Benchmark Self-Evolving:
A Multi-Agent Framework for Dynamic LLM Evaluation

Siyuan Wang[1]*, **Zhuohan Long**[2]*,
Zhihao Fan[3], **Zhongyu Wei**[2], **Xuanjing Huang**[2],
[1]University of Southern California, [2]Fudan University, [3]Alibaba Inc,
sw_641@usc.edu; 24210980127@m.fudan.edu.cn

Abstract

This paper presents a benchmark self-evolving framework to dynamically evaluate rapidly advancing Large Language Models (LLMs). We utilize a multi-agent system to reframe new evolving instances with high confidence that extend existing benchmarks. Towards a more scalable, robust and fine-grained evaluation, we implement six reframing operations to construct evolving instances testing LLMs against diverse queries, shortcut biases and probing their problem-solving sub-abilities. With this framework, we extend datasets across general and specific tasks, through various iterations. Experimental results show a performance decline in most LLMs against their original results under scalable and robust evaluations, offering a more accurate reflection of model capabilities alongside our fine-grained evaluation. Besides, our framework widens performance discrepancies both between different models and within the same model across various tasks, facilitating more informed model selection for specific tasks. We hope this framework contributes the research community for continuously evolving benchmarks alongside LLM development. [1]

1 Introduction

Recent advancements in Large Language Models (LLMs) (Touvron et al., 2023; Chiang et al., 2023; OpenAI, 2023; Jiang et al., 2023) have demonstrated remarkable performance across various tasks, ranging from text generation to complex problem-solving. The evaluation of LLMs thus has emerged as a crucial area of research (Chang et al., 2023; Espejel et al., 2023). It can provide a comprehensive understanding of the capabilities and limitations in these models, and guide the selection of the most applicable LLM for specific applications. Besides, a systematic assessment of LLMs would inspire further potential improvement.

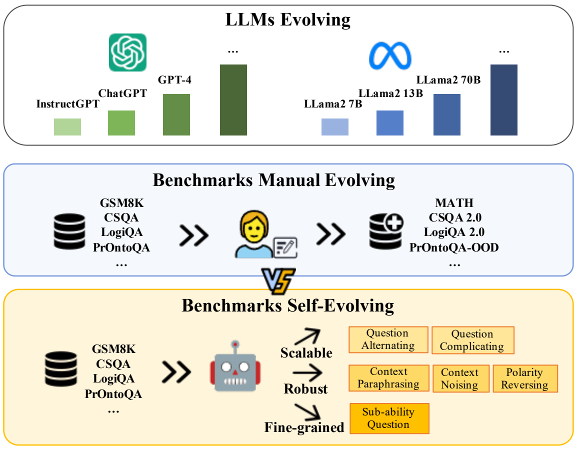


Figure 1: The evolution of LLMs necessitates benchmark self-evolving.

A multitude of benchmark datasets (Hendrycks et al., 2020; Liang et al., 2022; bench authors, 2023) have been proposed to evaluate LLMs. However, with the rapid development and emerging abilities of ever-evolving LLMs driven by increasing training data and parameters, as shown in Figure 1, these static datasets with limited reasoning difficulty and multifaceted diversity are increasingly inadequate for thorough assessment. Besides, the extensive use of data for improving LLMs leads to data contamination issues (Zhou et al., 2023; Shi et al., 2023), where in-domain training or even public test data may be inadvertently included during LLM training, resulting in biased evaluations. These challenges necessitate continual updates of static benchmark datasets, enabling more dynamic and accurate evaluations of LLMs. Since annotating new benchmarks from scratch is costly (Kiela et al., 2021), Wei et al. (2023) evaluate LLMs using perplexity on re-sampled data. However, this overreliance on perplexity may not fully reflect LLMs' performance beyond predictive accuracy. Zhu et al. (2023) dynamically synthesize test samples based on directed acyclic graphs, but this method struggles to generalize to tasks that cannot be graph-

[*] Equal contribution.

[1] Code and data are available at `https://github.com/NanshineLoong/Self-Evolving-Benchmark.git`.

Proceedings of the 31st International Conference on Computational Linguistics, pages 3310–3328
January 19–24, 2025. ©2025 Association for Computational Linguistics

represented. In this work, we propose to flexibly update existing benchmark datasets instead of constructing entirely new ones.

We introduce a benchmark self-evolving framework, which reframes existing benchmark instances into new variants for dynamic evaluation, by modifying their contexts or questions, and corresponding answers. This framework propels existing benchmarks towards self-evolution in three directions, providing a systematical dynamic evaluation of LLMs. First, to examine LLMs' ability to generalize across diverse and increasingly challenging queries, we introduce **scalable evaluation** by creating alternative or more complex questions requiring more reasoning steps based on original contexts. Second, to counteract LLMs' tendency to exploit shortcut biases or leaky instances (Gallegos et al., 2023; Yang et al., 2023) and their sensitivity to data noise (Dong et al., 2023; Pezeshkpour and Hruschka, 2023), our framework implements **robust evaluation**. This involves incorporating various perturbations to the contexts of original instances, including paraphrasing, adding noise, and reversing polarity. Finally, to mitigate the impact that outdated data and bias susceptibility could skew capability assessments, we design **fine-grained evaluation** to probe LLMs' sub-abilities for solving problems to drive further improvement.

Although LLM-driven data evolution has been explored previously (Wang et al., 2022; Xu et al., 2023), it primarily focus on generating training data with less emphasis on strictly ensuring data accuracy needed for evaluation. To address this, we design a multi-agent system to dynamically generate evolving instances towards above three directions from existing benchmarks while ensuring high accuracy. The system comprises four key components: an instance pre-filter, an instance creator, an instance verifier and a candidate option formulator. The workflow begins with the pre-filter to select manageable instances from the original evaluation set. The instance creator crafts new instances by editing their contexts or questions with answers, which the verifier checks for correctness. To further enhance reliability, the candidate option formulator subsequently creates an incorrect answer option for each new context-question pair, which the verifier need to identify as inconsistent with the new context-question. These rigorously generated and double-verified instances will be used for dynamic evaluation. All components can be powered by advanced LLMs for corresponding datasets (such as GPT-4 for most tasks and Med-Gemini for specific medical domains). This workflow can be iteratively conducted based on continuously developed LLMs for benchmark ever-evolving with increasingly challenging instances.

We dynamically extend benchmark datasets covering both general and specific tasks, including GSM8K, CLUTRR, StrategyQA, BoolQ, MedQA, HotpotQA, SPARTQA. Besides, we iteratively apply our framework twice on GSM8K to illustrate its continuous evolutionary effectiveness. Results show that our scalable and robust evaluation are more challenging compared to original benchmarks, leading to a general performance decline for all models. It helps reveal the limited generalizability and robustness of models to diverse and complex queries. This along with sub-ability probing offers a more accurate reflection of LLMs' true capabilities. Furthermore, our framework expands the performance gap between various models and also the differences of a single model across various tasks, which benefits the selection of the most suitable LLM for specific applications.

2 Benchmark Self-Evolving Framework

Our framework is illustrated in Figure 1. We first introduce different directions that we modify the contexts or questions of original instances along with their answers for newly evolving instances (see Section 2.1). We employ a multi-agent system to facilitate collaboration on evolving instance generation and double-verification. (see Section 2.2)

2.1 Evolving Instance Taxonomy

An instance can be formulated as a triplet consisting of a context (C), a question (Q) and an answer (A). For tasks involving only a question and an answer, the context is designated as null. Given an original evaluation instance (C_o, Q_o, A_o), we either perturb the context C_o or alter the questions Q_o, simultaneously forming the corresponding answer. We thereby reframe an evolving instance as (C_e, Q_o, A_e) or (C_o, Q_e, A_e), for scalable, robust and fine-grained evaluation.

Scalable Evaluation For scalable evaluation of evolving LLMs, we create various questions with corresponding new answers based on the original instance to examine whether LLMs can generalize to diverse and increasingly challenging queries. Our approach includes the creation of alternative questions (*Question Alternating*) that examine dif-

Original Instance

Context: Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for $2 per fresh duck egg.
Original Question: How much in dollars does she make every day at the farmers' market?
Original Answer: Janet sells 16 - 3 - 4 = «16-3-4=9»9 duck eggs a day. She makes 9 * 2 = $«9*2=18»18 every day at the farmer's market. #### 18

Directions	Operation Types	Changed Items	Examples
Scalable	Question Alternating	question, answer	**Alternative Question:** If Janet decides to use 2 of her daily eggs to make a special omelette for dinner each day, how much will she earn at the farmers' market in a week? **Alternative Answer:** $98
	Question Complicating	question, answer	**Complex Question:** How many days will it take for Janet to save $100 from her earnings at the farmers' market? **Complex Answer:** 6 days
Robust	Context Paraphrasing	context	**Paraphrased Context:** Janet's daily egg production from her ducks is 16. Each morning, she consumes three eggs for breakfast and uses four more to bake muffins for her friends. The remaining eggs are then sold at the farmers' market for $2 each.
	Context Noising	context	**Noised Context:** Janet's ducks lay 16 eggs per day and *her cows product 4L milk* per day. She eats three eggs and 1L milk for breakfast every morning and bakes muffins for her friends every day with four eggs. She *keeps the remainder milk for herself* and only sells the remainder eggs at the farmers' market daily for $2 per fresh duck egg.
	Polarity Reversing	context, answer	**Reversed Context:** Janet's ducks lay *20 eggs* per day. She eats *five* for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for *$2.5* per fresh duck egg. **Reversed Answer:** 27.5
Fine-grained	Sub-ability Question Generation	question, answer	**New Question:** What are the detailed reasoning steps required to calculate how much in dollars Janet makes every day at the farmers' market? **New Answer:** The solution involves 2 reasoning steps. [Step 1] calculates the number of eggs can be sold. [Step 2] calculate the money she earns.

Table 1: The reframing operations and examples for generating evolving instances.

ferent facets of the original context, as well as more complex questions requiring additional reasoning steps (*Question Complicating*). To maintain the accuracy of evolving instance, we conduct question generation without changing original contexts.

Robust Evaluation For more robust evaluation of LLMs, we introduce various perturbations to the contexts of original instances to generate evolving instances. Specifically, we apply three perturbation strategies: (1) *Context Paraphrasing*: paraphrasing the original context to obtain diverse formulations; (2) *Context Noising*: adding noise by introducing irrelevant or adversarial sentences into the original context; (3) *Polarity Reversing*: reversing the polarity or altering key details of the original context. The first two perturbations require maintaining the original answer labels while the third approach necessitates a corresponding answer change, offering a more rigorous test of the model's adaptability.

Fine-grained Evaluation We design fine-grained evaluation by generating *sub-ability questions* to probe LLMs' problem-solving capabilities. We focus on three explainability-related sub-abilities: (1) task planning capability that inquires about the details of planned reasoning steps, (2) implicit knowledge identification capability for recognizing underlying facts or rules, and (3) relevant context retrieval capability for extracting pertinent information from the given context to support its responses.

The detailed operations to reframe evolving instances and corresponding examples are in Table 1.

2.2 Multi-Agent Evolving Instance Generator

To generate evolving instances and ensure their correctness, we design a multi-agent instance creator system, incorporating four key agents: an instance pre-filter, an instance creator, an instance verifier and a candidate option formulator. All agents are built upon advanced LLMs (e.g., GPT-4) for corresponding datasets to fulfill their roles. The system's workflow is presented in Figure 2.

Instance Pre-Filter The instance pre-filter is designed to scan the original dataset to identify manageable instances that fall within the capability of our LLM backbone and can be answered correctly. This process establishes a correct foundation for subsequent operations and enhance the overall system's reliability. It takes the context and question

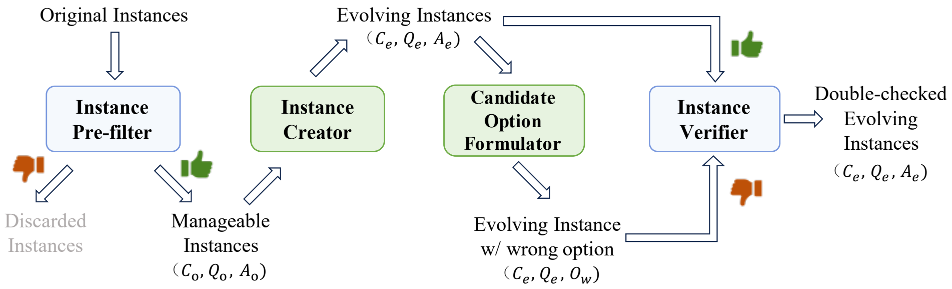


Figure 2: The workflow of our Multi-Agent Evolving Instance Generator system.

of the original instance as inputs, prompting the LLM to predict the answer and compare its prediction with the reference answer. A two-shot chain of thought (Wei et al., 2022) prompting setting is utilized to select manageable cases as (C_o, Q_o, A_o).

Instance Creator The instance creator agent is pivotal in generating different types of evolving instances (C_e, Q_e, A_e). Given an original instance including a context, question, answer, and its task description (e.g., "mathematical reasoning task"), the instance creator either modifies the contexts $(Q_e = Q_o)$ or forms new questions $(C_e = C_o)$. We design different prompts for six predefined reframing operations as Table 1. For operations without altering the answer $(A_e = A_o)$, the instance creator is instructed to maintain the original answer during the operation. For operations yielding new answers, the instance creator infers the new answer step-by-step after reformulating the context or question. This process adopts a one-shot prompting strategy to better understand operation requirements.

Instance Verifier The primary function of the instance verifier agent is to validate the correctness of newly evolving instance (C_e, Q_e, A_e), ensuring the answer correctly support the corresponding context and question. Since these evolving instances are auto-generated by an LLM-based agent, the inclusion of this verifier is essential to control our data quality. The instance verifier directly takes the context, question and answer of the new instance as inputs, and employs a two-shot CoT prompting strategy. It utilizes both a correct and an incorrect demonstrations to avoid potential biases.

Candidate Option Formulator The candidate option formulator aims to generate an incorrect answer option O_w for each new context-question pair (C_e, Q_e). It has two primary purposes: (1) mitigating the impact of LLM's shortcut biases on data reliability by enabling the instance verifier to double-check both the validity of the previous-generated instance (C_e, Q_e, A_e) and the inability of the candidate option O_w to answer the context-question pair (C_e, Q_e); (2) providing a standardized binary-choice assessment method for more accurate evaluation metrics. For new instances with fine-grained questions where their free-form answers are not easy to evaluate, we adopt this binary-choice evaluation. Specifically, the formulator takes the context-question pair and the correct answer as inputs, and adopts a one-shot prompting strategy to output a wrong candidate option.

System Workflow Our system involves the following steps. First, the instance pre-filter selects manageable instances (C_o, Q_o, A_o) from the original evaluation set. From these, the instance creator creates new instances (C_e, Q_e, A_e), and the candidate option formulator subsequently generates an incorrect option O_w for each new context-question pair. Then the instance verifier checks the correctness of both the new instance (C_e, Q_e, A_e) and its incorrect alternative (C_e, Q_e, O_w). Only instances that pass the double-check process, i.e., the generated instance is examined as correct and the alternative is incorrect, will be used for dynamic evaluation. This process can be iteratively conducted based on continuously developed LLMs for benchmark ever-evolving. Detailed algorithm and specific prompts for all agents are provided in Appendix A.

3 Experiments

3.1 Setup

Tasks and Datasets Using our benchmark self-evolving framework, we dynamically extend benchmark datasets of five tasks covering both general and specific domains: mathematical reasoning (GSM8K (Cobbe et al., 2021)), logical reasoning (CLUTRR (Sinha et al., 2019)), commonsense reasoning (StrategyQA (Geva et al., 2021)), reading comprehension (BoolQ (Clark et al., 2019)), and medical license exams (MedQA (Jin et al., 2021)).

Datasets	Models	Scalable Evaluation	Robust Evaluation	Overall
		(Evolving←Original)	(Evolving←Original)	(Evolving←Original)
GSM8K	GPT-4	85.00 ← 100.0 (-15.00)	97.10 ← 100.0 (- 2.90)	93.07 ← 100.0 (**- 6.93**)
	ChatGPT	60.83 ← 93.33 (-32.50)	79.25 ← 91.29 (-12.04)	73.13 ← 91.97 (**-18.84**)
	ChatGLM	42.50 ← 66.67 (-24.17)	62.66 ← 67.22 (- 4.56)	55.96 ← 67.04 (**-11.08**)
	LLama	40.83 ← 60.00 (-19.17)	60.58 ← 58.51 (+ 2.07)	54.02 ← 59.00 (**- 4.98**)
	Mistral	27.50 ← 41.67 (-14.17)	35.27 ← 39.42 (- 4.15)	32.69 ← 40.17 (**- 7.48**)
CLUTRR	GPT-4	77.11 ← 100.0 (-22.89)	93.42 ← 100.0 (- 6.58)	86.55 ← 100.0 (**-13.45**)
	ChatGPT	65.66 ← 83.13 (-17.47)	78.51 ← 82.02 (- 3.51)	73.10 ← 82.49 (**-9.39**)
	ChatGLM	55.42 ← 73.49 (-18.07)	67.11 ← 74.56 (- 7.45)	62.18 ← 74.11 (**-11.93**)
	LLama	47.59 ← 36.14 (+11.45)	36.40 ← 33.77 (+ 2.63)	41.12 ← 34.77 (+ 5.35)
	Mistral	45.78 ← 55.42 (- 9.64)	50.00 ← 53.95 (- 3.95)	48.22 ← 54.57 (**- 6.35**)
StrategyQA	GPT-4	98.25 ← 100.0 (- 1.75)	/	98.25 ← 100.0 (**- 1.75**)
	ChatGPT	64.91 ← 91.23 (-26.32)	/	64.91 ← 91.23 (**-26.32**)
	ChatGLM	66.67 ← 73.68 (- 7.01)	/	66.67 ← 73.68 (**- 7.01**)
	LLama	78.95 ← 75.44 (+ 3.51)	/	78.95 ← 75.44 (+ 3.51)
	Mistral	77.19 ← 73.68 (+ 3.51)	/	77.19 ← 73.68 (+ 3.51)
BoolQ	GPT-4	99.36 ← 100.0 (- 0.64)	97.35 ← 100.0 (- 2.65)	98.17← 100.0 (**- 1.83**)
	ChatGPT	92.31 ← 91.03 (+ 1.28)	91.15 ← 90.27 (+ 0.88)	91.62 ← 90.58 (+ 1.04)
	ChatGLM	86.54 ← 89.10 (- 2.56)	90.71 ← 88.05 (+ 2.66)	89.01 ← 88.48 (+ 0.53)
	LLama	84.62 ← 92.31 (- 7.69)	91.60 ← 91.60 (- 0.00)	88.74 ← 91.88 (**- 3.14**)
	Mistral	76.92 ← 80.13 (- 3.21)	83.19 ← 79.20 (+ 3.99)	80.63 ← 79.58 (+ 1.05)
MedQA	GPT-4	74.07 ← 100.0 (- 25.93)	90.20 ← 100.0 (- 9.80)	83.06← 100.0 (**- 16.94**)
	ChatGPT	64.81 ← 75.31 (-10.49)	65.20 ← 76.96 (-11.76)	65.03 ← 76.23 (**-11.20**)
	ChatGLM	51.23 ← 54.94 (- 3.70)	49.02 ← 55.88 (+ 6.86)	50.00 ← 55.46 (**-5.46**)
	LLama	32.72 ← 42.59 (- 9.88)	34.31 ← 43.63 (- 9.31)	33.61 ← 43.17 (**- 9.56**)
	Mistral	53.09 ← 46.91 (- 6.17)	51.47 ← 48.04 (+ 3.43)	52.19 ← 47.54 (+4.64)

Table 2: Comparison of evolving and original evaluations. Left of the arrow are evolving results; right shows original performance on respective instances. Values in parentheses are performance changes.

The most advanced LLM for the first four datasets is GPT-4, while for the last is Med-Gemini (Saab et al., 2024) followed by GPT-4 (as of May 2024). We build all agents in our system upon GPT-4 since Med-Gemini is not publicly available.

We randomly select 100 instances from publicly available dev/test sets of each dataset[2], and input them into our multi-agent system to generate new evaluation instances of various reframing types. For GSM8K, CLUTRR, BoolQ and MedQA, we generate new instances across all six types as in Table 1. For StrategyQA without context, we generate instances with complex and fine-grained questions. Specifically for sub-abilities, we focus on task planning ability for GSM8K and BoolQ, both task planning and implicit knowledge identification for StrategyQA, and all three sub-abilities for CLUTRR and MedQA. Detailed descriptions and statistics of generated datasets using various operations are summarized in Appendix B.1.

Examined LLMs We evaluate both closed-source models, ChatGPT and ChatGLM (Zeng et al., 2023), and open-source models, LLama (Touvron et al., 2023) and Mistral (Jiang et al., 2023), using our evolving evaluation datasets. We compare their performance against on original datasets to demonstrate the effectiveness of our framework. For closed-source models, we use gpt-3.5-turbo-1106 and chatglm-turbo versions, while for open-source models, we employ LLama2-70B-Chat and Mistral-7B-Instruct-v0.2. We also evaluate GPT-4 (gpt-4-1106-preview) despite its involvement in generating evolving instances, to test whether our framework can also provide more scalable and robust evaluation for its LLM backbone. More implementation details are described in Appendix B.2.

To show the broad applicability of our framework, we further experiment with two open-source LLMs, Qwen2-7B-Instruct (Yang et al., 2024) and Yi-34B-Chat (Young et al., 2024) and two datasets: HotpotQA (Yang et al., 2018) for reading comprehension and SPARTQA (Mirzaee et al., 2021) for spatial reasoning, as show in Appendix B.9.

3.2 Overall Comparison

We first provide an overall assessment of LLMs with scalable and robust evaluations, leaving fine-grained evaluation in Section 3.4. Scalable evaluation involves instances with alternative and complex questions, while robust evaluation using instances with paraphrased, noised and reversed contexts, and compare their performance against on corresponding original instances. For fair comparison, the average performance on original instances is reported for each evaluation type. Table 2 presents the main comparisons, with arrows indi-

[2] CLUTRR are sampled from clauses of length $k \leq 3$.

cating shifts from original to evolving evaluation results. We have the following findings.

(1) Overall, most models exhibit reduced performance in our scalable and robust evaluation compared to original results, especially in scalable evaluation. This offers more accurate measures of LLMs' abilities, highlighting that original results may overestimate their proficiency.

(2) Although these evolving instances are generated by GPT-4, GPT-4's performance still declines on them. Because they are generated given original instances with correct answers, which aids model's reasoning. Evaluating GPT-4 with evolving instances aims to highlight the dynamic and challenging nature of these instances, and uncover GPT-4's limitations in logical, medical, and mathematical reasoning.

(3) Our scalable evaluation effectively expands the performance gap between models. Initially, GPT-4 and ChatGPT exhibit less than a 10% accuracy difference on GSM8K and StrategyQA, while this gap increasing to 20-30% under our scalable evaluation. On the BoolQ where all models consistently perform well, our scalable evaluation further highlights their disparities.

(4) Our framework widens performance discrepancies of the same model across tasks. For example, while ChatGPT consistently achieves 80∼90% accuracy on five tasks, its proficiency diverges under our evolved evaluation, remaining stable only on BoolQ. Similarly, GPT-4 maintains effectiveness on most datasets while showing declines on CLUTRR and MedQA.

3.3 Analysis of Varied Reframing Operations

To further assess the impact of various reframing operations on model evaluation, we gather results of each operation across all datasets and compare them with corresponding original results. Our analysis as detailed in Figure 3 shows that among five reframing operations, *question complicating* causes the most disruption to models, followed by *polarity reversing* and *question alternating*. In contrast, *context paraphrasing* and *context noising* have a limited impact on model performance. These findings suggest that our framework primarily enhances the original benchmarks by highlighting LLMs' limitations regarding question generalizability and susceptibility to adversarial attacks.

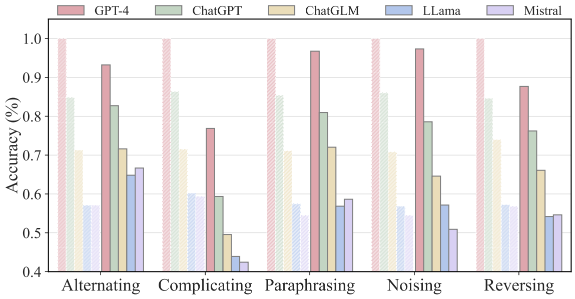


Figure 3: Comparison of evolving results using various reframing operations versus original results. Darker bars show accuracy for each operation across all datasets, with lighter bars ahead representing original accuracy.

We provide a perplexity-based analysis indicating that our generated instances exhibit greater complexity and diversity than the original instances for dynamic evaluation, along with an error analysis. Please refer to Appendix B.6 for details.

3.4 Further Analysis on Sub-Ability

The fine-grained evaluation paves a way to dissect models' sub-abilities. We aggregate the results of each sub-ability across all datasets and compare the models' rankings with their original ranking.

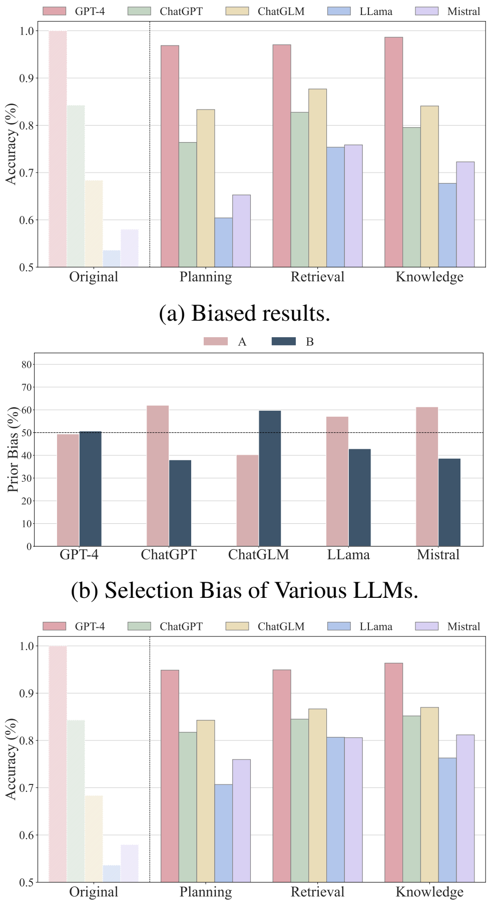
Substantial Discrepancy Between Original and Fine-grained Evaluations. Figure 4a shows that ChatGLM, initially behind ChatGPT in the original evaluation, surprisingly outperforms ChatGPT in all sub-ability evaluations. Scrutinizing ChatGPT's results reveals a significant selection bias towards option 'A', suggesting such bias impairs LLM's decision-making and leads to poorer performance.

Presence of Selection Bias in Certain LLMs. Following (Zheng et al., 2023a), We estimate the prior prediction distribution of different LLMs on options ID 'A' and 'B'. The result in Figure 4b shows that ChatGPT, LLama, and Mistral significantly prefer 'A', unlike the neutral stance of GPT-4 and ChatGLM. For a fair model evaluation, we utilize a bias calibrating method to obtain debiased results as shown in Figure 4c, with the bias mitigation method detailed in Appendix B.5.

Improvement Potential for Planning Ability As Figure 4c, GPT-4 consistently performs best across all three sub-abilities while Mistral showing the lowest performance. Among three sub-abilities, planning emerges as the weakest skill for all LLMs, highlighting a key area for further enhancements.

3.5 Quality of Evolving Instances

Human Verification To demonstrate the reliability of our dynamic evaluation, we sample a subset of our generated instances and conduct a human



(a) Biased results.

(b) Selection Bias of Various LLMs.

(c) Debiased results.

Figure 4: Results of fine-grained sub-ability evaluation.

annotation (manually verified by the authors) to assess their quality. Specifically, we randomly select five instances that are incorrectly answered by ChatGPT for each reframing operation across all datasets, with a total of 155 instances. Following human verification, 147 out of 155 instances (94.8%) are deemed accurate, reinforcing the credibility of our evolving instances.

Instance Filter Rate Our system incorporates a pre-filter and a double-verification process to enhance the reliability of generated instances. The pre-filter discards nearly 9% of original instances that exceed GPT-4's capabilities. Subsequently, the double-verification stage filters out approximately 22% instances initially processed correctly by GPT-4, underscoring the importance of this strategy for instance quality. Detailed statistics are in Appendix B.3. As our system driven by advanced backbone LLMs might introduce a slight favorable bias towards themselves, we have detailedly investigated this potential in Appendix B.4. Our analysis demonstrates that the bias is minimal and does not diminish the backbone LLMs' absolute superiority.

3.6 Impact on Data Contamination

To analyze our framework's ability to mitigate data contamination, we design controlled experiments to simulate data contamination. We construct two instruction-tuning datasets: one simulating in-domain contamination by including parts of our evaluation benchmark's training set, and the other simulating direct contamination by incorporating both training and evaluation sets. We respectively fine-tune LLama-2-7B-Chat on these two datasets, with training details in Appendix B.7. We assess the original model and two fine-tuned models using both original and generated evolving instances, with results shown in Figure 5.

- Compared to the original model, both in-domain and direct contaminated models show notable improvement under original evaluation, revealing how data contamination can skew results. In contrast, in our dynamic evaluation, the performance gap narrows, especially in scalable and fine-grained evaluations, indicating our framework's resilience to data contamination.
- In fine-grained evaluation, the in-domain contaminated model outperforms the original, indicating that in-domain training enhances task-related abilities. Yet, the direct contaminated model underperforms, suggesting that memorizing original answers may hinder solving new problems, highlighting the value of fine-grained evaluation in mitigating data contamination.

3.7 Multi-Iterations for Ever-Evolving

Our framework has the potential to continuously evolve benchmarks, by iteratively using more advanced LLMs to generate increasingly challenging instances based on data from preceding rounds. To illustrate this, we conduct a second iteration of benchmark evolution on the previously evolved GSM8K dataset, taking the current most advanced LLM, GPT-4o, as the agent backbone. Table 3 compares model performance after one and two iterations of evolution. Results show a significant performance decline in all models except GPT-4, particularly in scalable evaluation, indicating that further evolution can increase its challenge level.

4 Related Work

LLM-driven Data Evolution Leveraging LLMs to generate increasingly challenging data has been explored to enhance training. Xu et al. (2023) introduce Evol-Instruct, which refines data by con-

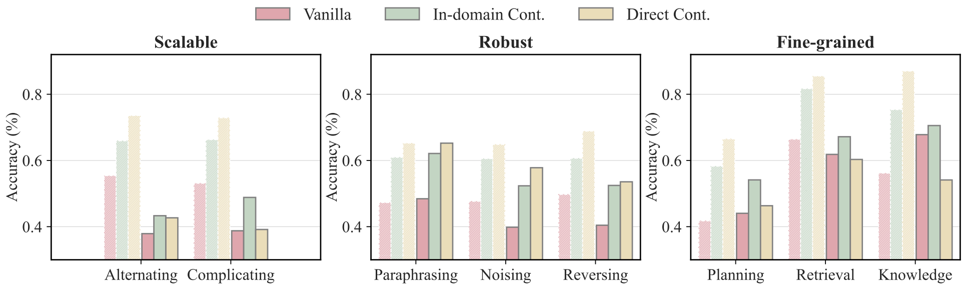


Figure 5: Comparison of LLama-2 models under different contamination conditions: “Vanilla” (original model), “In-domain Cont.” (in-domain contaminated) and “Direct Cont.” (direct contaminated).

Models	Scalable Evaluation (2 iterations←1 iteration)	Robust Evaluation (2 iterations←1 iteration)	Overall (2 iterations←1 iteration)
GPT-4	89.29 ← 85.00 (+4.29)	99.13 ← 97.10 (+2.03)	96.52 ← 93.07 (+3.45)
ChatGPT	50.00 ← 60.83 (-10.83)	70.69 ← 79.25 (-8.56)	65.19 ← 73.13 (**-8.94**)
ChatGLM	27.38 ← 42.50 (-15.12)	65.09 ← 62.66 (+2.43)	55.06 ← 55.96 (**-0.90**)
LLama	29.76 ← 40.83 (-11.07)	46.55 ← 60.58 (-14.03)	42.09 ← 54.02 (**-11.93**)
Mistral	21.43 ← 27.50 (-6.07)	26.29 ← 35.27 (-8.98)	25.00 ← 32.69 (**-7.69**)

Table 3: Comparison between GSM8K datasets evolved through two iterations and one iteration.

trolling difficulty and diversity. Lee et al. (2023) present a unified data creation pipeline across diverse tasks, including those with complex and semantically sparse label spaces. However, these training data synthesis strategies do not prioritize sample accuracy for evaluation. For this, Zhu et al. (2023) propose to use graph structures for dynamic test synthesis. Fan et al. (2023) propose NPHardEval, which updates evaluation samples for NP-hard problems. However, these methods are task-specific and may not generalize well.

LLMs Evaluation LLMs’ advanced performance (OpenAI, 2023; Jiang et al., 2023) across tasks has sparked interest in their evaluation, which includes automatic (Liang et al., 2022), human (Zheng et al., 2023b), and LLM-based evaluation (Liu et al., 2023). Automatic evaluation is cost-effective for extensive assessments, requiring diverse task-specific (Yu et al., 2023; Wang et al., 2024) and general (Hendrycks et al., 2020) benchmarks. However, evolving LLMs and potential data contamination make static benchmarks insufficient. Dynamic evaluations (Wei et al., 2023; Zhu et al., 2023) address this but struggle on dataset difficulty, diversity and ability evaluation. A concurrent work (Zhu et al., 2024) also uses LLM to generate samples from existing benchmarks, but emphasizes robust evaluation while neglecting more complex questions requiring more reasoning steps and fine-grained ability assessments. Besides, its generate-then-verify two-agent system struggles with data accuracy and continuous benchmark evolution.

Data Contamination The expansion of LLMs training datasets poses a data contamination challenge, leading to in-domain overlaps with existing or public development and test sets and risking biased evaluation (Sainz et al., 2023). This undermines benchmark fairness and accuracy (Zhou et al., 2023), casting doubt on whether high performance reflects true generalization or mere data memorization (Biderman et al., 2023). Shi et al. (2023) and Golchin and Surdeanu (2023) propose detecting and removing contaminated data from benchmark. Wei et al. (2023) utilize perplexity for evaluation on newly sampled data without extra annotations, yet this may not fully reflect models’ capabilities. Our benchmark self-evolving framework can mitigate bias from data contamination.

5 Conclusion

Our study introduce a benchmark self-evolving framework that iteratively employs a multi-agent system to enhance existing benchmarks for more scalable, robust and fine-grained LLM evaluations. Results show a general decline in LLM performance and significant discrepancies across various models and tasks, highlighting that our framework can provide more accurate and comprehensive evaluations. We hope our sustainable framework contributes the research community to continuously evolves benchmarks alongside LLM development, helping select the most capable LLMs for specific applications, and evaluate LLMs’ drawbacks to guide their further improvement.

Limitations

Limitation on benchmark coverage Due to computational limit, our dynamic evaluation study only explores seven datasets across various textual tasks and select 100 instances from each dataset to construct nearly 1600 evolving instances. Our framework can flexibly generalize to other tasks and even different modalities for a broader analysis.

Limitation on examining LLMs We evaluate three closed-source (GPT-4, ChatGPT, ChatGLM) and four open-source LLMs (LLama, Mistral, Qwen, Yi) using our crafted evolving instances to illustrate our scalable, robust, and fine-grained evaluation. We acknowledge the limitation in the scope of LLMs, and will later provide further experiments on more LLMs.

Limitation on instance accuracy Despite incorporating a pre-filtering and double-verification procedure, our system, which is entirely powered by GPT-4, may inevitably generate a small number of instances with inaccuracies, as evidenced by human verification. This might result in less accurate assessments of LLMs.

Risks

Introduction of Factual Errors For benchmark datasets containing factual information, such as BoolQ, our framework may generate counterfactual information to alter the key details of the original context during the polarity reversing operation. Such inaccuracies, if inadvertently used as learning material by the models, could negatively impact their performance and reliability.

Environmental Impact A significant risk associated with our methodology is the potential increase in environmental impact due to the extensive use of OpenAI's APIs for large language models. This is particularly concerning for benchmarks of substantial size, as the energy consumption and carbon footprint associated with generating evolving instances could be considerable.

Ethics Statement

All data utilized in our benchmark self-evolving framework are sourced from publicly available datasets. Our generated evolving instances for dynamic evaluation are also publicly released for usage and have been subjected to a thorough review by the authors. This setting guarantees transparency and reproducibility in our experiments, allowing other researchers to evaluate and expand upon our work. Our benchmark-evolving framework is strictly limited to be used for instance generation that follow the ethical guidelines of the community. The authors emphatically denounce the use of our framework for generating inaccurate or harmful instances.

Acknowledgements

This research is supported by National Key R&D Program of China (2023YFF1204800) and National Natural Science Foundation of China (No.62176058). The project's computational resources are supported by CFFF platform of Fudan University.

References

BIG bench authors. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Kaijie Zhu, Hao Chen, Linyi Yang, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2023. A survey on evaluation of large language models. *arXiv preprint arXiv:2307.03109*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *NAACL*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Guanting Dong, Jinxu Zhao, Tingfeng Hui, Daichi Guo, Wenlong Wan, Boqi Feng, Yueyan Qiu, Zhuoma Gongque, Keqing He, Zechen Wang, and Weiran

Xu. 2023. Revisit input perturbation problems for llms: A unified robustness evaluation framework for noisy slot filling task. *Preprint*, arXiv:2310.06504.

Jessica López Espejel, El Hassane Ettifouri, Mahaman Sanoussi Yahaya Alassan, El Mehdi Chouham, and Walid Dahhane. 2023. Gpt-3.5, gpt-4, or bard? evaluating llms reasoning ability in zero-shot setting and performance boosting through prompts. *Natural Language Processing Journal*, 5:100032.

Lizhou Fan, Wenyue Hua, Lingyao Li, Haoyang Ling, Yongfeng Zhang, and Libby Hemphill. 2023. Nphardeval: Dynamic benchmark on reasoning ability of large language models via complexity classes. *arXiv preprint arXiv:2312.14890*.

Isabel O Gallegos, Ryan A Rossi, Joe Barrow, Md Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K Ahmed. 2023. Bias and fairness in large language models: A survey. *arXiv preprint arXiv:2309.00770*.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Shahriar Golchin and Mihai Surdeanu. 2023. Time travel in llms: Tracing data contamination in large language models. *arXiv preprint arXiv:2308.08493*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, et al. 2021. Dynabench: Rethinking benchmarking in nlp. *arXiv preprint arXiv:2104.14337*.

Dong-Ho Lee, Jay Pujara, Mohit Sewak, Ryen W White, and Sujay Kumar Jauhar. 2023. Making large language models better data creators. *arXiv preprint arXiv:2310.20111*.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. 2022. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. Gpteval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*.

Roshanak Mirzaee, Hossein Rajaby Faghihi, Qiang Ning, and Parisa Kordjmashidi. 2021. Spartqa:: A textual question answering benchmark for spatial reasoning. *arXiv preprint arXiv:2104.05832*.

OpenAI. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Pouya Pezeshkpour and Estevam Hruschka. 2023. Large language models sensitivity to the order of options in multiple-choice questions. *Preprint*, arXiv:2308.11483.

Khaled Saab, Tao Tu, Wei-Hung Weng, Ryutaro Tanno, David Stutz, Ellery Wulczyn, Fan Zhang, Tim Strother, Chunjong Park, Elahe Vedadi, et al. 2024. Capabilities of gemini models in medicine. *arXiv preprint arXiv:2404.18416*.

Oscar Sainz, Jon Ander Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. 2023. Nlp evaluation in trouble: On the need to measure llm data contamination for each benchmark. *arXiv preprint arXiv:2310.18018*.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. 2023. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789*.

Koustuv Sinha, Shagun Sodhani, Jin Dong, Joelle Pineau, and William L Hamilton. 2019. Clutrr: A diagnostic benchmark for inductive reasoning from text. *arXiv preprint arXiv:1908.06177*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Siyuan Wang, Zhongyu Wei, Yejin Choi, and Xiang Ren. 2024. Can llms reason with rules? logic scaffolding for stress-testing and improving llms. *arXiv preprint arXiv:2402.11442*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

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. *Preprint*, arXiv:2310.19341.

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

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Bing Yin, and Xia Hu. 2023. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *arXiv preprint arXiv:2304.13712*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, et al. 2024. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Zhiyuan Liu, Peng Zhang, Yuxiao Dong, and Jie Tang. 2023. GLM-130b: An open bilingual pre-trained model. In *The Eleventh International Conference on Learning Representations (ICLR)*.

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. 2023a. Large language models are not robust multiple choice selectors. *arXiv e-prints*, pages arXiv–2309.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023b. Judging llm-as-a-judge with mt-bench and chatbot arena. *Preprint*, arXiv:2306.05685.

Kun Zhou, Yutao Zhu, Zhipeng Chen, Wentong Chen, Wayne Xin Zhao, Xu Chen, Yankai Lin, Ji-Rong Wen, and Jiawei Han. 2023. Don't make your llm an evaluation benchmark cheater. *arXiv preprint arXiv:2311.01964*.

Kaijie Zhu, Jiaao Chen, Jindong Wang, Neil Zhenqiang Gong, Diyi Yang, and Xing Xie. 2023. Dyval: Graph-informed dynamic evaluation of large language models. *Preprint*, arXiv:2309.17167.

Kaijie Zhu, Jindong Wang, Qinlin Zhao, Ruochen Xu, and Xing Xie. 2024. Dyval 2: Dynamic evaluation of large language models by meta probing agents. *arXiv preprint arXiv:2402.14865*.

A Details of Framework

A.1 Algorithm Design

The pseudo-code for the algorithm of our multi-agent evolving instance setting system is presented in Algorithm 1.

A.2 Prompts of Multi-Agent Evolving Instance Setter

Table 4, 5, 6, 7, 8, 9, 10 and 11 present prompts for Instance Generator of different reframing operations. Table 12 presents the prompt for Instance Verifier. Table 13 presents the prompt for Candidate Option Formulator.

B Experimental Analysis

B.1 Dataset Descriptions and Statistics

- GSM8K: a collection of grade school math problems in a free-form QA format, featuring diverse arithmetic and algebraic problems.
- CLUTRR: a synthesized free-form question answering dataset designed for evaluating logical reasoning over kinship relationships.
- StrategyQA: consists of crowdsourced yes/no questions that require implicit reasoning steps and commonsense strategies. The instances in StrategyQA consist solely of questions and answers, with their contexts being null.
- BoolQ: a reading comprehension dataset sourced from Google's Natural Questions, offers yes/no questions based on real Google searches paired with answers from Wikipedia articles.
- MedQA: consists of multiple-choice questions collected from professional medical board exams. It is designed to test deep understanding and application of medical knowledge.

The statistics of our generated datasets using various operations are summarized in Table 14.

Algorithm 1 Multi-Agent Evolving Instance Setter

```
Require: An original evaluation instance $(C_o, Q_o, A_o)$, its task description $s$.
Ensure: An evolving instance $(C_e, Q_e, A_e)$.
$(C_o, Q_o, A_o) \leftarrow$ Instance Pre-filter($(C_o, Q_o, A_o)$, $s$)
$(C_e, Q_e, A_e) \leftarrow$ Instance Generator($(C_o, Q_o, A_o)$, $s$)
$O_w \leftarrow$ Candidate Option Formulator($(C_e, Q_e, A_e)$, $s$)
if Instance Verifier($C_e, Q_e, A_e$) and not Instance Verifier($C_e, Q_e, O_w$) then
    return $(C_e, Q_e, A_e)$
else
    return $NULL$
end if
```

Prompt for Instance Generator on Question Alternating

You are an expert Question Creator. You will receive an instance of {task description}, including a context, a question and its answer.
You are tasked with creating an alternative question to explore a different aspect of the original problem.
Please do not change the context but just edit the question and the answer.
Please first generate the question. Then think step-by-step in one line to give an brief analysis of the question, Finally, directly present a short answer omitting the intermediate steps, in a single line.

Context: {context C_o}
Original Question: {question Q_o}
Original Answer: {answer A_o}
Alternative Question:

Table 4: Prompt on Question Alternating.

Prompt on Question Complicating

You are an expert Question Creator. You will receive an instance of {task description}, including a context, an original question and its answer.
Your task is to generate a more complex question and its corresponding answer based on the given context, with the goal of incorporating additional reasoning steps beyond what is required by the original question and answer. Please do not change the context but just edit the question and the answer.
Please first generate the question. Then think step-by-step in one line to give an brief analysis of the question, Finally, directly present a short answer omitting the intermediate steps, in a single line.
Context: {context C_o}
Original Question: {question Q_o}
Original Answer: {answer A_o}
Alternative Question:

Table 5: Prompt on Question Complicating.

Prompt for Instance Generator on Context Paraphrasing

You are an expert Question Creator.You will receive an instance of {task description}, including a context, a question and its answer.
Your task is to rephrase the given context in a short and easy-readable manner without summarizing or explaining.
Confirm that the rephrased context do not change the answer to the original question.
Simply output the rephrased context and do not output the original question.

Context: {context C_o}
Original Question: {question Q_o}
Original Answer: {answer A_o}
Alternative Context:

Table 6: Prompt on Context Paraphrasing.

Prompt for Instance Generator on Context Noising

You are an expert Question Creator. You will receive an instance of {task description}, including a context, a question and its answer.
You are tasked with creating a new context by inserting irrelevant facts within the critical sentences of the original context. Make sure these facts shouldn't change the correct answer to the question.
Simply output the rephrased context and do not output the original question.

Context: {context C_o}
Original Question: {question Q_o}
Original Answer: {answer A_o}
Alternative Context:

Table 7: Prompt on Context Noising.

Prompt for Instance Generator on Polarity Reversing

You are an expert Question Creator. You will receive an instance of {task description}, including a context, a question and its answer.
Your task is to generate a new context by altering key details in the original context. Ensure that the rest of the original context remains unchanged. The altered details should change the answer to the question.
Please first output the rephrased context. Then give an one-line step-by-step analysis of the original question based on the new context. Finally, generate the corresponding direct answer in a newline.

Context: {context C_o}
Original Question: {question Q_o}
Original Answer: {answer A_o}
Alternative Context:

Table 8: Prompt on Polarity Reversing.

Prompt for Instance Generator on Planning

You are an expert Task Planner. You will receive an instance of {task description}, including a context, a question and its answer.
Your task is to generate a new question and its corresponding answer, aiming to ask about the plan to solve the original question given the context. Your new question can either inquire about all reasoning steps required or ask for the specific details about a certain (e.g., first, second, or last) step.
Please first generate the question. Then think step-by-step in one line to give an brief analysis of the question, Finally, directly present a short answer omitting the intermediate steps, in a single line.

Context: {context C_o}
Original Question: {question Q_o}
Original Answer: {answer A_o}
Alternative Question:

Table 9: Prompt on Planning.

Prompt for Instance Generator on Retrieval

You are an expert Relevant Context Retriever. You will receive an instance of {task description}, including a context, a question and its answer.
Your task is to generate a new question and its corresponding answer, aiming to identify the relevant information from the given context necessary to solve the original question with the original answer. Your answer must be exclusively from the given context, to contain all required information to solve the original question and cover the original answer.
Please first generate the question. Then think step-by-step in one line to give an brief analysis of the question, Finally, directly present a short answer omitting the intermediate steps, in a single line.

Context: {context C_o}
Original Question: {question Q_o}
Original Answer: {answer A_o}
Alternative Question:

Table 10: Prompt on Retrieval.

Prompt for Instance Generator on Knowledge

You are an expert Relevant Context Retriever. You will receive an instance of {task description}, including a context, a question and its answer.
Your task is to generate a new question and its corresponding answer, aiming to ask about the implicit knowledge (e.g., facts, rules, commonsense, ...) required to solve the original question. Your new answer should directly list all required implicit knowledge for the question.
Please first generate the question. Then think step-by-step in one line to give an brief analysis of the question, Finally, directly present a short answer omitting the intermediate steps, in a single line.

Context: {context C_o}
Original Question: {question Q_o}
Original Answer: {answer A_o}
Alternative Question:

Table 11: Prompt on Knowledge.

Prompt for Instance Verifier

You are an expert Question-Answer Validator. You will receive an instance of {task description}, including a context, a question and its answer.
Your task is to validate whether the answer is correct to solve the question given the context.
Please think step-by-step in one line to analyze whether the answer is correct for the question and the context. Then give your final judgement with Yes or No in a newline.

Context: {context C}
Question: {question Q}
Answer: {answer A}
Judgement:

Table 12: Prompt for Instance Verifier.

Prompt for Candidate Option Formulator

You are an expert Candidate Option Generator. You will receive an instance of {task description}, including a context, a question and its answer.
Your task is to modify the provided answer to generate a candidate option that wrongly answer the question given the context.

Context: {context C}
Question: {question Q}
Answer: {answer A}
Option:

Table 13: Prompt for Candidate Option Formulator.

Dataset	Manageable	Scalable		Robust			Fine-Grained			Total
		Alternating	Complicating	Paraphrasing	Noising	Reversing	Planning	Knowledge	Retrieval	
GSM8K	96/100	65	55	90	90	61	71	/	/	432
CLUTRR	96/100	88	78	76	80	72	69	81	64	608
StrategyQA	83/100	/	57	/	/	/	78	65	/	200
BoolQ	90/100	88	68	90	86	50	/	/	67	382
MedQA	88/100	83	79	80	80	44	70	74	72	582

Table 14: Statistics of our evolving instances from five original datasets.

Task Formats We adopt two task formats for different evaluation directions. For scalable and robust evaluations, we follow the original datasets' task types, and employ a two-shot CoT prompting strategy. For fine-grained evaluation, we create binary-choice questions featuring two options, A and B, where one is the correct answer and the other, generated by our Candidate Option Formulator, is incorrect. We implement a zero-shot prompting approach for fine-grained evaluation.

Dataset	Scalable		Robust			Fine-grained			Average
	Alternating	Complicating	Paraphrasing	Noising	Reversing	Planning	Knowledge	Retrieval	
GSM8K	32.29%	42.71%	6.25%	6.25%	36.46%	26.04%	/	/	25.00%
CLUTRR	8.33%	18.75%	20.83%	16.67%	25.00%	28.13%	15.63%	33.33%	20.83%
StrategyQA	/	31.33%	/	/	/	6.02%	21.69%	/	19.68%
BoolQ	2.22%	24.44%	0.00%	4.44%	44.44%	/	/	25.56%	29.26%
MedQA	5.68%	10.23%	9.09%	9.09%	50.00%	20.45%	15.91%	18.18%	17.33%
Average	12.13%	25.38%	9.04%	9.11%	38.98%	20.16%	17.74%	25.69%	22.42%

Table 15: Percentage (%) of instances filtered by double-verification.

B.2 Implementation Details

We used GPT-4 and GPT-4o for instance creation, setting the temperature to 0.8, while keeping all other parameters at their default values. Regarding the context length, GPT-4o supports a maximum context length of 128K tokens, and we used approximately 300-500 tokens for input and 250 max_tokens for output. In terms of sampling parameters, we set the top_p parameter to its default value of 1 and did not specify a seed value.

B.3 Instance Filtering Statistics

Table 15 shows the percentage of instances filtered by double-verification.

B.4 Analysis of Favorable Bias Towards Backbone LLMs

LLMs-powered framework may cause a slight favorable bias towards the backbone LLMs, specifically GPT-4 in this case. This bias may arise from two sources:

(1) Instances generated and verified by GPT-4 might be easier for GPT-4 to answer. For this point, as shown in Table 2, GPT-4's performance significantly declines on evolving data compared to the original data, sometimes more than other LLMs (e.g., on CLUTRR, BoolQ, and MedQA). **This suggests almost no bias from this source.**

(2) Some instances pre-filtered by GPT-4 (where it gave incorrect answers) might be correctly answered by other models, potentially underestimating their performance. To investigate this, we test other models on these GPT-4 discarded instances from GSM8K, CLUTRR, StrategyQA, BoolQ and MedQA datasets, as shown in Table 16. With only a small portion of questions cannot be answered by GPT-4, we find that most of them are also incorrectly answered by other models, indicating that **the favorable bias from GPT-4's Pre-Filter is minimal.**

To mitigate the minimal bias introduced by pre-filtering, we use a Bayesian approach to estimate the performance of GPT-4 and another model C (e.g., LLaMa and ChatGLM) on all data (both pre-filtered and manageable instances). Let M and N respectively represent the number of manageable and pre-filtered instances by GPT-4, with K denoting the number of pre-filtered instances correctly answered by model C. The performance of GPT-4 and model C on original data and evolving data are R^o_{gpt4}, R^o_C, R^e_{gpt4} and R^e_C (as reported in Table 2).

The performance on all data before pre-filtering can be estimated as $R^o_{gpt4} * \frac{M}{M+N}$ and $R^o_C * \frac{M+K}{M+N}$. After benchmark evolving, the lower bound of GPT-4's performance (assuming GPT-4 cannot answer any of the evolved variants of pre-filtered data) and the upper bound of model C's performance (assuming model C can answer all evolved variants of the correctly answered but pre-filtered data) can be estimated as $R^e_{gpt4} * \frac{M}{M+N}$ and $R^e_C * \frac{M+K}{M+N}$. Here we ignore the impact of instances filtered out by the verifier, which has minimal effect as previously noted. The calibrated results for all models are listed in the Table 17. After bias calibration, GPT-4 still shows significant improvement.

Synthesizing data using LLMs for training and evaluating models is a common practice, typically relying on the best-performing model available that outperforms others. In this case, comparing other models to GPT-4 aims to demonstrate their upper bound and highlight the challenging and dynamic nature of our generated samples. This minimal bias does not diminish GPT-4's absolute superiority.

B.5 Selection Bias Analysis

Selection bias denotes the tendency that a model inherently assigns a higher probability to specific ID tokens, such as *A* or *B*, in multi-choice questions. The impact of *selection bias* varies across different

	GSM8k	CLUTRR	StrategyQA	BoolQ	MedQA	Overall
ChatGPT	3/4 (75%)	4/4 (100%)	11/17 (65%)	8/10 (80%)	8/12(67%)	34/47 (72%)
ChatGLM	3/4 (75%)	4/4 (100%)	9/17 (53%)	5/10 (50%)	7/12(58%)	28/47 (60%)
LLama	3/4 (75%)	3/4 (75%)	13/17 (77%)	5/10 (50%)	8/12(67%)	32/47 (68%)
Mistral	4/4 (100%)	4/4 (100%)	15/17 (88%)	7/10 (70%)	9/12(75%)	39/47 (83%)

Table 16: Error Rates for Other Models on Discarded Questions.

Models	Scalable Evaluation (Evolving←Original)	Robust Evaluation (Evolving←Original)	Overall (Evolving←Original)
		GSM8K	
GPT-4	85.00 [81.60] ← 100.0 [96.00] (-15.00 [-14.40])	97.10 [93.21] ← 100.0 [96.00] (- 2.90 [-2.79])	93.07 [89.35] ← 100.0 [96.00] (**- 6.93 [-6.65]**)
ChatGPT	60.83 [59.01] ← 93.33 [90.53] (-32.50 [-31.52])	79.25 [76.88] ← 91.29 [88.55] (-12.04 [-11.67])	73.13 [70.94] ← 91.97 [89.21] (**-18.84 [-18.27]**)
ChatGLM	42.50 [41.23] ← 66.67 [64.67] (-24.17 [-23.44])	62.66 [60.78] ← 67.22 [65.20] (- 4.56 [-4.42])	55.96 [54.28] ← 67.04 [65.02] (**-11.08 [-10.74]**)
LLama	40.83 [39.61] ← 60.00 [58.20] (-19.17 [-18.59])	60.58 [58.76] ← 58.51 [56.75] (+ 2.07 [+1.01])	54.02 [52.40] ← 59.00 [57.23] (**- 4.98 [-4.83]**)
Mistral	27.50 [26.40] ← 41.67 [40.00] (-14.17 [-13.60])	35.27 [33.86] ← 39.42 [37.84] (- 4.15 [-3.98])	32.69 [31.38] ← 40.17 [38.56] (**- 7.48 [-7.18]**)
		CLUTRR	
GPT-4	77.11 [74.02] ← 100.0 [96.00] (-22.89 [-21.98])	93.42 [89.68] ← 100.0 [96.00] (- 6.58 [-6.32])	86.55 [83.09] ← 100.0 [96.00] (**-13.45 [-12.91]**)
ChatGPT	65.66 [63.04] ← 83.13 [79.81] (-17.47 [-16.77])	78.51 [75.37] ← 82.02 [78.74] (- 3.51 [-3.37])	73.10 [70.17] ← 82.49 [79.19] (**-9.39 [-9.02]**)
ChatGLM	55.42 [53.20] ← 73.49 [70.55] (-18.07 [-17.35])	67.11 [64.42] ← 74.56 [71.58] (- 7.45 [-7.16])	62.18 [59.70] ← 74.11 [71.15] (**-11.93 [-11.45]**)
LLama	47.59 [46.16] ← 36.14 [35.06] (+11.45 [+11.10])	36.40 [35.31] ← 33.77 [32.76] (+ 2.63 [+2.55])	41.12 [39.88] ← 34.77 [33.73] (+ 5.35 [+6.15])
Mistral	45.78 [43.95] ← 55.42 [53.20] (- 9.64 [-9.25])	50.00 [48.00] ← 53.95 [51.79] (- 3.95 [-3.79])	48.22 [46.29] ← 54.57 [52.39] (**- 6.35 [-6.10]**)
		StrategyQA	
GPT-4	98.25 [81.54] ← 100.0 [83.00] (- 1.75 [-1.46])	/	98.25 [81.54] ← 100.0 [83.00] (**- 1.75 [-1.46]**)
ChatGPT	64.91 [57.77] ← 91.23 [81.19] (-26.32 [-23.42])	/	64.91 [57.77] ← 91.23 [81.19] (**-26.32 [-23.42]**)
ChatGLM	66.67 [60.67] ← 73.68 [67.05] (- 7.01 [-6.38])	/	66.67 [60.67] ↔ 73.68 [67.05] (**- 7.01 [-6.38]**)
LLama	78.95 [68.68] ← 75.44 [65.63] (+ 3.51 [+3.05])	/	78.95 [68.68] ← 75.44 [65.63] (+ 3.51 [+3.05])
Mistral	77.19 [65.61] ← 73.68 [62.63] (+ 3.51 [+2.98])	/	77.19 [65.61] ← 73.68 [62.63] (+ 3.51 [+3.00])
		BoolQ	
GPT-4	99.36 [89.42] ← 100.0 [90.00] (- 0.64 [-0.58])	97.35 [87.61] ← 100.0 [90.00] (- 2.65 [-2.39])	98.17 [88.35] ← 100.0 [90.00] (**- 1.83 [-1.65]**)
ChatGPT	92.31 [84.92] ← 91.03 [83.74] (+ 1.28 [+1.18])	91.15 [83.86] ← 90.27 [83.04] (+ 0.88 [+0.82])	91.62 [84.29] ← 90.58 [83.33] (+ 1.04 [+0.96])
ChatGLM	86.54 [82.21] ← 89.10 [84.65] (- 2.56 [-2.44])	90.71 [86.17] ← 88.05 [83.65] (+ 2.66 [+2.52])	89.01 [84.56] ← 88.48 [84.06] (+ 0.53 [+0.50])
LLama	84.62 [80.38] ← 92.31 [87.69] (- 7.69 [-7.31])	91.60 [87.01] ← 91.60 [87.01] (- 0.00 [+0.00])	88.74 [84.31] ← 91.88 [87.29] (**- 3.14 [-2.98]**)
Mistral	76.92 [71.54] ← 80.13 [74.52] (- 3.21 [-2.98])	83.19 [77.36] ← 79.20 [73.66] (+ 3.99 [+3.70])	80.63 [74.98] ← 79.58 [74.01] (+ 1.05 [+0.97])
		MedQA	
GPT-4	74.07 [65.19] ← 100.0 [88.00] (- 25.93 [-22.81])	90.20 [79.37] ← 100.0 [88.00] (- 9.80 [-8.63])	83.06 [73.09] ← 100.0 [88.00] (**- 16.94 [-14.91]**)
ChatGPT	64.81 [59.63] ← 75.31 [69.28] (-10.49 [-9.65])	65.20 [59.98] ← 76.96 [70.80] (-11.76 [-10.82])	65.03 [59.83] ← 76.23 [70.13] (**-11.20 [-10.30]**)
ChatGLM	51.23 [47.65] ← 54.94 [51.09] (- 3.70 [-3.44])	49.02 [45.59] ← 55.88 [51.97] (+ 6.86 [-6.38])	50.00 [46.50] ← 55.46 [51.58] (**-5.46 [-5.08]**)
LLama	32.72 [30.10] ← 42.59 [39.19] (- 9.88 [-9.09])	34.31 [31.57] ← 43.63 [40.14] (- 9.31 [-8.57])	33.61 [30.92] ← 43.17 [39.72] (**- 9.56 [-8.80]**)
Mistral	53.09 [48.31] ← 46.91 [42.69] (+ 6.17 [+5.62])	51.47 [46.84] ← 48.04 [43.72] (+ 3.43 [+3.12])	52.19 [47.49] ← 47.54 [43.26] (+4.64 [+4.23])

Table 17: Comparison of evolving and original evaluations. Left of the arrow are evolving results; right shows original performance on respective instances. Values in parentheses are performance changes. Values in blue display the corresponding calibrated results.

models and tasks, influencing the models' performance by reducing their robustness in handling multi-choice problems (Zheng et al., 2023a). The permutation-based debiasing method, which averages the model's prediction distributions across various option permutations, theoretically eliminates selection bias. However, due to limited access to the prediction distributions of closed-source models, we employ a sampling approximation approach to estimate and mitigate selection bias.

To formalize our discussion, we use C to denote the concatenation of context and question. Suppose an multi-choice problem consists of n options, id_i denotes the ith option ID in the default order, and o_i denotes the ith option content in the default order, $i \in \{1, 2, ..., n\}$. The option IDs remain in their default order in all multi-choice questions, while the option contents can be rearranged through different permutations to match with different option IDs. We introduce I as a permutation of the set $\{1, 2, ..., n\}$, and $\mathcal{I}$ represents the set of all possible permutations I. We use x^I to denote the concatenation of the default-ordered option IDs with the option contents permuted according to I.

We assume that given C and x^I, the observed probability of the model selecting id_i is $P_{\text{biased}}\left(id_i \mid C, x^I\right)$, which can be decomposed as:

$$Z_{x^I}^{-1} P_{\text{prior}}\left(id_i \mid C\right) P_{\text{debiased}}\left(o_{f_I(i)} \mid C, x^I\right) \quad (1)$$

where $Z_{x^I}^{-1}$ is the normalization factor, $P_{\text{prior}}\left(id_i \mid C\right)$ represents the prior probability of selecting id_i, $P_{\text{debiased}}\left(o_{f_I(i)} \mid C, x^I\right)$ represents the debiased prediction probability of $o_{f_I(i)}$, and $f_I(i)$ denotes the ith element in permutation I.

We assume P_{debiased} is invariant to the ordering of options. By applying logarithms to both P_{biased} and its decomposed expression, and then summing across all I in $\mathcal{I}$ on both sides of the equation, we can simplify and compute $P_{\text{prior}}\left(id_i \mid C\right)$ as:

$$\operatorname{softmax}\left(\frac{1}{|\mathcal{I}|}\sum_{I\in\mathcal{I}}\log P_{\text{biased}}\left(id_i \mid C, x^I\right)\right) \quad (2)$$

Based on the estimated $P_{\text{prior}}\left(id_i \mid C\right)$ and Equation 1, we can compute the normalized $P_{\text{debiased}}\left(o_{f_I(i)} \mid C, x\right)$.

For our analysis, the $P_{\text{prior}}\left(id_i \mid C\right)$ and $P_{\text{debiased}}\left(o_{f_I(i)} \mid C, x\right)$ are computed independently for different datasets. Based on the correct option's ID, we categorize all multi-choice questions in a dataset into different permutation sets to support x^I. We calculate the frequency of the model selecting id_i at different permutation sets to estimate $P_{\text{biased}}\left(id_i \mid C, x^I\right)$.

B.6 Dataset Perplexity Analysis

Perplexity is a metric that quantifies the complexity and the predictability of a dataset. By analyzing the perplexity of dataset, we can gain insights into the relative difficulty models may encounter during testing, as well as the diversity of information within the dataset. In this analysis, we calculate the perplexity of newly evolving datasets derived by reframing the original GSM8K dataset and compared them with their original counterparts. The comparison results are presented in Figure 6.

Our findings indicate that the datasets created through different reframing operations exhibit an increase in perplexity compared to the original instances. This indicates that the reframed instances are more complex and less predictable. These results, aligning with the experimental observations discussed in Section 3.3, suggest that our framework has the ability to generate instances with enhanced linguistic structure and diversity compared to the original instances.

B.7 Data Contamination Experiment Details

We construct an instruction tuning dataset comprising 4,000 general instances from alpaca-gpt4-data (Peng et al., 2023) and additional 4,000 instances, with 1,000 each from the training sets of GSM8K, CLUTRR, StrategyQA and BoolQ. This dataset is used to fine-tune a model to simulate in-domain contamination. Furthermore, we incorporate 400 instances from the original benchmark into the instruction tuning dataset to fine-tune another model to simulate the direct contamination.

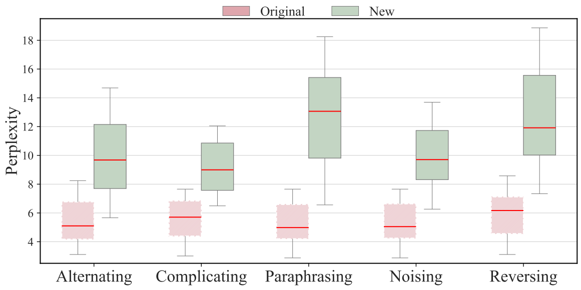


Figure 6: Perplexity comparison between original and reframed datasets.

B.8 Error Analysis

Tables 18, 19and 20 present cases from GSM8K where ChatGPT successfully responds to the original instance but fails to answer the evolved version. The case in Table 18 introduces an additional reasoning step to the original instance, which causes ChatGPT to misinterpret the context and perform an incorrect calculation. In Table 19, the evolved instance alters two data points from the original case, leading ChatGPT to neglect the critical information "4 times a week," resulting in an inaccurate response. The example in Table 20 demonstrates that, although ChatGPT correctly calculates the total number of hours the candle will burn in the original instance, it fails to address a direct question about the first reasoning step in the new instance.

By utilizing instances from our evolving framework, we can expose model's limitations more effectively, including its vulnerability to perturbations, diminished generalization capabilities on more complex problems, and inadequacies in addressing questions targeting specific sub-abilities.

B.9 Main Results on More LLMs and Datasets

- HotpotQA: a question answering dataset featuring natural, multi-hop questions, with strong supervision for supporting facts to enable more explainable question answering systems.
- SPARTQA: a textual question answering benchmark for spatial reasoning on natural language text which contains more realistic spatial phenomena not covered by prior datasets and that is challenging for state-of-the-art language models.

Original Instance	**New Instance**
Context: A robe takes 2 bolts of blue fiber and half that much white fiber.	**Context:** A robe takes 2 bolts of blue fiber and half that much white fiber.
Question: How many bolts in total does it take?	**Question:** If a merchant has 20 bolts of blue fiber and 10 bolts of white fiber, how many robes can he make using all the fibers without any leftovers?
Answer: 3	**Answer:** 10
Response: It takes 2 + 1 = 3 bolts in total.	**Response:** The merchant can **make 20/2 = 10 robes using the blue fiber**. He can **make 10/2 = 5 robes using the white fiber**. So, he can make a total of 5 robes using all the fibers without any leftovers.
Judgement: ✓	**Judgement:** ✗

Table 18: Instance Before and After Question Complicating

Original Instance	**New Instance**
Context: Jim spends 2 hours watching TV and then decides to go to bed and reads for half as long. He does this 3 times a week.	**Context:** Jim spends 3 hours watching TV and then decides to go to bed and reads for a third as long. He does this 4 times a week.
Question: How many hours does he spend on TV and reading in 4 weeks?	**Question:** How many hours does he spend on TV and reading in 4 weeks?
Answer: 36	**Answer:** 64
Response: Jim spends 2 + (2/2) = 3 hours on TV and reading each time. So, in 4 weeks, he spends 3 * 3 * 4 = 36 hours on TV and reading. The answer is 36.	**Response:** Jim spends 3 + (3/3) = 4 hours on TV and reading each time. In 4 weeks, he spends **4 * 4 = 16 hours** on TV and reading. The answer is 16.
Judgement: ✓	**Judgement:** ✗

Table 19: Instance Before and After Polarity Reversing

Original Instance	**New Instance**
Context: A candle melts by 2 centimeters every hour that it burns.	**Context:** A candle melts by 2 centimeters every hour that it burns.
Question: How many centimeters shorter will a candle be after burning from 1:00 PM to 5:00 PM?	**Question:** What is the first reasoning step to calculate how many centimeters shorter a candle will be after burning from 1:00 PM to 5:00 PM? A. The first reasoning step is to measure the initial height of the candle. B. The first reasoning step is to determine the total number of hours the candle will burn.
Answer: 8	**Answer:** B
Response: The candle burns for 5-1=4 hours. So, it will be 2*4=8 centimeters shorter. The answer is 8.	**Response:** A
Judgement: ✓	**Judgement:** ✗

Table 20: Instance Before and After Planning Operation

We provide additional experimental results with two open-source LLMs, Qwen2-7B-Instruct and Yi-34B-Chat and two datasets, HotpotQA and SPARTQA in Table 21. The results highlight the broad applicability of our framework.

Models	Scalable Evaluation (Evolving←Original)	Robust Evaluation (Evolving←Original)	Overall (Evolving←Original)
GSM8K			
GPT-4	85.00 ← 100.0 (-15.00)	97.10 ← 100.0 (- 2.90)	93.07 ← 100.0 (**- 6.93)**
ChatGPT	60.83 ← 93.33 (-32.50)	79.25 ← 91.29 (-12.04)	73.13 ← 91.97 (**-18.84)**
ChatGLM	42.50 ← 66.67 (-24.17)	62.66 ← 67.22 (- 4.56)	55.96 ← 67.04 (**-11.08)**
LLama	40.83 ← 60.00 (-19.17)	60.58 ← 58.51 (+ 2.07)	54.02 ← 59.00 (**- 4.98)**
Mistral	27.50 ← 41.67 (-14.17)	35.27 ← 39.42 (- 4.15)	32.69 ← 40.17 (**- 7.48)**
Qwen	63.33 ← 87.50 (-24.17)	80.70 ← 89.63 (-8.93)	74.93 ← 88.92 (**-13.99)**
Yi	45.83 ← 67.50 (-21.67)	63.87 ← 69.71 (-5.84)	57.88 ← 68.98 (**-11.10)**
CLUTRR			
GPT-4	77.11 ← 100.0 (-22.89)	93.42 ← 100.0 (- 6.58)	86.55 ← 100.0 (**-13.45)**
ChatGPT	65.66 ← 83.13 (-17.47)	78.51 ← 82.02 (- 3.51)	73.10 ← 82.49 (**-9.39)**
ChatGLM	55.42 ← 73.49 (-18.07)	67.11 ← 74.56 (- 7.45)	62.18 ← 74.11 (**-11.93)**
LLama	47.59 ← 36.14 (+11.45)	36.40 ← 33.77 (+ 2.63)	41.12 ← 34.77 (+ 5.35)
Mistral	45.78 ← 55.42 (- 9.64)	50.00 ← 53.95 (- 3.95)	48.22 ← 54.57 (**- 6.35)**
Qwen	57.83 ← 65.66 (-7.83)	58.33 ← 66.23 (-7.90)	58.12 ← 65.99 (**-7.87)**
Yi	55.42 ← 72.29 (-16.87)	60.53 ← 71.93 (-11.40)	58.38 ← 72.08 (**-13.70)**
StrategyQA			
GPT-4	98.25 ← 100.0 (- 1.75)	/	98.25 ← 100.0 (**- 1.75)**
ChatGPT	64.91 ← 91.23 (-26.32)	/	64.91 ← 91.23 (**-26.32)**
ChatGLM	66.67 ← 73.68 (- 7.01)	/	66.67 ← 73.68 (**- 7.01)**
LLama	78.95 ← 75.44 (+ 3.51)	/	78.95 ← 75.44 (+ 3.51)
Mistral	77.19 ← 73.68 (+ 3.51)	/	77.19 ← 73.68 (+ 3.51)
Qwen	84.21 ← 82.46 (+1.75)	/	84.21 ← 82.46 (+1.75)
Yi	77.19 ← 78.95 (-1.75)	/	77.19 ← 78.95 (**-1.75)**
BoolQ			
GPT-4	99.36 ← 100.0 (- 0.64)	97.35 ← 100.0 (- 2.65)	98.17← 100.0 (**- 1.83)**
ChatGPT	92.31 ← 91.03 (+ 1.28)	91.15 ← 90.27 (+ 0.88)	91.62 ← 90.58 (+ 1.04)
ChatGLM	86.54 ← 89.10 (- 2.56)	90.71 ← 88.05 (+ 2.66)	89.01 ← 88.48 (+ 0.53)
LLama	84.62 ← 92.31 (- 7.69)	91.60 ← 91.60 (- 0.00)	88.74 ← 91.88 (**- 3.14)**
Mistral	76.92 ← 80.13 (- 3.21)	83.19 ← 79.20 (+ 3.99)	80.63 ← 79.58 (+ 1.05)
Qwen	89.74 ← 91.67 (-1.93)	95.13 ← 90.27 (+4.86)	92.93 ← 90.84 (+2.09)
Yi	72.44 ← 76.28 (-3.84)	72.57 ← 76.55 (-3.98)	72.51 ← 76.44 (**-3.93)**
MedQA			
GPT-4	74.07 ← 100.0 (- 25.93)	90.20 ← 100.0 (- 9.80)	83.06← 100.0 (**- 16.94)**
ChatGPT	64.81 ← 75.31 (-10.49)	65.20 ← 76.96 (-11.76)	65.03 ← 76.23 (**-11.20)**
ChatGLM	51.23 ← 54.94 (- 3.70)	49.02 ← 55.88 (+ 6.86)	50.00 ← 55.46 (**-5.46)**
LLama	32.72 ← 42.59 (- 9.88)	34.31 ← 43.63 (- 9.31)	33.61 ← 43.17 (**- 9.56)**
Mistral	53.09 ← 46.91 (- 6.17)	51.47 ← 48.04 (+ 3.43)	52.19 ← 47.54 (+4.64)
Qwen	59.26 ← 55.56 (+3.70)	54.90 ← 55.39 (-0.49)	56.83 ← 55.46 (+1.37)
Yi	59.26 ← 54.94 (+4.32)	55.88 ← 56.37 (-0.49)	57.38 ← 55.74 (+1.64)
HotpotQA			
GPT-4	84.62 ← 100.00 (-15.38)	89.16 ← 100.00 (-10.84)	87.10 ← 100.00 (**-12.90)**
ChatGPT	92.90 ← 89.94 (+ 2.96)	83.25 ← 89.66 (-6.40)	87.63 ← 89.78 (**-2.15)**
ChatGLM	91.72 ← 92.90 (-1.18)	84.24 ← 93.60 (-9.36)	87.63 ← 93.28 (**-5.65)**
LLama	89.94 ← 92.31 (-2.37)	84.24 ← 91.63 (-7.39)	86.83 ← 91.94 (**-5.11)**
Mistral	76.33 ← 83.43 (-7.10)	76.85 ← 83.25 (-6.40)	76.61 ← 83.33 (**-6.72)**
Qwen	82.84 ← 85.80 (-2.96)	81.77 ← 84.24 (-2.46)	82.26 ← 84.95 (**-2.69)**
Yi	72.19 ← 84.62 (-12.43)	72.91 ← 83.74 (-10.84)	72.58 ← 84.14 (**-11.56)**
SpartQA			
GPT-4	72.22 ← 100.00 (-27.78)	74.75 ← 100.00 (-25.25)	73.83 ← 100.00 (**-26.17)**
ChatGPT	53.84 ← 58.81 (-14.97)	54.03 ← 58.98 (-14.97)	53.96 ← 58.92 (**-14.96)**
ChatGLM	41.76 ← 62.84 (-21.07)	47.21 ← 57.12 (-9.91)	45.23 ← 59.19 (**-13.96)**
LLama	42.34 ← 53.88 (-11.54)	40.43 ← 52.03 (-11.60)	41.13 ← 52.70 (**-11.58)**
Mistral	31.54 ← 34.48 (-2.94)	31.79 ← 35.08 (-3.29)	31.70 ← 34.86 (**-3.17)**
Qwen	52.76 ← 67.31 (-14.55)	47.72 ← 69.83 (-22.11)	49.55 ← 68.92 (**-19.37)**
Yi	45.76 ← 50.90 (-5.14)	46.22 ← 52.88 (-6.66)	46.05 ← 52.16 (**-6.11)**

Table 21: Comparison of evolving and original evaluations. Left of the arrow are evolving results; right shows original performance on respective instances. Values in parentheses are performance changes.