for tool index i ∈ 0, 1, ..., |T_s| do
        C_{ti}, y_{ti} = M(P_{ti}, T, Q, S_{ti})    ▷ Refinement Module
    end for
    y_{⊔+1} = arg max_a Σ_{i=1}^{|T_s|} Σ_{j=1}^{n} 𝟙(a_{ti}^j = a)
                                                ▷ Multi-tool Consistency
    S_{ti}, needRefinement = M(P^{ti}_critic, T, Q, C_{ti})
                                                ▷ Criticism Module
    if not needRefinement then
        return y_{⊔+1}
    end if
end for
return y_⊔
```

Experiments

In this section, we will talk about the implementation of the experiment. All of the algorithms are achieved in Python 3.

Datasets and Metrics

To evaluate the efficacy of the TableCritic framework, we have curated six benchmark datasets from the realms of TQA and TFV: **WikiTableQuestions(WTQ)** (Pasupat & Liang, 2015) is a dataset of complex questions on semi-structured Wikipedia tables. **FinQA** (Z. Chen et al., 2021) is a large-scale dataset with 2.8k financial reports for 8k QA pairs annotated by finance expert. **TabMWP** (P. Lu et al., 2022) is a dataset containing open-domain grade-level problems that require mathematical reasoning on both textual and tabular data. **TabFact** (W. Chen et al., 2019) is a dataset of 118k statements and 16k Wikipedia tables for verifying whether a statement is entailed or refuted by the table. **Scitab** (X. Lu, Pan, Liu, Nakov, & Kan, 2023) is a dataset consisting of 1.2K expert-verified scientific claims originating from authentic scientific publications and requiring compositional reasoning for verification. **Pubhealth(PubHT)** (Kotonya & Toni, 2020) is a comprehensive dataset for explainable automated fact-checking of public health claims.

Given that the ground-truth for Scitab and Pubhealth includes a third option beyond true and false, specifically "not enough info", we have extracted subsets, denoted as **Scitab*** and **Pubhealth***, that focus exclusively on judging true or false for our experiments. This approach aims to align with the data format of TabFact and concentrate more on the model's reasoning capabilities when information is available. Due to budget constraints, we randomly sampled 500 examples from the validation set of each dataset.

Following prior studies (Cheng et al., 2023; Ye et al., 2023), we use denotation accuracy to evaluate the TQA tasks, which measures the proportion of predictions that exactly match the true answers. Specifically, for the WTQ dataset, we adopt and improve upon the Semantic Match Evaluator used in Binder (Cheng et al., 2023). For TFV datasets, given that the answers are binary (true or false), we employ binary classification accuracy.

Baselines

To verify the enhancement effect of TableCritic on the base model, we selected two widely recognized benchmark mod-

Table 1: Performance evaluation across baseline models using the TableCritic framework. We use the test set of each dataset to calculate the accuracy. For the WTQ dataset, we adopt the Semantic Match Evaluator. The last column shows the average accuracy. The red numbers indicate the average increase over self-consistency.

		TQA			TFV			AVG.
Models	**Settings**	**WTQ**	**FinQA**	**TabMWP**	**TabFact**	**Scitab***	**PubHT***	**ACC (%)**
Performance of TableCritic on GPT models								
GPT-3.5	w/Cot	60.2	48.6	74.0	86.8	63.8	82.2	69.3
	w/SC	65.8	52.0	77.4	86.8	63.8	82.2	71.3
	w/TableCritic	79.4$_{(\uparrow 13.6)}$	54.4$_{(\uparrow 2.4)}$	85.6$_{(\uparrow 8.2)}$	89.6$_{(\uparrow 2.8)}$	67.4$_{(\uparrow 3.6)}$	87.5$_{(\uparrow 5.3)}$	77.3($\uparrow$ 6.0)
GPT-4o	w/Cot	77.8	53.2	84.6	90.6	74.2	84.6	77.5
	w/SC	80.6	55.2	87.4	90.6	74.2	87.4	79.2
	w/TableCritic	83.8$_{(\uparrow 3.2)}$	57.4$_{(\uparrow 2.2)}$	94.4$_{(\uparrow 7.0)}$	93.2$_{(\uparrow 2.6)}$	75.8$_{(\uparrow 1.6)}$	90.8$_{(\uparrow 3.4)}$	82.6($\uparrow$ 3.4)
Performance of TableCritic on Llama models								
Llama3-8B	w/Cot	45.6	5.0	22.2	39.6	21.2	23.0	26.1
	w/SC	50.2	5.8	26.6	39.6	26.6	28.3	29.5
	w/TableCritic	46.0$_{(\downarrow 4.2)}$	7.2$_{(\uparrow 1.4)}$	31.2$_{(\uparrow 4.6)}$	44.6$_{(\uparrow 5.0)}$	43.8$_{(\uparrow 17.2)}$	42.1$_{(\uparrow 13.8)}$	35.8($\uparrow$ 6.3)
Llama3-70B	w/Cot	58.2	37.8	48.0	70.2	45.5	56.6	52.7
	w/SC	63.4	41.3	56.8	76.6	59.0	75.7	62.1
	w/TableCritic	73.0$_{(\uparrow 9.6)}$	53.8$_{(\uparrow 12.5)}$	82.2$_{(\uparrow 25.4)}$	74.6$_{(\downarrow 2.0)}$	59.5$_{(\uparrow 0.5)}$	78.3$_{(\uparrow 2.6)}$	70.2($\uparrow$ 8.1)

els to establish a comparative baseline: GPT-4o* (Achiam et al., 2023) and GPT-3.5 (Brown, 2020). Additionally, we evaluated the potential of smaller, open-source alternatives, particularly the LLaMA-3 variants with 8B and 70B parameters (Dubey et al., 2024). Consistent prompting strategies were maintained across all model variants to isolate framework effects

Furthermore, we compared the WTQ and TabFact results from other table-reasoning models. Specifically, the baselines can be divided into those based on the pre-LLMs and the LLMs. For pre-LLM methods, we select TAPEX (Q. Liu et al., 2022), TaCube (Zhou et al., 2022), ReasTAP (Zhao, Nan, Qi, Zhang, & Radev, 2022), OmniTab (Z. Jiang, Mao, He, Neubig, & Chen, 2022) and CABINET (Patnaik et al., 2024). For LLM methods, we included the approaches using the same underlying model (GPT-3.5-turbo), including Binder (Cheng et al., 2023), Dater (Ye et al., 2023), Text-to-SQL (Rajkumar, Li, & Bahdanau, 2022), Alter (H. Zhang, Ma, & Yang, 2024), Chain-of-Table (Z. Wang et al., 2024) and Mix-SC(T. Liu et al., 2023). Where methodologically appropriate, we incorporate published results from Chain-of-Table (GPT-3.5-turbo implementation) and pre-LLM baselines reported in Alter (H. Zhang et al., 2024) to optimize computational resource allocation.

Experimental Settings

Based on previous studies (Gou et al., 2023; Madaan et al., 2024), 2-3 rounds of self-correction yield most of the benefits. Therefore, we set the maximum number of self-criticism iterations to 3, stopping early if the answer remains unchanged for two consecutive corrections. To ensure consistent results, we few-shot prompt the model with 2 examples, apply a self-consistency technique with five sampling times for each output, and set the temperature parameter to 1.0 to enhance output diversity. Specifically, for GPT models, we utilized APIs directly without retraining. For the LLaMA model, the inference is conducted using vLLM (Kwon et al., 2023) on NVIDIA Tesla A100 40G GPUs.

Main Results

We initially conducted experiments to verify whether TableCritic can enhance the model's performance. Our experimental design encompassed three distinct settings designed to assess table understanding: (1) chain-of-thought(CoT) (Wei et al., 2022), (2) self-consistency (X. Wang et al., 2022), and (3) TableCritic. As delineated in Table 1, the experimental results demonstrate that our TableCritic pipeline significantly improves the model's capacity to interpret and understand tables. Our findings are as follows:

1. **TableCritic has consistently enhanced table understanding across all models evaluated.** As shown in Table 1, the implementation of TableCritic on the GPT-3.5 model resulted in a 13.6% enhancement in performance on the WTQ dataset, achieving an overall average improvement of 6.0% across all datasets. Comparatively, GPT-4o showed a 3.4% improvement, while the LLaMA 7B and 70B models demonstrated improvements of 6.3% and 8.1%, respectively. The improvement of all models underscores the significant impact of the TableCritic framework in enhancing the model's capacity to comprehend and interpret tabular data.

2. **TableCritic has outperformed all baseline methods, including both fine-tuned and non-fine-tuned approaches.** Table 2 presents a comparative analysis of various methods, focusing on their performance on the WTQ and TableFact datasets. Notably, the results demonstrate that TableCritic, our proposed method, outperforms other techniques, achieving the highest accuracy of 75.4% on WTQ and 89.6% on TableFact. Additionally, compared to existing methods,

* Due to limited budget, we use 4o-mini for the experiment.

Table 2: Comparation results with baseline methods on GPT-3.5. Underline denotes the second-best performance; **bold** denotes the best performance. ⋆ represents using the official denotation accuracy (Pasupat & Liang, 2015) for WTQ evaluation to align with these methods.

Methods	WTQ	TableFact	AVG.
♡ *Pre-LLM era*			
TAPEX (Q. Liu et al., 2022)	57.2	85.9	71.6
TaCube (Zhou et al., 2022)	60.8	-	60.8
ReasTAP (Zhao et al., 2022)	58.6	86.2	72.4
OmniTab (Z. Jiang et al., 2022)	62.7	-	62.7
CABINET (Patnaik et al., 2024)	69.1	-	69.1
♢ *LLMs era*			
Text-to-SQL (Rajkumar et al., 2022)	52.9	67.7	60.3
Binder (Cheng et al., 2023)	56.7	79.2	68.0
Dater (Ye et al., 2023)	52.8	78.0	65.4
Chain-of-Table (Z. Wang et al., 2024)	59.9	80.2	70.1
Alter (H. Zhang et al., 2024)	70.4	87.2	78.8
Mix-SC (T. Liu et al., 2023)	73.3	88.5	80.8
TableCritic(Ours)	**75.4**⋆	**89.6**	**82.5**

TableCritic achieves the highest average accuracy of 82.5%, indicating balanced performance across two different table-based tasks and demonstrating strong scalability and robustness.

Ablation analysis

In this section, we will conduct ablation studies to investigate the importance of tools and criticism.

Table 3: Ablation study of TableCritic on GPT-3.5

Settings	WTQ	TabFact	AVG.
TableCritic	**79.4**	**89.6**	**84.5**
w/o Criticism module	77.8(↓ 1.6)	88.6(↓ 1.0)	83.2(↓ 1.3)
w/o Tool Library	63.8(↓ 15.6)	86.2(↓ 3.4)	75.0(↓ 9.5)
+ Sub-table extractor	69.8(↓ 9.6)	86.4(↓ 3.2)	78.1(↓ 6.4)
+ Python code	71.6(↓ 7.8)	88.0(↓ 1.6)	79.8(↓ 4.7)

Impact of tools

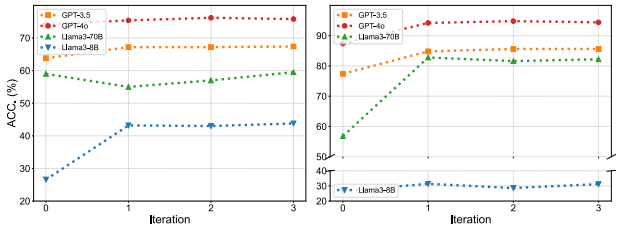
As presented in Table 3, the performance of TableCritic decreased by 9.5% on GPT-3.5 without using tools. Specifically, when not using Python code, the performance of GPT-3.5 decreased by 6.4%, and when not using sub-table extraction, the performance decreased by 4.7%, which demonstrates that both Python code and sub-table extraction are essential.

By leveraging multi-tool consistency and implementing a voting mechanism across tools, the model can select the most appropriate tool and answer for the current table, thereby enhancing overall performance.

Impact of criticism

As shown in Table 3, omitting the iterative process and directly invoking tools resulted in a 1.3% accuracy decline for GPT-3.5. The decline is attributed to the inherent limitations of the tools, which are incapable of resolving all issues (such as those related to semantics). The criticism module also allows for corrections based on feedback from these tools. Consequently, the removal of the criticism module results in a deterioration of model performance. Further analysis of the TabFact dataset reveals that 61.0% of answers generated by the initialization module required no additional tool utilization, as the criticism module confirmed their validity during the first iteration cycle, achieving an accuracy assessment of 87.5%. Through progressive iterations, an increasing proportion of issues were either corrected or validated through tool-assisted refinement processes. Notably, compared to direct tool invocation approaches, the criticism module delivered dual advantages: it not only enhanced overall system performance but also optimized computational resource allocation through strategic reduction of tool-calling overhead.

We further analyzed the effects of different criticism iterations on TabMWP and Scitab*. As illustrated in Fig.3, iterative refinements can lead to improvements, with the highest improvements observed on the TabMWP dataset, achieving an increase of 25.4% on Llama3-70B. For the GPT models, performance consistently improves from zero to three iterations, while for the Llama models, performance gains are irregular and less stable. This suggests that the criticism module performs better when paired with more powerful LLMs, consistent with the findings of Critic (Gou et al., 2023).



(a) Scitab* (b) TabMWP

Figure 3: Iterations on 2 example datasets.

Conclusion

Our study presents TableCritic, a framework that elevates table-based reasoning through LLM self-critique and adaptive tool orchestration. Aligned with functional computational models, TableCritic implements cognitive-inspired iterative refinement, dynamically guiding error correction through contextual tool selection. Experimental results demonstrate the framework's effectiveness and task-general adaptability, advancing functional AI systems that synergize cognitive principles with algorithmic transparency. This work bridges cognitive strategy formalization with practical AI development, particularly in structured data domains. We envision this methodology inspiring hybrid models that embed neurosymbolic constraints while maintaining computational efficiency across data modalities.

Acknowledgments

This work is partially supported by NSFC 62302503, NUDT Youth Independent Innovation Science Fund Project (Grant No. ZK23-15), and the Open Research Fund from State Key Laboratory of High Performance Computing of China Grant No.202401-09.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., ... others (2023). Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Aly, R., Guo, Z., Schlichtkrull, M. S., Thorne, J., Vlachos, A., Christodoulopoulos, C., ... Mittal, A. (2021). The fact extraction and verification over unstructured and structured information (feverous) shared task. In *Proceedings of the fourth workshop on fact extraction and verification (fever)* (pp. 1–13).

Brown, T. B. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. D. O., Kaplan, J., ... others (2021). Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Chen, W. (2022). Large language models are few (1)-shot table reasoners. *arXiv preprint arXiv:2210.06710*.

Chen, W., Chang, M.-W., Schlinger, E., Wang, W., & Cohen, W. W. (2020). Open question answering over tables and text. *arXiv preprint arXiv:2010.10439*.

Chen, W., Ma, X., Wang, X., & Cohen, W. W. (2022). Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*.

Chen, W., Wang, H., Chen, J., Zhang, Y., Wang, H., Li, S., ... Wang, W. Y. (2019). Tabfact: A large-scale dataset for table-based fact verification. *arXiv preprint arXiv:1909.02164*.

Chen, Z., Chen, W., Smiley, C., Shah, S., Borova, I., Langdon, D., ... others (2021). Finqa: A dataset of numerical reasoning over financial data. *arXiv preprint arXiv:2109.00122*.

Cheng, Z., Xie, T., Shi, P., Li, C., Nadkarni, R., Hu, Y., ... Yu, T. (2023). *Binding language models in symbolic languages.* Retrieved from https://arxiv.org/abs/2210.02875

Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., ... others (2024). The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Gehman, S., Gururangan, S., Sap, M., Choi, Y., & Smith, N. A. (2020). *Realtoxicityprompts: Evaluating neural toxic degeneration in language models.* Retrieved from https://arxiv.org/abs/2009.11462

Gou, Z., Shao, Z., Gong, Y., Shen, Y., Yang, Y., Duan, N., & Chen, W. (2023). Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., ... Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Computing Surveys*, *55*(12), 1–38.

Jiang, J., Zhou, K., Dong, Z., Ye, K., Zhao, W. X., & Wen, J.-R. (2023). Structgpt: A general framework for large language model to reason over structured data. *arXiv preprint arXiv:2305.09645*.

Jiang, Z., Mao, Y., He, P., Neubig, G., & Chen, W. (2022, July). OmniTab: Pretraining with natural and synthetic data for few-shot table-based question answering. In M. Carpuat, M.-C. de Marneffe, & I. V. Meza Ruiz (Eds.), *Proceedings of the 2022 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 932–942). Seattle, United States: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2022.naacl-main.68 doi: 10.18653/v1/2022.naacl-main.68

Kotonya, N., & Toni, F. (2020). Explainable automated fact-checking for public health claims. *arXiv preprint arXiv:2010.09926*.

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., ... Stoica, I. (2023). Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles* (pp. 611–626).

Lieto, A. (2021). *Cognitive design for artificial minds*. Routledge.

Liu, Q., Chen, B., Guo, J., Ziyadi, M., Lin, Z., Chen, W., & Lou, J. (2022). TAPEX: table pre-training via learning a neural SQL executor. In *The tenth international conference on learning representations, ICLR 2022, virtual event, april 25-29, 2022*.

Liu, T., Wang, F., & Chen, M. (2023). Rethinking tabular data understanding with large language models. *arXiv preprint arXiv:2312.16702*.

Lu, P., Peng, B., Cheng, H., Galley, M., Chang, K.-W., Wu, Y. N., ... Gao, J. (2024). Chameleon: Plug-and-play compositional reasoning with large language models. *Advances in Neural Information Processing Systems*, *36*.

Lu, P., Qiu, L., Chang, K.-W., Wu, Y. N., Zhu, S.-C., Rajpurohit, T., ... Kalyan, A. (2022). Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. *arXiv preprint arXiv:2209.14610*.

Lu, X., Pan, L., Liu, Q., Nakov, P., & Kan, M.-Y. (2023). Scitab: A challenging benchmark for compositional reasoning and claim verification on scientific tables. *arXiv preprint arXiv:2305.13186*.

Lu, X., Pan, L., Ma, Y., Nakov, P., & Kan, M.-Y. (2024). *Tart: An open-source tool-augmented framework for explainable table-based reasoning.* Retrieved from https://arxiv.org/abs/2409.11724

Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, S., ... others (2024). Self-refine: Iterative refinement with self-feedback. *Advances in Neural Informa-*

tion Processing Systems, *36*.

Mashaabi, M., Al-Khalifa, S., & Al-Khalifa, H. (2024). *A survey of large language models for arabic language and its dialects.* Retrieved from https://arxiv.org/abs/2410.20238

Pasupat, P., & Liang, P. (2015). Compositional semantic parsing on semi-structured tables. *arXiv preprint arXiv:1508.00305*.

Patnaik, S., Changwal, H., Aggarwal, M., Bhatia, S., Kumar, Y., & Krishnamurthy, B. (2024). CABINET: Content relevance-based noise reduction for table question answering. In *The twelfth international conference on learning representations,ICLR 2024, vienna, austria, may 7-11, 2024*.

Pourreza, M., & Rafiei, D. (2024). Din-sql: decomposed in-context learning of text-to-sql with self-correction. In *Proceedings of the 37th international conference on neural information processing systems.* Red Hook, NY, USA: Curran Associates Inc.

Rajkumar, N., Li, R., & Bahdanau, D. (2022). Evaluating the text-to-sql capabilities of large language models. *arXiv preprint arXiv:2204.00498*.

Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., ... Scialom, T. (2024). Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, *36*.

Shi, T., Zhao, C., Boyd-Graber, J., Daumé III, H., & Lee, L. (2020). On the potential of lexico-logical alignments for semantic parsing to sql queries. *arXiv preprint arXiv:2010.11246*.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., ... Zhou, D. (2022). Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Wang, Z., Zhang, H., Li, C.-L., Eisenschlos, J. M., Perot, V., Wang, Z., ... others (2024). Chain-of-table: Evolving tables in the reasoning chain for table understanding. *arXiv preprint arXiv:2401.04398*.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., ... others (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, *35*, 24824–24837.

Ye, Y., Hui, B., Yang, M., Li, B., Huang, F., & Li, Y. (2023). Large language models are versatile decomposers: Decomposing evidence and questions for table-based reasoning. In *Proceedings of the 46th international acm sigir conference on research and development in information retrieval* (pp. 174–184).

Zhang, H., Ma, Y., & Yang, H. (2024). Alter: Augmentation for large-table-based reasoning. *arXiv preprint arXiv:2407.03061*.

Zhang, Y., Henkel, J., Floratou, A., Cahoon, J., Deep, S., & Patel, J. M. (2023). Reactable: Enhancing react for table question answering. *arXiv preprint arXiv:2310.00815*.

Zhao, Y., Chen, L., Cohan, A., & Zhao, C. (2024). Tapera: Enhancing faithfulness and interpretability in long-form table qa by content planning and execution-based reasoning. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 12824–12840).

Zhao, Y., Nan, L., Qi, Z., Zhang, R., & Radev, D. (2022). Reastap: Injecting table reasoning skills during pre-training via synthetic reasoning examples. In *2022 conference on empirical methods in natural language processing, emnlp 2022*.

Zhou, F., Hu, M., Dong, H., Cheng, Z., Cheng, F., Han, S., & Zhang, D. (2022). Tacube: Pre-computing data cubes for answering numerical-reasoning questions over tabular data. In *Proceedings of the 2022 conference on empirical methods in natural language processing* (pp. 2278–2291).

Zhu, F., Lei, W., Huang, Y., Wang, C., Zhang, S., Lv, J., ... Chua, T.-S. (2021). Tat-qa: A question answering benchmark on a hybrid of tabular and textual content in finance. *arXiv preprint arXiv:2105.07624*.

Zhuang, Y., Yu, Y., Wang, K., Sun, H., & Zhang, C. (2023). Toolqa: A dataset for llm question answering with external tools. *Advances in Neural Information Processing Systems*, *36*, 50117–50143.