Do LLMs Behave as Claimed? Investigating How LLMs Follow Their Own Claims using Counterfactual Questions

Haochen Shi, Shaobo Li[*], Guoqing Chao, Xiaoliang Shi, Wentao Chen, Zhenzhou Ji

Harbin Institute of Technology (Weihai)
2022210159@stu.hit.edu.cn, {lishaobo,guoqingchao}@hit.edu.cn,
{2022210214, 2022210202}@stu.hit.edu.cn, jizhenzhou@hit.edu.cn

Abstract

Large Language Models (LLMs) require robust evaluation. However, existing frameworks often rely on curated datasets that, once public, may be accessed by newer LLMs. This creates a risk of data leakage, where test sets inadvertently become part of training data, compromising evaluation fairness and integrity. To mitigate this issue, we propose Behave as Claimed (BaC), a novel evaluation framework inspired by counterfactual reasoning. BaC constructs a "what-if" scenario where LLMs respond to counterfactual questions about how they would behave if the input were manipulated. We refer to these responses as claims, which are verifiable by observing the LLMs' actual behavior when given the manipulated input. BaC dynamically generates and verifies counterfactual questions using various few-shot in-context learning evaluation datasets, reducing their susceptibility to data leakage. Moreover, BaC provides a more challenging evaluation paradigm for LLMs. LLMs must thoroughly understand the prompt, the task, and the consequences of their responses to achieve better performance. We evaluate several LLMs and find that, while most perform well on the original datasets, they struggle with BaC. This suggests that LLMs usually fail to align their claims with their actual behavior and that high performance on standard datasets may be less stable than previously assumed.

1 Introduction

In-context learning is a fundamental capability of LLMs. To evaluate specific aspects of LLM performance, researchers publish datasets comprising domain-specific questions and corresponding ground-truth answers. Various datasets have been introduced, such as mathematics (Collins et al., 2023; Zhang et al., 2024a; Toshniwal et al., 2024), logical reasoning (Parmar et al., 2024; Patel et al.,

[*] Corresponding author

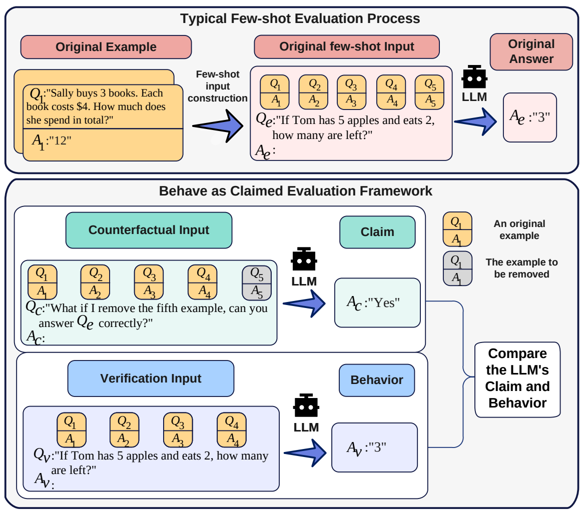

Typical Few-shot Evaluation Process
Original Example
Original few-shot Input
Original Answer
Q_1:"Sally buys 3 books. Each book costs $4. How much does she spend in total?"
A_1:"12"
Few-shot input construction
Q_1 A_1 Q_2 A_2 Q_3 A_3 Q_4 A_4 Q_5 A_5
Q_e:"If Tom has 5 apples and eats 2, how many are left?"
A_e:
LLM
A_e:"3"
Behave as Claimed Evaluation Framework
Counterfactual Input
Claim
Q_c:"What if I remove the fifth example, can you answer Q_e correctly?"
A_c:
A_c:"Yes"
Verification Input
Behavior
Q_v:"If Tom has 5 apples and eats 2, how many are left?"
A_v:
A_v:"3"
An original example
The example to be removed
Compare the LLM's Claim and Behavior

Figure 1: The comparison between the typical few-shot evaluation process and the proposed BaC framework. In BaC, the LLM claims it would answer Q_e correctly with the answer "Yes" and the behavior is "3" as the correct answer of Q_e. So, the LLM behaved as claimed.

2024), and causal reasoning (Jin et al., 2023a; Wang, 2024). A significant issue arises once a dataset is published: the samples from the dataset may be crawled from the Web and included in the training data of newer LLMs (Aiyappa et al., 2023), thereby undermining the reliability and fairness of the evaluation (Wei et al., 2023). For example, when newer LLMs outperform older ones, it becomes difficult to determine whether the improvement stems from the advances in the LLMs' capabilities or from the leakage of the published datasets during training (Samala et al., 2020a; Zylberberg et al., 2018; Wang et al., 2024a).

In this paper, we propose to evaluate LLMs' abilities using dynamically generated and verifiable counterfactual questions (Gill, 2020). As shown in Figure 1, during a typical few-shot prompting evaluation (Ma et al., 2023), we construct a few-shot input using original examples (question-answer pairs) and ask the LLM to respond to an additional question based on these demonstrations. The proposed Behave as Claimed (BaC) framework transforms the original few-shot input into a *counterfactual input*, prompting the LLM what its output would be if the last example were removed. The LLM's response to the counterfactual input is called its *Claim*. This Claim is verifiable by modifying the original input into a corresponding *verification input* (e.g., removing the last example as specified in the counterfactual question) and whether the LLM's actual behavior aligns with its claim.

The typical few-shot evaluation process in Figure 1 evaluates an LLM's in-context learning (ICL) ability (Dong et al., 2024; Li et al., 2024; Xu et al., 2024) by prompting the model to answer an additional question. However, if this additional question and its answer have been included in the LLM's training data, the LLM might rely on its parametric memory rather than leveraging its ICL ability to answer (Zhang et al., 2024b), leading to data leakage and undermining the evaluation's reliability (Zhang et al., 2024c).

BaC uses counterfactual inputs to create "what if" scenarios, requiring LLMs to predict changes in output as inputs vary, e.g, how the output would change if the last example were removed. The ground-truth answers for counterfactual inputs depend on actual model behaviors, which evolves dynamically during training and is almost impossible to be explicitly contained in the training data, making them resistant to data leakage. This distinguishing it from typical evaluation methods (Wang et al., 2024b; Chang et al., 2024b; Zhao et al., 2023; Zeng et al., 2024).

Moreover, BaC can also serve as a method to evaluate the counterfactual reasoning ability of LLMs. To predict the outcome of the assumed action, the LLM must understand the cause-effect relationship between changes in the input and the corresponding changes in the output, specifically the causal relationship between the input demonstrations and the additional question. Since counterfactual investigate alternative outcomes by asking what would have happened had some initial conditions been different, it can be challenging to determine a ground-truth answer to such questions. Because, in many cases, we cannot physically alter initial conditions and observe the resulting outcomes in the real world. The proposed BaC framework overcomes this limitation by focusing on counterfactual questions about the LLMs themselves, which can be empirically verified. As shown in Figure 1, we can create a verification input by removing the last example from the input. Then, we feed the verification input back into the LLM and observe its output. So, we can verify whether the LLM's behavior changes as predicted, making the counterfactual question empirically verifiable.

The contributions of this paper can be summarized as follows:

- **We propose BaC, a novel framework to assess whether LLMs behave as claimed.** Unlike conventional few-shot evaluations that focus on associative reasoning by requiring models to generate responses from a set of examples, BaC introduces a challenging "what-if" dimension. Ground-truth answers to these counterfactual scenarios are empirically derived from the behavior of the LLM itself, ensuring the correctness of ground truth without relying on static or heuristic assumptions.

- **BaC dynamically generates evaluation data in the form of counterfactual questions paired with verification inputs.** This process can mitigate the issue of data leakage, as ground-truth answers are generated during the evaluation process rather than pre-existing in static datasets. Furthermore, we design targeted experiments that intentionally create data leakage scenarios to empirically validate the effectiveness of the framework in addressing this critical issue.

- **Using BaC, we evaluate a wide range of LLMs**, including open-source models such as 7B and 13B, as well as more advanced models like GPT-3.5 and the GPT-4 models. Our findings reveal that, while some LLMs perform well on traditional metrics such as accuracy, they struggle to achieve comparable results on the proposed metrics, highlighting the limitations of existing evaluation methods and the importance of the BaC framework.

2 Related Work

Evaluation Framework For LLMs The evaluation of LLMs has garnered significant attention as their capabilities expand across diverse applications. For automatic evaluation, standardized metrics such as accuracy (Hendrycks et al., 2021), calibration (Guo et al., 2017), fairness (Hardt et al., 2016), and robustness (Zhu et al., 2023) are widely used to quantify model performance across various tasks. Meanwhile, human evaluation remains essential for subjective tasks like dialogue generation (Wang et al., 2023) and open-ended text synthesis (Dhamala et al., 2021), where aspects such as fluency, relevance, and safety require nuanced judgment. However, traditional automatic evaluation frameworks suffer from test data leakage, while human evaluation is labor-intensive and costly, posing challenges for scalable and reliable assessment (Chang et al., 2024a). To address these limitations, there is a growing need for automatic dynamic evaluation methods that can adapt to diverse tasks, mitigate data leakage risks, and provide more robust performance insights.

Counterfactual Reasoning Ability Evaluation Counterfactual reasoning is a fundamental aspect of human cognition (Gill, 2020; Mercier and Sperber, 2011). For LLMs, it plays a crucial role in understanding hypothetical scenarios and reasoning about causal relationships. While LLMs have shown proficiency in associative reasoning, their ability to perform counterfactual reasoning remains an open research challenge (Liu et al., 2024b). Several studies have been proposed to evaluate LLMs' counterfactual reasoning ability. (Hobbhahn et al., 2022) find that LLMs can be misled by superficial prompt variations, suggesting a reliance on surface-level patterns rather than genuine causal understanding. Similarly, (Yin et al., 2023) show that LLMs struggle with causal discovery and precise effect estimation. (Zecevic et al., 2023) demonstrate that LLMs often rely on correlations rather than true causal inference, limiting their ability to generate counterfactual predictions. These findings emphasize that LLMs still lack a robust understanding of causality, which is essential for tasks requiring reliable decision-making and hypothetical scenario evaluation (Jin et al., 2023b).

A major challenge in counterfactual evaluation is obtaining ground-truth answers, especially when interventions are hard to implement. The proposed BaC framework addresses this by deriving ground-truth answers from the model's own outputs. This self-referential approach eliminates the need for external annotations or real-world interventions, making BaC a scalable and efficient framework for automatic evaluation.

3 Methods

The proposed Behave as Claimed (BaC) framework consists of three key components: *counterfactual input*, *verification input*, and *evaluation metrics*. In the counterfactual input stage, we construct a hypothetical scenario in which the original few-shot input is modified, requiring the model to predict its expected behavior under these altered conditions. Next, we generate the verification input, which implements the hypothetical scenario to evaluate the model's actual output when provided with the modified input. This step evaluates whether the model's responses to counterfactual inputs align with its behavior when given the altered input. Based on the model's answers to these two types of questions, we define three metrics to quantify the model's performance:

- Counterfactual Accuracy (CFA): Measures whether the model behaves as claimed;
- In-Context Learning Accuracy (ICL-A): Evaluates whether the model not only behaves as claimed but also provides the correct answer;

3.1 Counterfactual Input

This section provides a detailed explanation of counterfactual input. First, we introduce the concept of counterfactual input and its role in evaluating LLMs. Then, we describe the process of constructing counterfactual inputs from original examples.

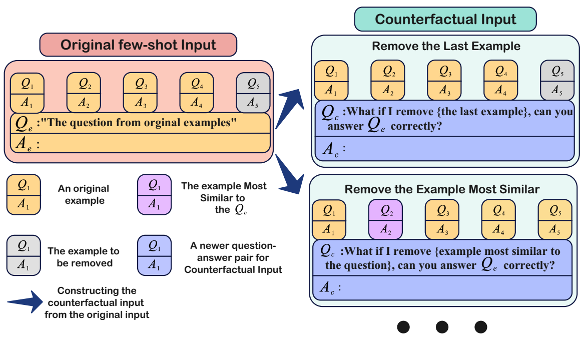
3.1.1 The Details of Counterfactual Input

Counterfactual input refers to a hypothetical modification of the original few-shot input, designed to evaluate how a model predicts its expected behavior under altered conditions. These modifications challenge the model to determine whether and how its response should change.

In the *Behave as Claimed (BaC)* framework, counterfactual inputs play a central role in evaluating an LLM's counterfactual reasoning abilities. By modifying the original input, BaC examines whether the model can adjust its responses. This approach ensures a rigorous and fair evaluation of the model's capacity to generalize and adapt to hypothetical changes.

Typically, counterfactual questions are structured clearly, such as asking: "What will your answer be if a specific part *X* is removed from the text?" Such carefully formulated questions require the model to recognize causal dependencies between input and output and adjust its reasoning accordingly. By evaluating the model's ability to respond to these alterations, BaC provides insights into its capacity for counterfactual reasoning (Li et al., 2023).

3.1.2 Constructing the Counterfactual Input from the Original Example



Original few-shot Input
Q_1 A_1 Q_2 A_2 Q_3 A_3 Q_4 A_4 Q_5 A_5
Q_e :"The question from orginal examples"
A_e :
Counterfactual Input
Remove the Last Example
Q_c :What if I remove {the last example}, can you answer Q_e correctly?
A_c :
Remove the Example Most Similar
Q_c :What if I remove {example most similar to the question}, can you answer Q_e correctly?
A_c :
An original example
The example Most Similar to the Q_e
The example to be removed
A newer question-answer pair for Counterfactual Input
Constructing the counterfactual input from the original input

Figure 2: The process of generating counterfactual inputs from the original few-shot input. The original input is displayed on the left side and consists of five question-answer pairs $(Q_1, A_1), (Q_2, A_2), \ldots, (Q_5, A_5)$. The right side displays two examples of counterfactual inputs constructed by selectively removing either the last example or the example most similar to the question. Other possible counterfactual inputs are omitted for brevity (indicated by the ellipsis).

To generate a counterfactual input, we start with an original set of question-answer pairs

$$\{(Q_1, A_1), (Q_2, A_2), \ldots, (Q_5, A_5)\}$$

and selectively modify its elements. This process tests the model's ability to reason under altered conditions.

As shown in Figure 2, the original few-shot input (left) consists of five examples, each represented by a question-answer pair. On the right, various counterfactual inputs are constructed by altering the original input and posing a counterfactual question to challenge the model's reasoning. We designed five distinct scenarios for constructing counterfactual inputs:

(1) Remove the Last Example: The last example is excluded (gray box), and the model is asked: "What if I remove the last example, can you answer Q_e correctly?"

(2) Remove the Most Similar Example (TF-IDF): Utilizing the TF-IDF algorithm, the most similar example is identified and removed, prompting:"What if I remove the example most similar to the question (using TF-IDF), can you answer Q_e correctly?"

(3) Remove the Most Similar Example (Cosine Similarity): Leveraging cosine similarity, the most similar example is identified and excluded, prompting the question:"What if I remove the example most similar to the question (using cosine similarity), can you answer Q_e correctly?" In our implementation, cosine similarity is computed over embeddings generated by the all-MiniLM-L6-v2 model from the Sentence-BERT library (Reimers and Gurevych, 2019).

(4) Remove the Last and Second Last Examples: Both the last and second last examples are removed, prompting: "What if I remove the last and second last examples, can you answer Q_e correctly?"

(5) Remove Examples Randomly: Two examples are removed randomly from the original set, prompting the model with:"What if I remove randomly selected two examples, can you answer Q_e correctly?"

All counterfactual questions follow the same structured template: "What if I remove M, can you answer Q_e correctly?", where M varies depending on the modification method. Figure 2 illustrates this process, showcasing different counterfactual inputs derived from a single original input.

To motivate the design of these removal strategies, we highlight two considerations. First, the removal of the "last example(s)" is motivated by prior findings on positional bias in LLMs (Liu et al., 2024a), which demonstrate that examples located near the beginning or end of a context often exert disproportionate influence on model reasoning. Examining the impact of excluding these final examples therefore allows us to test whether models can adapt when the most immediately influential demonstrations are removed. Second, in the case of the "most similar example," the rationale is that LLMs frequently rely on semantically related demonstrations to anchor their predictions. Eliminating the example most similar to the query—identified through TF-IDF or cosine similarity—directly challenges this reliance. While in rare cases the most similar example may coincide with the last one, in our design each counterfactual scenario is treated independently, and such overlaps occur infrequently enough that they do not undermine the distinct purposes of positional versus semantic similarity removal.

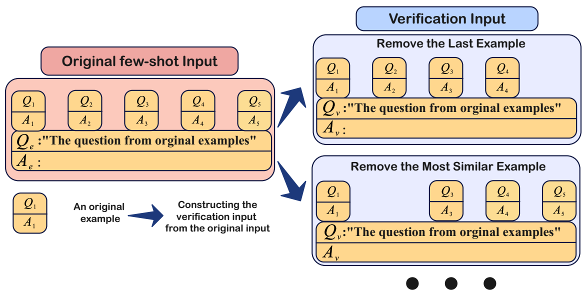

Verification Input
Remove the Last Example
Original few-shot Input
Q1 A1 Q2 A2 Q3 A3 Q4 A4 Q5 A5
Qe :"The question from orginal examples"
Ae :
Qv :"The question from orginal examples"
Av :
Remove the Most Similar Example
An original example
Constructing the verification input from the original input

Figure 3: The process of constructing verification inputs from the original few-shot input. The illustration on the left depicts the original input. The right side illustrates two distinct verification inputs generated by removing either the last example or the example most similar to the question. Additional verification inputs are omitted for brevity (indicated by the ellipsis).

3.2 Verification Input

This section provides a detailed explanation of the verification input. First, we describe the verification input and its role in validating the model's behavior in response to counterfactual scenarios. Then, we outline the process of constructing verification input from the original example.

3.2.1 The Details of Verification Input

In the *Behave as Claimed (BaC)* framework, verification input plays a crucial role in evaluating the behaved as claimed ability of counterfactual reasoning. By modifying the original input and observing the model's actual response, BaC examines whether the model's behavior matches its stated claims under hypothetical conditions.

This process provides a robust validation of the model's counterfactual reasoning performance, ensuring that its responses are not just declarative but also consistent with its own outputs when confronted with modified inputs.

3.2.2 Constructing the Verification Input from the Original Example

In the *Behave as Claimed (BaC)* framework, verification input serves as an empirical test to assess whether an LLM's counterfactual claims align with its actual behavior. We generate verification inputs by modifying the original few-shot input using five distinct methods, each corresponding to a specific counterfactual scenario. As illustrated in Figure 3, these modifications implement the conditions described in the counterfactual input while keeping identical to :

(1) Remove the Last Example: The last example is removed, reducing the number of examples from five to four.

(2) Remove the Most Similar Example (TF-IDF): The example identified as most similar to using the TF-IDF algorithm is excluded from the verification input.

(3) Remove the Most Similar Example (Cosine Similarity): The example identified as most similar to using cosine similarity is removed.

(4) Remove the Last and Second Last Examples: Both the last two examples are excluded, leaving three examples.

(5) Remove Examples Randomly: Randomly selected two examples from the original set are removed, providing a randomized scenario to test the model's generalization.

By comparing the predictions on the counterfactual input with actual outputs on the corresponding verification input, we can evaluate whether the model's reasoning is consistent and reliable. Without this verification step, ensuring the correctness of the ground-truth answers to the counterfactual inputs would be challenging.

3.3 Counterfactual Metrics Evaluation

3.3.1 Counterfactual Accuracy (CFA)

The Counterfactual Accuracy (CFA) is a metric that measures how accurately the model responds to counterfactual questions when compared to the correct or expected answer. It indicates whether the model can reason about the hypothetical change introduced in the counterfactual scenario and adjust its output accordingly. Table 1 illustrates how we calculate the Counterfactual Accuracy (CFA) in detail. In Table 1, A_c is the LLM's answer to the counterfactual input we constructed, which we define in this paper as the "Claim". "A_v is correct" indicates the accuracy of the LLM's answer to the verification input compared to the correct answer. The CFA determines whether the LLM "Behaves As Claimed" based on whether both "A_c is Yes" and "A_v is correct" are either "Yes" and "True", or "No" and "False". Lastly, ICL-A refers to whether the LLM correctly answers the verification input, assessing its in-context learning ability.

Specifically, a response is considered correct if: 1. When the verification answer (A_v) is correct, the model's counterfactual answer (A_c) must affirm this (i.e., A_c = Yes). 2. When the verification answer (A_v) is incorrect, the model's counterfactual answer (A_c) must acknowledge this (i.e.,

	Outcomes			Metrics	
	A_e is correct	A_c is Yes	A_v is correct	CFA	ICL-A
1	True	Yes	True	Behave As Claimed	✓
2	True	Yes	False	Behave Contrary To Claimed	✗
3	True	No	True	Behave Contrary To Claimed	✗
4	True	No	False	Behave As Claimed	✗
5	False	Yes	True	Behave As Claimed	✓
6	False	Yes	False	Behave Contrary To Claimed	✗
7	False	No	True	Behave Contrary To Claimed	✗
8	False	No	False	Behave As Claimed	✗

Table 1: Visualization of the determination process for CFA and ICL-A for each example, based on all possible combinations of the model's response correctness within the BaC framework, incorporating both counterfactual and verification inputs. Notably, ICL-A deliberately integrates both correctness on the verification input and consistency with the model's self-assessment, since our goal is to capture a stricter notion of in-context learning beyond either component alone.

$A_c =$ No).

The CFA metric is then calculated as:

$$\text{CFA} = \frac{\text{BAC\_count}}{N}, \tag{1}$$

where BAC_count represents the number of instances satisfying the "Behave as Claimed" condition, and N is the total number of evaluated instances.

From Table 1, the "Behave as Claimed" cases correspond to rows 1, 4, 5, and 8, illustrating the conditions under which the model's responses align with expected counterfactual reasoning ability. A higher CFA score reflects stronger counterfactual reasoning abilities, highlighting the model's capacity to respond to hypothetical input modifications and accurately predict their effects.

3.3.2 In-context Learning Accuracy (ICL-A)

The proposed BaC framework also assesses the model's in-context learning ability (ICL-A) under a more challenging setting. The ICL-A score is calculated as the proportion of cases where the model correctly answers the verification input (A_v) and simultaneously affirms its ability to do so in the counterfactual input ($A_c =$ Yes). This ensures that the model's self-assessment aligns with its actual in-context reasoning capability.

Mathematically, the ICL-A metric is calculated as:

$$\text{ICL-A} = \frac{\text{ICL-A\_count}}{N}, \tag{2}$$

where ICL-A_count represents the number of instances where both: 1. The model's answer to the verification input (A_v) is correct. and 2. The model's response to the counterfactual input (A_c) is "Yes" (indicating confidence in its ability to answer correctly).

From Table 1, A_e represents the model's in-context learning ability in the original few-shot evaluation, while A_v reflects its ability under the modified few-shot input in the BaC framework. Aligning A_e and A_v offers a more comprehensive and challenging assessment of the model's ICL capabilities.

For example, in Table 1, rows 2 and 4 show cases where A_e is correct, but A_v is incorrect, while rows 5 and 8 show cases where A_e is incorrect, but A_v is correct. These variations provide deeper insights into how the model's performance is influenced by contextual modifications. A significant drop in accuracy when removing the most similar example suggests that the model heavily relies on contextual similarity for in-context learning (ICL).

3.3.3 Behave as Claimed Confidence Level

Interestingly, based on the A_c values in Table 1, we can calculate the Confidence Level (CL), which quantifies the model's "confidence" when responding to counterfactual inputs. The formula for this calculation is as follows:

$$\text{CL} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}(A_c = \text{True}) \tag{3}$$

The detailed results are provided in Appendix A.2.4

3.3.4 Connections to the Causality Framework

Pearl's Ladder of Causation classifies causal reasoning into three hierarchical levels: (i) Association, (ii) Intervention, and (iii) Counterfactuals (Gill, 2020). The proposed BaC can align with the counterfactual level of Pearl's framework by testing whether LLMs can reason about hypothetical modifications to their input and predict how their responses should change accordingly. Unlike traditional few-shot evaluations that primarily assess associative learning, BaC requires models to anticipate and verify the consequences of their own claims, making it a more rigorous test of their causal reasoning abilities. By empirically verifying counterfactual behavior, BaC provides insights into whether LLMs truly "understand" causal dependencies.

Models	Remove the Last Example				Remove the Most Similar Example *(C)*				Remove the Most Similar Example *(T)*			
	GSM8K		TriviaQA		GSM8K		TriviaQA		GSM8K		TriviaQA	
	CFA	ICL-A	CFA	ICL-A	CFA	ICL-A	CFA	ICL-A	CFA	ICL-A	CFA	ICL-A
Mistral-7B	0.3798	0.3798	**0.6640**	**0.6638**	0.3475	0.3374	**0.6606**	**0.6606**	0.3472	0.3472	**0.6613**	**0.6562**
Llama3-8B	**0.5428**	0.0000	0.3287	0.0003	**0.5372**	0.0000	0.3304	0.0002	0.5663	0.0000	0.3306	0.0002
Llama3.1-8B	0.5216	0.0000	0.3389	0.0001	0.5302	0.0000	0.3383	0.0000	0.5406	0.0000	0.3414	0.0002
Yi-1.5-6B	0.5095	**0.4897**	0.5117	0.3609	0.5073	**0.5072**	0.5116	0.4109	0.5057	0.5041	0.4994	0.3849
Qwen2-7B	0.2654	0.0394	0.4573	0.1358	0.3168	0.1221	0.4905	0.2470	**0.6975**	**0.6709**	0.5294	0.4110
Qwen2.5-7B	0.4397	0.2039	0.4463	0.0001	0.5288	0.4253	0.4444	0.0000	0.6141	0.5527	0.4454	0.0000
InternLM2.5-7B	0.4154	0.0091	0.4372	0.930	0.4281	0.0098	0.4439	0.0941	0.6080	0.0043	0.4395	0.0000
Qwen2.5-13B	0.4511	0.0061	0.4463	0.0057	0.4778	0.0067	0.4345	0.0075	0.4576	0.0024	0.4376	0.0047
Yi-1.5-6B-*CoT*	0.5201	0.1873	0.5155	0.1456	0.5163	0.1457	0.5042	0.1286	0.5265	0.1793	0.5167	0.2003
Llama3.1-8B-*CoT*	0.5216	0.0000	0.3504	0.0000	0.5504	0.0000	0.3775	0.0000	0.5368	0.0000	0.3417	0.0000
Qwen2.5-7B-*CoT*	0.4567	0.2103	0.4352	0.0001	0.4357	0.2286	0.4329	0.0001	0.4892	0.2768	0.4589	0.0001
GPT-4o	0.7718	0.5178	0.8052	0.5045	0.7953	0.5086	0.7854	0.5023	0.7749	0.5203	0.7965	0.4905
GPT-4.1	**0.8264**	0.0000	**0.8265**	0.5061	**0.8357**	0.5042	**0.8265**	0.5208	**0.8348**	0.5221	**0.8216**	0.5307

Table 2: Performance of various LLMs across different counterfactual input generation methods: (1) Remove the Last Example, (2) Remove the Most Similar Example *(Cosine)*, and (3) Remove the Most Similar Example *(TF-IDF)*. CFA and ICL-A are the evaluation metrics proposed by BaC, measuring whether LLMs behave as claimed and can answer few-shot questions consistently. *CoT* refers to constructing the questions using the Chain-of-Thought reasoning approach. The results show no substantial improvement in CFA or ICL-A. The results for generation methods (4) and (5) are presented in Table 7. Bolded values indicate the best performance among models of the same parameter scale.

4 Experiment

4.1 Evaluation Setup

We evaluate our framework using GSM8K and TriviaQA, two widely used benchmarks for testing LLMs' reasoning abilities. We selected these datasets because GSM8K's focus on multi-step problem-solving aligns with our counterfactual reasoning framework, while TriviaQA allows us to evaluate model performance on knowledge-intensive tasks. Additionally, both datasets are well-represented in the LLM research community and included in the Hugging Face LLM leaderboard, ensuring reliable benchmarking and reproducibility.

To assess the effect of reasoning augmentation, we applied Chain-of-Thought (CoT) prompting (Wei et al., 2022; Kojima et al., 2022) to three representative models. An example of our CoT-style prompt is as follows:

> These are the five few-shot examples to help you answer the following question. Now, think step by step about how each of the five examples helps you answer this question: *[Question]"...*

4.2 Results

In this section, we present the results of evaluating several LLMs using the BaC framework. The evaluation primarily employs two proposed metrics: Counterfactual Accuracy (CFA) and In-Context Learning Accuracy (ICL-A). CFA assesses whether the LLM behaves as claimed, regardless of whether the final answer to the original question is correct. In contrast, ICL-A evaluates whether the LLM not only behaves as claimed but also provides the correct final answer. The models are evaluated across three counterfactual scenarios: (1) Remove the Last Example, (2) Remove the Most Similar Example *cosine similarity based*, and (3) Remove the Most Similar Example *TF-IDF based*. The results are summarized in Tables 2.

Conterfactual Ability Evaluation CFA evaluates a model's accuracy in responding to counterfactual inputs by comparing its output to its actual behavior, thereby assessing whether the LLM behaves as claimed. As shown in Table 2, the evaluated LLMs achieve relatively low CFA scores, averaging 0.5223 across all models and datasets. This result highlights the limited general counterfactual reasoning capabilities of these models in understanding and reflecting on their own behavior. Such findings raise a critical question about the potential for LLMs to develop a degree of self-awareness, particularly their ability to comprehend and reason about the implications of their own claims.

Across all evaluated LLMs, a consistent trend emerges: more resource-intensive models, outperform smaller models such as Mistral-7B and Qwen2-7B, on CFA. This highlights the critical need for further advancements in model architectures and training methodologies to enhance selfreasoning capabilities. Interestingly, LLMs perform worse on CFA when the removed example is the last one in the input sequence (averaging 0.5093 on GSM8K) compared to scenarios where the removed example is identified as the most similar using cosine similarity or TF-IDF (averaging 0.5775 on GSM8K). This observation suggests that LLMs may rely more heavily on the structural position of examples within a context than on their semantic similarity to the query.

In-Context Learning Ability Evaluation ICL-A requires LLMs to not only answer the original task correctly but also behave as claimed. As shown in Table 2, certain models, such as Llama3-8B and Llama3.1-8B, struggle to achieve a meanful ICL-A score, demonstrating that ICL-A is a more challenging metric compared to CFA. This difficulty underscores the inherent challenge of ensuring both alignment with claimed behavior and correctness in responses across original tasks. Establishing a more robust benchmark for evaluating LLMs on their ICL-A performance could provide better guidance for future research and model development.

Even the more advanced LLMs fail to achieve consistently high ICL-A scores. This suggests that even advanced models struggle to generalize effectively to counterfactual reasoning tasks that incorporate verification. These results highlight a critical limitation: **LLMs do not yet fully understand the causal relationships between inputs and outputs, nor do they exhibit self-awareness in reasoning about their own behavior**.

4.3 Empirical Verification of the BaC's Robustness to Data Leakage

Mechanically, BaC is inherently robust to data leakage due to its dynamic nature, whereby the ground-truth responses for counterfactual inputs depend on the continuously evolving behavior of the model rather than fixed, static answers. This dynamic property inherently restricts the precise replication of verification data within the training datasets.

To empirically validate the effectiveness of the proposed metrics, CFA and ICL-A, in addressing data leakage issues, we conducted targeted experiments that deliberately introduced test data into the training dataset to create explicit data leakage. Across these experiments, the proportion of leaked test data was systematically increased from 0% to 25%, and subsequently to 50%, thereby simulating scenarios of progressively worsening data leakage. If a metric does not consistently increase as the extent of leakage information grows, it suggests that the metric is less influenced by data leakage (Samala et al., 2020b).

We fine-tuned the LLMs (Llama3-8B and InternLM2.5-7B) on the curated training data of GSM8K and TriviaQA using QLoRA (Dettmers et al., 2023), with a batch size of 16 and a learning rate of 1e-5. The same number of training steps was employed while varying the percentages of leaked test data, as previously described. The fine-tuned LLMs were subsequently evaluated on both datasets. The evaluation was conducted using the proposed metrics, CFA and ICL-A, alongside the overall 5-shot Accuracy (ACC).

The results are presented in Table 3. It can be observed that the traditional metric ACC consistently increases as the level of data leakage rises. In contrast, CFA and ICL-A remain largely stable across all data leakage scenarios, with their values in some cases remaining unchanged. This observation indicates that the proposed metrics are not sensitive to contamination in the training set. Conversely, the upward trend in ACC scores as the leakage level increases from 0% to 25% and then to 50% confirms that ACC is more susceptible to the effects of data leakage. This discrepancy underscores the robustness of BaC metrics in assessing true model generalization and demonstrates the efficacy of the BaC framework in mitigating data leakage biases during LLM evaluation.

To complement the above fine-tuning experiments, we additionally design a loss-based diagnostic metric, $\Delta 5$, to further evaluate the robustness of CFA and ICL-A against data leakage. The intuition is that, under normal conditions, reducing the number of in-context examples should degrade model performance, whereas anomalous stability or improvement may indicate reliance on memorized answers. This metric is principled and interpretable in a manner comparable to established evaluation metrics such as BLEURT (Sellam et al., 2020)and BERTScore (Zhang et al., 2020), which similarly ground their design in intuitive expectations of model behavior. Due to space constraints, detailed definitions, results, and discussions of $\Delta 5$ are provided in Appendix Appendix A.2.6, but we reference it here to clarify its role in strengthening the empirical validation of BaC's robustness to leakage.

Models	Leaked Percentage	GSM8K			TriviaQA		
		CFA	ICL-A	ACC	CFA	ICL-A	ACC
Llama3-8B	0%	0.5919	0.0000	0.5905	0.5648	0.0000	0.6689
	25%	0.5914	0.0000	0.5913	0.5762	0.0000	0.6796
	50%	0.5853	0.0000	0.5996	0.5830	0.0000	0.6874
InternLM2.5-7B	0%	0.4255	0.1235	0.5728	0.4481	0.1654	0.5711
	25%	0.4197	0.1456	0.6141	0.4394	0.1719	0.5822
	50%	0.4206	0.1349	0.6428	0.4454	0.1786	0.5976

Table 3: CFA, ICL-A, and 5-shot accuracy (ACC) on GSM8K and TriviaQA with different percentages of leaked test data.

5 Conclusion

In this paper, we introduced the Behave as Claimed (BaC) framework, a novel approach for evaluating LLMs using dynamically generated counterfactual questions and answers derived from existing few-shot evaluation datasets. By dynamically generating evaluation data rather than relying on static datasets, BaC mitigates the risk of data leakage, ensuring a more robust and fair evluation. BaC also introduces two new metrics: Counterfactual Ability (CFA) and In-Context Learning Ability (ICL-A), which quantify a model's performance in counterfactual reasoning and in-context learning in a more challenging way, respectively. Our evaluation results show that even LLMs are good enough on some complex tasks but still lack a deeper understanding of how changes in input could affect their outputs, limiting their ability to handle counterfactual reasoning tasks effectively. These findings highlight the need for more advanced evaluation methodologies beyond standard few-shot assessments to better measure the reasoning capabilities of LLMs.

Limitations

The proposed framework offers a novel approach to evaluating LLMs through dynamically generated counterfactual questions. However, it has certain limitations. For example, BaC has not yet been applied to long-term or multi-turn interactions, which are essential for evaluating models in more complex conversational settings. Furthermore, the framework presumes that models can reliably interpret and respond to counterfactual queries, an assumption that may not hold if the evaluated LLM's overall capability is inadequate. Future research should prioritize refining counterfactual generation techniques, broadening the evaluation scope, and exploring BaC's applicability to a wider range of LLMs and tasks, including code generation and commonsense reasoning.

Acknowledgments

This research was funded by National Natural Science Foundation of China under Grant No. 62406087, Natural Science Foundation of Shandong Province under Grant No. ZR2024QF139, State Key Laboratory of Processors (ICT, CAS) under Grant No. CLQ202406.

References

Rachith Aiyappa, Jisun An, Haewoon Kwak, and Yong-Yeol Ahn. 2023. Can we trust the evaluation on chatgpt? *CoRR*, abs/2303.12767.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. 2024a. A survey on evaluation of large language models. *ACM Trans. Intell. Syst. Technol.*, 15(3):39:1–39:45.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2024b. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45.

Katherine M. Collins, Albert Q. Jiang, Simon Frieder, Lionel Wong, Miri Zilka, Umang Bhatt, Thomas Lukasiewicz, Yuhuai Wu, Joshua B. Tenenbaum, William Hart, Timothy Gowers, Wenda Li, Adrian Weller, and Mateja Jamnik. 2023. Evaluating language models for mathematics through interactions. *CoRR*, abs/2306.01694.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. In *NeurIPS*.

Jwala Dhamala, Tony Sun, Varun Kumar, Satyapriya Krishna, Yada Pruksachatkun, Kai-Wei Chang, and Rahul Gupta. 2021. BOLD: dataset and metrics for measuring biases in open-ended language generation. In *FAccT*, pages 862–872. ACM.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Baobao Chang, Xu Sun, Lei Li, and Zhifang Sui. 2024. A survey on in-context learning. In *EMNLP*, pages 1107–1128. Association for Computational Linguistics.

Karamjit S. Gill. 2020. Pearl, judea and mackenzie, dana: The book of why: the new science of cause and effect (2018). *AI Soc.*, 35(3):767–768.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.

Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. In *NIPS*, pages 3315–3323.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *ICLR*. OpenReview.net.

Marius Hobbhahn, Tom Lieberum, and David Seiler. 2022. Investigating causal understanding in llms. In *NeurIPS ML Safety Workshop*.

Zhijing Jin, Yuen Chen, Felix Leeb, Luigi Gresele, Ojasv Kamal, Zhiheng Lyu, Kevin Blin, Fernando Gonzalez Adauto, Max Kleiman-Weiner, Mrinmaya Sachan, and Bernhard Schölkopf. 2023a. Cladder: A benchmark to assess causal reasoning capabilities of language models. In *NeurIPS*.

Zhijing Jin, Yuen Chen, Felix Leeb, Luigi Gresele, Ojasv Kamal, LYU Zhiheng, Kevin Blin, Fernando Gonzalez Adauto, Max Kleiman-Weiner, Mrinmaya Sachan, et al. 2023b. Cladder: Assessing causal reasoning in language models. In *Thirty-seventh conference on neural information processing systems*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *NeurIPS*.

Jiaxuan Li, Lang Yu, and Allyson Ettinger. 2023. Counterfactual reasoning: Testing language models' understanding of hypothetical scenarios. In *ACL (2)*, pages 804–815. Association for Computational Linguistics.

Tianle Li, Ge Zhang, Quy Duc Do, Xiang Yue, and Wenhu Chen. 2024. Long-context llms struggle with long in-context learning. *CoRR*, abs/2404.02060.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024a. Lost in the middle: How language models use long contexts. *Trans. Assoc. Comput. Linguistics*, 12:157–173.

Xiaoyu Liu, Paiheng Xu, Junda Wu, Jiaxin Yuan, Yifan Yang, Yuhang Zhou, Fuxiao Liu, Tianrui Guan, Haoliang Wang, Tong Yu, Julian J. McAuley, Wei Ai, and Furong Huang. 2024b. Large language models and causal inference in collaboration: A comprehensive survey. *CoRR*, abs/2403.09606.

Huan Ma, Changqing Zhang, Yatao Bian, Lemao Liu, Zhirui Zhang, Peilin Zhao, Shu Zhang, Huazhu Fu, Qinghua Hu, and Bingzhe Wu. 2023. Fairness-guided few-shot prompting for large language models. *Advances in Neural Information Processing Systems*, 36:43136–43155.

Hugo Mercier and Dan Sperber. 2011. Why do humans reason? arguments for an argumentative theory. *Behavioral and brain sciences*, 34(2):57–74.

Mihir Parmar, Nisarg Patel, Neeraj Varshney, Mutsumi Nakamura, Man Luo, Santosh Mashetty, Arindam Mitra, and Chitta Baral. 2024. Towards systematic evaluation of logical reasoning ability of large language models. *CoRR*, abs/2404.15522.

Nisarg Patel, Mohith Kulkarni, Mihir Parmar, Aashna Budhiraja, Mutsumi Nakamura, Neeraj Varshney, and Chitta Baral. 2024. Multi-logieval: Towards evaluating multi-step logical reasoning ability of large language models. *arXiv preprint arXiv:2406.17169*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP/IJCNLP (1)*, pages 3980–3990. Association for Computational Linguistics.

Ravi K. Samala, Heang-Ping Chan, Lubomir M. Hadjiiski, and Sathvik Koneru. 2020a. Hazards of data leakage in machine learning: a study on classification of breast cancer using deep neural networks. In *Medical Imaging: Computer-Aided Diagnosis*, volume 11314 of *SPIE Proceedings*. SPIE.

Ravi K. Samala, Heang-Ping Chan, Lubomir M. Hadjiiski, and Sathvik Koneru. 2020b. Hazards of data leakage in machine learning: a study on classification of breast cancer using deep neural networks. In *Computer-Aided Diagnosis*, volume 11314 of *SPIE Proceedings*. SPIE.

Thibault Sellam, Dipanjan Das, and Ankur P. Parikh. 2020. BLEURT: learning robust metrics for text generation. In *ACL*, pages 7881–7892. Association for Computational Linguistics.

Shubham Toshniwal, Ivan Moshkov, Sean Narenthiran, Daria Gitman, Fei Jia, and Igor Gitman. 2024. Openmathinstruct-1: A 1.8 million math instruction tuning dataset. In *NeurIPS*.

Hongru Wang, Rui Wang, Fei Mi, Yang Deng, Zezhong Wang, Bin Liang, Ruifeng Xu, and Kam-Fai Wong. 2023. Cue-cot: Chain-of-thought prompting for responding to in-depth dialogue questions with llms. In *EMNLP (Findings)*, pages 12047–12064. Association for Computational Linguistics.

Jeffrey G Wang, Jason Wang, Marvin Li, and Seth Neel. 2024a. Pandora's white-box: Increased training data leakage in open llms. *CoRR*, abs/2402.17012. Withdrawn.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024b. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Zeyu Wang. 2024. Causalbench: A comprehensive benchmark for evaluating causal reasoning capabilities of large language models. In *Proceedings of the 10th SIGHAN Workshop on Chinese Language Processing (SIGHAN-10)*, pages 143–151.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*.

Tianwen Wei, Liang Zhao, Lichang Zhang, Bo Zhu, Lijie Wang, Haihua Yang, Biye Li, Cheng Cheng, Weiwei Lü, Rui Hu, Chenxia Li, Liu Yang, Xilin Luo, Xuejie Wu, Lunan Liu, Wenjun Cheng, Peng Cheng, Jianhao Zhang, Xiaoyu Zhang, Lei Lin, Xiaokun Wang, Yutuan Ma, Chuanhai Dong, Yanqi Sun, Yifu Chen, Yongyi Peng, Xiaojuan Liang, Shuicheng Yan, Han Fang, and Yahui Zhou. 2023. Skywork: A more open bilingual foundation model. *CoRR*, abs/2310.19341.

Junjielong Xu, Ziang Cui, Yuan Zhao, Xu Zhang, Shilin He, Pinjia He, Liqun Li, Yu Kang, Qingwei Lin, Yingnong Dang, et al. 2024. Unilog: Automatic logging via llm and in-context learning. In *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*, pages 1–12.

Zhangyue Yin, Qiushi Sun, Qipeng Guo, Jiawen Wu, Xipeng Qiu, and Xuanjing Huang. 2023. Do large language models know what they don't know? In *ACL (Findings)*, pages 8653–8665. Association for Computational Linguistics.

Matej Zecevic, Moritz Willig, Devendra Singh Dhami, and Kristian Kersting. 2023. Causal parrots: Large language models may talk causality but are not causal. *Trans. Mach. Learn. Res.*, 2023.

Zhiyuan Zeng, Jiatong Yu, Tianyu Gao, Yu Meng, Tanya Goyal, and Danqi Chen. 2024. Evaluating large language models at evaluating instruction following. In *ICLR*. OpenReview.net.

Boning Zhang, Chengxi Li, and Kai Fan. 2024a. MARIO eval: Evaluate your math LLM with your math LLM-A mathematical dataset evaluation toolkit. *CoRR*, abs/2404.13925.

Danyang Zhang, Lu Chen, Situo Zhang, Hongshen Xu, Zihan Zhao, and Kai Yu. 2024b. Large language models are semi-parametric reinforcement learning agents. *Advances in Neural Information Processing Systems*, 36.

Kaiqi Zhang, Shuai Yuan, and Honghan Zhao. 2024c. TALEC: teach your LLM to evaluate in specific domain with in-house criteria by criteria division and zero-shot plus few-shot. *CoRR*, abs/2407.10999.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *ICLR*. OpenReview.net.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Neil Zhenqiang Gong, Yue Zhang, and Xing Xie. 2023. Promptbench: Towards evaluating the robustness of large language models on adversarial prompts. *CoRR*, abs/2306.04528.

Ariel Zylberberg, Daniel M Wolpert, and Michael N Shadlen. 2018. Counterfactual reasoning underlies the learning of priors in decision making. *Neuron*, 99(5):1083–1097.

A Appendix

A.1 Notation and Definitions

A.1.1 Counterfactual Input

To formally describe the structure of counterfactual input, we define the key elements as follows: Let X denote the counterfactual input. Each example, as illustrated in the yellow boxes in Figure 1, consists of corresponding questions $Q_1, Q_2, \ldots, Q_n$ and their respective answers $A_1, A_2, \ldots, A_n$. The specific counterfactual question that the LLM must answer is denoted as Q_c, and the model's generated response to input X is represented as $\text{LLM}(X) = A_c$. Here, A_c serves as the answer to the counterfactual question, which we refer to as the model's "Claim" throughout this paper.

A.1.2 Verification Input

To formally define the structure of verification input, we introduce the following notation: Let Y denote the Verification Input. In each example E, there are corresponding questions $Q_1, Q_2, \ldots, Q_n$ and their respective answers $A_1, A_2, \ldots, A_n$. The specific verification question that the LLM must answer is represented as Q_v, and the model's output for input Y is given by $\text{LLM}(Y) = A_v$. Here, A_v serves as the answer to the verification question, which we refer to as the model's "Behavior" throughout this paper.

A.2 Detailed Results

A.2.1 Comparison of GSM8K-acc, ICL-A, and CFA

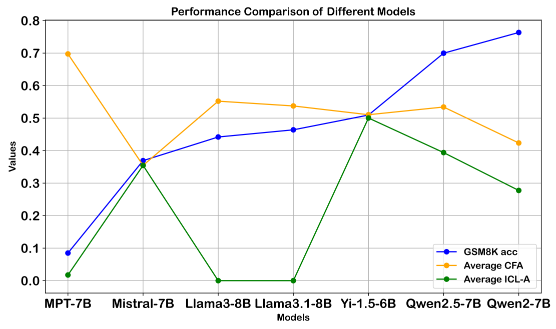


Figure 4: Performance comparison of different language models across three evaluation metrics. Average ICL-A is defined as the mean of the ICL-A obtained in the three tasks of removing the last example, removing the last and second examples, and removing the most similar example (TF-IDF). Similarly, Average CFA is the mean CFA across these same tasks.

ICL-A and CFA do not increase steadily with GSM8K-acc. The line graph shows that while some models perform well in GSM8K-acc, there is no consistent upward trend in ICL-A and CFA. For example, Qwen2-7B achieves a GSM8K-acc of 0.7635, but its ICL-A of 0.6709 is not a significant improvement. Similarly, MPT-7B, despite a low GSM8K-acc of 0.0849, performs well in CFA. Meanwhile, Llama3-8B, with a GSM8K-acc of 0.442, shows no capacity for in-context learning (ICL-A = 0.0000). These observations demonstrate that an increase in GSM8K-acc does not necessarily correlate with a consistent rise in ICL-A or CFA, suggesting that these metrics capture different aspects of the models' abilities.**ICL-A is a more challenging metric for evaluating in-context learning than GSM8K-acc.** Models struggle more with ICL-A, as shown by Llama3.1-8B, which scores 0.0000 on ICL-A despite a GSM8K-acc of 0.464. In contrast, Yi-1.5-6B shows better adaptability, with a GSM8K-acc of 0.5095 and an ICL-A of 0.5072. However, even models with high GSM8K-acc face challenges in ICL-A, especially in complex tasks like "Remove the Most Similar Example." This highlights that ICL-A is a more stringent measure of a model's adaptability and in-context learning capabilities compared to GSM8K-acc.

A.2.2 Extending BaC Evaluation to DROP and Winogrande

To further demonstrate the applicability of the BaC framework across a broader range of tasks, we extended our evaluation to two additional benchmark datasets: **DROP** (?) and **Winogrande** (?). These datasets differ substantially from GSM8K: DROP focuses on reading comprehension with discrete reasoning, while Winogrande targets commonsense reasoning in a pronoun resolution setting. Evaluating BaC on these diverse tasks allows us to assess the framework's robustness beyond mathematical problem solving.

Models	**DROP**		**Winogrande**	
	CFA	**ICL-A**	**CFA**	**ICL-A**
Qwen2-7B	0.3162	0.1232	0.4905	0.2081

Table 4: Evaluation of BaC metrics (CFA and ICL-A) on additional datasets DROP and Winogrande.

As shown in Table 4, BaC yields consistent and interpretable outcomes across both datasets. By incorporating DROP and Winogrande, we demonstrate that the framework generalizes to tasks involving different forms of reasoning, thereby strengthening the empirical validation of BaC's effectiveness.

A.2.3 Effect of Different Embedding Methods on Cosine Similarity Results

To clarify, the cosine similarity results reported in our main experiments were computed using embeddings generated by the `all-MiniLM-L6-v2` model from the Sentence-BERT library (Reimers and Gurevych, 2019).

To further examine the robustness of our findings and address potential concerns that different embedding methods may yield different results, we additionally experimented with pre-trained GloVe embeddings to compute cosine similarity. We then re-evaluated a representative subset of models on GSM8K. This comparison enables us to contrast transformer-based embeddings with classic count-based embeddings.

As shown in Table 5, while the use of GloVe embeddings results in minor variations in absolute CFA/ICL-A scores, the overall performance trends remain consistent. Therefore, we report SBERT-based results as our primary setting in the main text, and include this GloVe-based comparison here for completeness.

Models	Metric	SBERT	GloVe
Mistral-7B	CFA	0.3475	0.3461
	ICL-A	0.3374	0.3372
Qwen2-7B	CFA	0.3168	0.3152
	ICL-A	0.1221	0.1201
Yi-1.5-6B	CFA	0.5073	0.5087
	ICL-A	0.5072	0.5065

Table 5: Comparison of cosine similarity results computed using SBERT (all-MiniLM-L6-v2) versus GloVe embeddings on GSM8K.

A.2.4 Confidence Level Results

The Confidence Level (CL) reflects a model's confidence in its response, indicating how strongly it believes in its ability to provide the correct answer. The results in Table 6 compare the confidence scores of various models across three counterfactual scenarios: (1) Remove the Last Example, (3) Remove the Most Similar Example (*TF-IDF*), and (4) Remove the Last and Second Last Examples.

A.2.5 Loss Curve of the Fine-Tuning Process

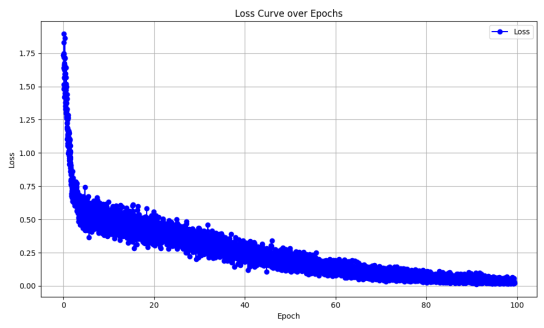


Figure 5: Loss curve for fine-tuning Llama3-8B in the data leakage setting.

Figure 5 presents the loss curve for fine-tuning Llama3-8B on the crafted training data, which includes test data leakage. As expected, the loss decreases over the course of training and stabilizes at 60 Epoch, indicating that the training process is proceeding as intended.

A.2.6 Further Empirical Validation of BaC's Effectiveness Against Data Leakage

The experiments in Section 4.3 require fine-tuning the LLMs. In contrast, the experiment presented here aims to validate BaC's effectiveness against data leakage while keeping the LLMs unchanged, without applying any fine-tuning. To achieve this, we require metrics to evaluate the presence of potential data leakage in existing LLMs, rather than precisely controlling the extent of data leakage as in Table 3. Inspired by the loss-based measurements proposed in (Wei et al., 2023), we introduce $\Delta 5$, a metric grounded in the assumption that the few-shot learning performance of a reasonable LLM should decline as the number of examples in the prompt decreases. This expectation arises from the fact that fewer examples provide less contextual information for effective in-context learning. However, in the presence of data leakage, the model may exhibit anomalous behavior, as it no longer depends on the provided context and instead relies on memorized answers. The metric is rooted in interpretable principles. Similar to (Zhang et al., 2020; Sellam et al., 2020), such metrics remain valuable and provide practical insights.

Here is the definition of $\Delta 5$. Let Accuracy$_{i\text{-shot}}$ and Accuracy$_{j\text{-shot}}$ denote the model's performance with i and j examples in the prompt. We can define the metric $\Delta 5$:

$$\Delta 5 = \sum_{1 \le i < j \le 5} \left(\frac{\text{Accuracy}_{i\text{-shot}} - \text{Accuracy}_{j\text{-shot}}}{N} \right) \tag{4}$$

where

$$N = \sum_{i<j} (j-i), \quad i, j \in \{1,2,3,4,5\}, \text{ and } i < j. \tag{5}$$

As shown in Table 8, a higher $\Delta 5$ value suggests potential data leakage, as the model maintains or even improves accuracy.

We can quantify the robustness of the metric to data leakage by examining the correlation between $\Delta 5$ and evaluation metrics. For instance, if an evaluation metric increases as $\Delta 5$ increases, it suggests that the metric is more susceptible to data leakage, and vice versa. The correlations, based on the results in Table 8, are summarized in Table 9.

The positive correlation (0.8011) between 5-shot accuracy and $\Delta 5$ suggests that models achieving higher accuracy in many-shot settings are more prone to data leakage. This is because their performance does not degrade as expected when provided with fewer prompt examples, indicating potential memorization rather than in-context learning. **The proposed ICL-A metric exhibits a negative correlation (-0.4727) with $\Delta 5$, suggesting that it degrades as the extent of data leakage increases.** This is because ICL-A evaluates the consistency between behaviors and claims across counterfactual and verification inputs, which are dynamically generated during evaluation, thereby mitigating the effects of data leakage.

Models	Remove the Last Example		Remove the Last and Second Last Examples		Remove the Most Similar Example *(T)*	
	GSM8K	TriviaQA	GSM8K	TriviaQA	GSM8K	TriviaQA
Mistral-7B	1.0000	0.6642	1.0000	0.6607	1.0000	0.6637
Llama3-8B	0.0015	0.6714	0.0000	0.6698	0.0023	0.6694
Llama3.1-8B	0.0000	0.6611	0.0000	0.6616	0.0008	0.6586
Yi-1.5-6B	0.9712	0.5205	0.9879	0.5179	0.9962	0.5243
Qwen2-7B	0.0743	0.5693	0.1842	0.5653	0.8939	0.5681
Qwen2.5-7B	0.3169	0.5538	0.6748	0.5555	0.8324	0.5560
InternLM2.5-7B	0.2145	0.5756	0.2411	0.5711	0.6141	0.5762

Table 6: Confidence Level (CL) comparison of various models across different counterfactual scenarios: (1) Remove the Last Example, (3) Remove the Most Similar Example *(TF-IDF)*, (4) Remove the Last and Second Last Examples.

Models	Remove the Last and Second Last Examples				Remove Examples Randomly			
	GSM8K		TriviaQA		GSM8K		TriviaQA	
	CFA	ICL-A	CFA	ICL-A	CFA	ICL-A	CFA	ICL-A
Mistral-7B	0.3374	0.3374	0.6606	0.6606	0.3475	0.3475	0.6509	0.6509
Llama3.1-8B	0.5504	0.0000	0.3383	0.0000	0.5605	0.0000	0.3306	0.0000
Yi-1.5-6B	0.5163	0.5072	0.5116	0.4109	0.4979	0.5067	0.4908	0.4079
Qwen2.5-7B	0.5490	0.4253	0.4444	0.0000	0.5320	0.4324	0.4308	0.2466
InternLM2.5-7B	0.4481	0.0098	0.4439	0.0941	0.4316	0.0101	0.4478	0.0893

Table 7: Performance of various LLMs across different counterfactual input generation methods: (4) Remove the Last and Second Last Examples and (5) Remove Examples Randomly.

Models	5-shot	4-shot	3-shot	2-shot	1-shot	ICL-A	$\Delta5$
Mistral-7B	0.3692	0.3791	0.3374	0.3116	0.1319	0.35	-0.10842
Llama3-8B	0.442	0.4556	0.4526	0.4041	0.3108	0.55	-0.06278
Yi-1.5-6B	0.5095	0.4989	0.5102	0.4678	0.213	0.49	-0.12482
Qwen2.5-7B	0.6998	0.6513	0.627	0.74	0.5512	0.39	-0.04170
Qwen2-7B	0.7635	0.7392	0.7521	0.7536	0.74	0.28	-0.00652

Table 8: The k-shot accuracy and $\Delta5$ on the GSM8K test set. ICL-A is the average ICL-A score for GSM8K in Table 2.

Metrics	Correlation with $\Delta5$
5-shot accuracy	0.8011
ICL-A	-0.4727

Table 9: Correlations between $\Delta5$ and the metrics

A.2.7 Additional Methods for Creating Counterfactual and Verification Inputs

We designed five distinct scenarios for constructing counterfactual inputs. In Table 2, we present scenarios (1)–(3), while the remaining scenarios (4)–(5) are shown in Table 7. The general trend remains consistent: the selected LLMs perform poorly on the CFA tasks and even worse on the ICL-A tasks, demonstrating that the proposed metrics are challenging.

A.2.8 The claim of error bounds

We conducted multiple runs of our key experiments on ChatGPT-3.5-turbo and ChatGPT-4o-mini to assess the stability of our metrics. Across repeated trials with different random seeds and prompt orderings, we observed that the variation in CFA and ICL-A scores remained within a narrow range of ±0.008.

A.3 Information About Use Of AI Assistants

We used AI models for grammar checking and in a few instances for code writing.