Dependency Matters: Enhancing LLM Reasoning with Explicit Knowledge Grounding

Xiangyu Wen[1], Min Li[3], Junhua Huang[2], Jianyuan Zhong[1], Zhijian Xu[1], Zeju Li[1], Yongxiang Huang[4], Mingxuan Yuan[2], Qiang Xu[1][*]

[1]The Chinese University of Hong Kong [2]Huawei Noah's Ark Lab [3]Southeast University
[4]Huawei Hong Kong Research Center

Abstract

Large language models (LLMs) often produce reasoning steps that are superficially coherent yet internally inconsistent, leading to unreliable outputs. Since such failures typically arise from implicit or poorly-grounded knowledge, we introduce *Grounded Reasoning in Dependency (GRiD)*, a novel dependency-aware reasoning framework that explicitly grounds reasoning steps in structured knowledge. GRiD represents reasoning as a graph consisting of interconnected knowledge extraction nodes and reasoning nodes, enforcing logical consistency through explicit dependencies. Each reasoning step is validated via a lightweight, step-wise verifier that ensures logical correctness relative to its premises. Extensive experiments across diverse reasoning benchmarks—including StrategyQA, CommonsenseQA, GPQA, and TruthfulQA—demonstrate that GRiD substantially improves reasoning accuracy, consistency, and faithfulness compared to recent state-of-the-art structured reasoning methods. Notably, GRiD enhances performance even when applied purely as a lightweight verification module at inference time, underscoring its generalizability and practical utility[†].

1 Introduction

Large language models (LLMs) have achieved remarkable advances in diverse applications, including conversational agents [6], machine translation [50], and automated code generation [11]. Despite these successes, LLMs continue to face significant limitations in reliably performing tasks that demand rigorous reasoning, logical consistency, and faithful knowledge utilization [7]. Such reasoning deficiencies critically undermine their trustworthiness in real-world scenarios.

To address these reasoning deficits, several structured reasoning paradigms have been developed. The seminal Chain-of-Thought (CoT) approach [39] explicitly prompts models to generate sequential reasoning steps leading toward a final conclusion. Subsequent frameworks expanded this concept with more structured reasoning topologies, including branching in Tree-of-Thought (ToT) [46], independent reasoning paths in Forest-of-Thought (FoT) [4], iterative self-correcting interactions with LLMs in iterative Chain-of-Thought (Iter-CoT) [32], and interconnected steps in Graph-of-Thought (GoT) [3]. Collectively referred to as *X-of-Thought (XoT)* methods, these structured reasoning techniques aim to improve the coherence and reliability of reasoning chains. Additionally, incorporating in-context learning (ICL), where related reasoning examples are included in prompts, further enhances reasoning capabilities [10, 45, 22, 48].

However, despite these promising developments, significant challenges remain. XoT approaches often suffer from inherent fragility arising from long reasoning traces, making them susceptible to reasoning

[*]Corresponding author. Mail to: qxu@cse.cuhk.edu.hk

[†]Code is available at: https://github.com/cure-lab/GRiD.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

inconsistencies and hallucinations [43, 49]. Specifically, two primary forms of inconsistency persist: (1) logical inconsistencies between intermediate reasoning steps [2, 42, 16], and (2) misalignments between a model's implicit knowledge and its explicit reasoning process [2, 18, 25].

To mitigate inconsistencies within reasoning processes, verification-based approaches have been widely explored [35]. For instance, self-consistency methods [37] sample multiple reasoning sequences and utilize majority voting to enhance reliability, though they primarily target final answer correctness without explicitly verifying intermediate reasoning steps. By contrast, methods such as Best-of-N [31] explicitly evaluate and select high-quality reasoning steps throughout the process. More recently, Process Reward Models (PRMs) [35, 21, 23] have highlighted the importance of step-wise verification, assigning explicit scores to intermediate reasoning steps. However, PRMs are often domain-specific—typically optimized for mathematical reasoning—and rely heavily on domain-tailored verifiers, thus limiting their generalizability [15]. Alternative techniques, such as Monte Carlo Tree Search (MCTS) [33], attempt to balance exploration and exploitation in verifying reasoning paths, but their computational complexity renders them impractical for real-time deployment scenarios.

Furthermore, existing research has paid insufficient attention to the critical misalignment between a model's implicit knowledge and its explicit reasoning processes. Some recent approaches incorporate external knowledge into the model's reasoning context [27, 18, 25]. Nevertheless, these strategies predominantly rely on prompting mechanisms and external databases or larger auxiliary models, introducing significant limitations in scalability, independence, and adaptability.

To overcome these fundamental limitations, we introduce **Grounding Reasoning in Dependency (GRiD)**, a reasoning framework explicitly designed to ground reasoning steps directly in the LLM's own internal knowledge base. The core insight behind GRiD is that LLMs naturally acquire substantial factual knowledge during pre-training [5], yet during inference, their statistical generation process often leads to hallucinations or inaccurate extraction of knowledge premises, negatively impacting reasoning accuracy. To mitigate this, GRiD introduces a structured reasoning graph comprising two distinct node types: *knowledge extraction nodes* and *reasoning nodes*. Knowledge extraction nodes explicitly prompt the model to retrieve factual knowledge pertinent to subsequent reasoning steps, while reasoning nodes perform logical inferences based strictly on these grounded premises, thereby ensuring concise and dependency-aware reasoning.

Leveraging this structured dependency graph, GRiD incorporates a *lightweight verification module* that explicitly checks each reasoning step's correctness against its parent nodes. This verification mechanism ensures logical consistency and tightly aligns the model's explicit reasoning outputs with its implicit internal knowledge. A key advantage of GRiD is that both its reasoning graph construction and verification processes rely exclusively on the base model itself, eliminating dependency on external databases or larger auxiliary LLMs, and thus greatly enhancing scalability and adaptability. Overall, grounding each reasoning step in intrinsic knowledge ensures accurate and reliable extraction, improves interpretability, anchors reasoning in known facts, and enables consistency checks, keeping traces coherent and answers reliable.

We extensively evaluate GRiD on diverse question-answering benchmarks where precise reasoning is essential. Empirical results demonstrate that GRiD significantly improves the reasoning accuracy of widely-used LLMs, including LLaMA and Qwen, achieving approximately 7% accuracy improvement over established state-of-the-art structured reasoning methods and 25% over the original base models. Furthermore, GRiD matches the performance of advanced models such as GPT-4o and DeepSeek-R1, underscoring its effectiveness, scalability, and potential as a generalizable reasoning framework.

2 Related Work and Motivational Examples

2.1 Inconsistency Issues in X-of-Thought

In addition to standard XoTs, some studies reveal that CoT-like reasoning can emerge intrinsically through alternative decoding strategies (e.g., top-k sampling) without explicit prompting [38]. These methods collectively advance LLMs' systematic reasoning by combining structured exploration.

Despite the success of XoTs on reasoning tasks, these methods rely on the assumption that generated reasoning steps faithfully reflect valid logical progressions—an assumption increasingly challenged by empirical studies [41, 26, 30, 43, 8]. Key issues with XoTs include factual inaccuracies, hallucinated conditions, post-hoc rationalizations, and silent error corrections, which propagate through reasoning

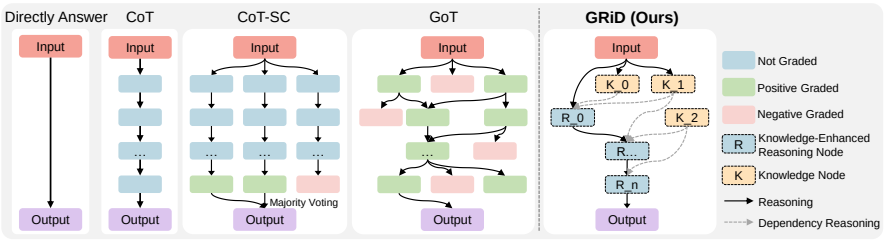

Figure 1: This figure compares the reasoning structures of our GRiD method with previous CoT-based approaches. Our GRiD method introduces two distinct types of nodes: *Knowledge Nodes* and *Knowledge-Enhanced Reasoning Nodes*. The *Reasoning Nodes* perform dependency reasoning by acquiring premise knowledge from the *Knowledge Nodes*.

chains and reduce accuracy by up to 40.4% in noisy environments [43, 49]. For instance, models frequently overlook critical problem constraints or introduce unsupported premises, as observed in tasks requiring arithmetic or demographic reasoning [43, 40]. Furthermore, implicit rationalizations and unfaithful shortcuts often yield surface-level coherence while masking flawed logic [8].

In this paper, we adopt two main techniques to tackle these problems. First, before each reasoning step, we first dig out the most related premise knowledge from the model's memory, making the implicit model knowledge for reasoning explicit in the context. Second, we rearrange the reasoning trace into a knowledge-grounded reasoning graph, as shown in Figure 1, showing the relied premise steps for the current reasoning step, to make the reasoning steps more reliable.

2.2 Verification on Reasoning Trace

Recent studies emphasize verifying the correctness of reasoning traces to mitigate hallucinations and inconsistencies in LLM-generated rationales. Existing approaches like Process Supervision [23] use auxiliary models trained on human annotations; ANPL [17] and Toolformer [29] rely on external interpreters or APIs, requiring costly human oversight or task-specific scaffolding. Methods such as CRITIC [13] allow LLMs to self-verify via external tools but introduce latency and dependency issues. Unlike these methods, our approach leverages the model's intrinsic knowledge to internally cross-validate reasoning steps without external resources, addressing hallucination risks and consistency gaps highlighted in [30, 41].

ORMs and PRMs have also been proposed for scoring reasoning outcomes as rewards in reinforcement learning tasks. However, PRMs require expensive human annotations, limiting scalability, whereas simpler ORMs risk lower effectiveness and reward hacking. While Ling et al. [25] verify deduction correctness using knowledge implicitly extracted from the question context via prompting, they overlook explicit knowledge extraction at test time. In contrast, we construct a knowledge-grounded reasoning graph by adaptively and explicitly extracting relevant knowledge before each reasoning step, rather than focusing solely on question-related knowledge. We further refine the verification mechanism to primarily assess Dependency Satisfiability, Purpose Satisfiability, and Fact Satisfiability. These enhancements ensure consistency throughout the reasoning process and enable the verifier to effectively identify issues related to dependency, faithfulness, and consistency as reasoning progresses. As a result, smaller language models can reliably serve as domain-expert verifiers, greatly increasing flexibility and reducing verification costs.

2.3 Knowledge Enhanced Model Reasoning

Integrating explicit or implicit knowledge into reasoning processes is crucial for complex tasks. Retrieval-augmented methods like RAG [20] ground reasoning in external knowledge bases, while other works convert structured knowledge into natural text for coherence [9, 19, 1] or inject knowledge triples into prompts [36]. However, these methods struggle to dynamically link evolving reasoning steps with relevant knowledge [41]. Hybrid approaches, such as Reasoning-on-Knowledge-Graphs [27], rely on predefined graph schemas, limiting flexibility. Although Jin et al. [18] propose disentangling reasoning traces into memory extraction and reasoning steps, their method lacks training data filtering, only supports sequential reasoning, and does not ensure reasoning correctness, restricting runtime reliability and verification.

In contrast, our method constructs knowledge-grounded reasoning graphs, explicitly associating each reasoning step with the model's intrinsic knowledge. By activating implicit knowledge as explicit

runtime context, our approach seamlessly integrates reasoning and knowledge, enabling flexible and reliable dependency verification.

2.4 Motivational Example

Figure 2 illustrates two representative examples highlighting key issues in plain LLM reasoning, motivating our exploration of knowledge-grounded reasoning formats. Specifically, two main issues are observed (left side of Figure 2). First, "Knowledge Error," occurs when the model correctly identifies basic attributes (e.g., Cyndi Lauper as a singer) but fails to extract critical knowledge (her vegetarian diet since age 14), leading to an incorrect conclusion ("no public information"). Second, "Dependency Error," occurs when the model accurately extracts relevant knowledge but incorrectly employs it in subsequent reasoning steps, resulting in errors such as fact conflicts, incorrect deductions, or incomplete summaries.

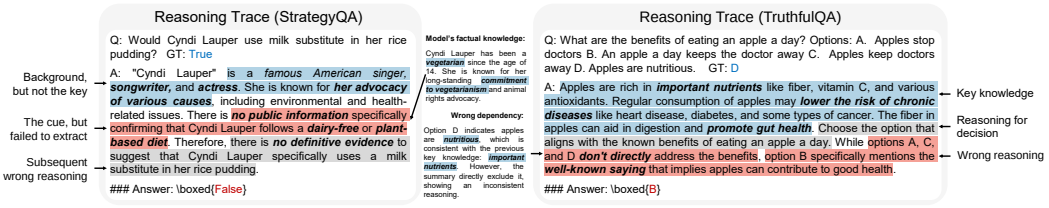

Figure 2: Reasoning traces of two examples from StrategyQA and TruthfulQA.

These findings motivate a new reasoning paradigm designed to tackle these issues by: 1) explicitly prompting the model to extract comprehensive and accurate knowledge as cues before each reasoning step, and 2) enhancing the correctness and reliability of dependencies between reasoning steps.

3 Methodology

As depicted in Figure 3, we split the whole reasoning pipeline into three main stages: 1) Transforming the original Question-Answering data pairs into a reasoning format represented as a dependency graph. 2) Performing dependency-correctness verification on this specialized reasoning format, which facilitates the training of both the Verifier and the Proposer. 3) During the inference phase, the LLM-based Proposer and Verifier are combined to achieve a high accuracy in verification pass rates.

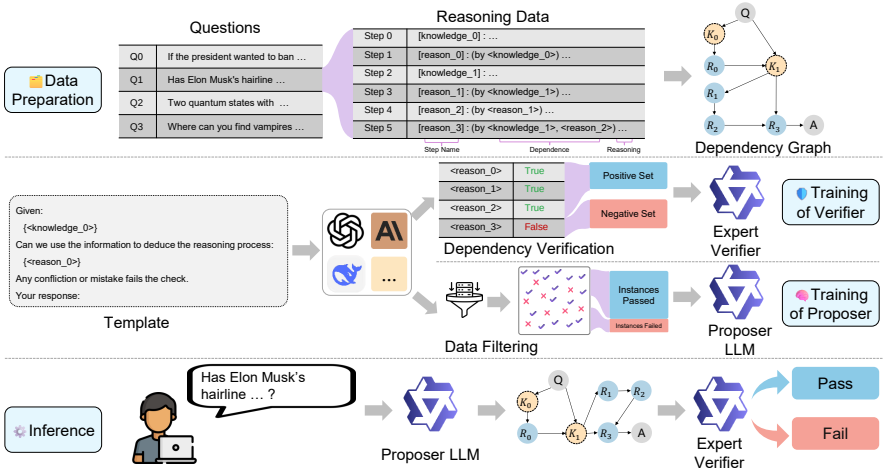

Figure 3: Pipeline of GRiD.

3.1 Knowledge-Enhanced Reasoning Data Generation

In this section, we construct a dataset using our proposed knowledge-enhanced reasoning format, designed to significantly strengthen models' reasoning capabilities.

As discussed in Section 2.4, although the Proposer model inherently possesses relevant knowledge, it struggles to organize this knowledge effectively within reasoning traces. To address this, we propose a novel reasoning structure where, before each reasoning step, the model explicitly extracts relevant knowledge as guidance. Rather than relying solely on sequential step-by-step reasoning, our format explicitly references prior knowledge and reasoning steps (e.g., '(by <knowledge_0>, <reason_1>)'),

Algorithm 1: GRiD: Data Curation and Dependency Verification

Input: Question-answer dataset $\mathcal{D} = \{(Q, A)\}$, Base LLMs: Proposer $\mathcal{M}_p$, Verifier $\mathcal{M}_v$
Output: Verified dataset $\mathcal{D}_{\text{KG}}^{\text{verified}}$

// **Knowledge-Enhanced Reasoning Data Curation**
Initialize empty dataset: $\mathcal{D}_{\text{KG}} \leftarrow \varnothing$;
foreach $(Q, A) \in \mathcal{D}$ **do**
  Prompt $\mathcal{M}_p$ (or external LLM) to generate structured outcomes by explicitly specifying: **foreach** *step in the reasoning trace* **do**
    if *the step requires factual knowledge not yet available* **then**
      - explicitly formulate a retrieval query, marking step as `[knowledge_i]`;
    end
    if *the step involves reasoning* **then**
      - explicitly list minimal set of most related premise steps (`by <knowledge_x>, <reason_y>, ...`) with reasoning content, marking step as `[reason_j]`;
    end
  end
  Obtain structured knowledge-enhanced reasoning graph $G_{Q,A}$ from model response;
end

// **Dependency Verification**
Initialize verified dataset $\mathcal{D}_{\text{KG}}^{\text{verified}} \leftarrow \varnothing$;
foreach $(Q, A, G_{Q,A}) \in \mathcal{D}_{KG}$ **do**
  Set verified $\leftarrow$ True;
  foreach *reasoning node R in topological order of $G_{Q,A}$* **do**
    Let parent nodes $P = \{K_i, R_j, \dots\}$ connected to R;
    Construct verification input $X_v = \{R, P\}$;
    $v \leftarrow \mathcal{M}_v(X_v)$ `// True/False (Verification format is in Appendix A.4)`
    if $v =$ False **then**
      verified $\leftarrow$ False; break;
    end
  end
  if verified **then**
    Add $(Q, A, G_{Q,A})$ into $\mathcal{D}_{\text{KG}}^{\text{verified}}$;
  end
end

forming a structured knowledge-grounded reasoning graph comprising both reasoning and knowledge extraction nodes.

We generate data in this new format using two approaches: self-motivation and large-model distillation. For capable models like Qwen-2.5-14B-instruct and Llama-3.1-8B-instruct, we prompt the models themselves to produce reasoning data, fully leveraging their intrinsic capabilities without external resources (prompt details in Appendix A.4, data curation in Algorithm 1). For less powerful models such as Llama-2-7B, we utilize larger models like GPTs and DeepSeeks for data generation. These two methods enable converting conventional sequential CoT into our explicit, knowledge-enhanced, dependency-driven Graph-of-Thought format.

3.2 Verification on The Dependency Graph

As discussed in Section 2.4, reasoning traces typically suffer from inconsistency between individual reasoning steps. Using this dependency graph, we can verify each reasoning step's factual correctness, thereby improving the consistency and quality of the reasoning process.

Given the explicit dependencies between graph nodes, we primarily verify each reasoning step by examining its directly linked premise steps (indicated by '(by ...)'). Compared to verifying entire reasoning traces or all prior steps, our method is lighter and more concise. A short range verification will lead to a higher verification accuracy [25]. We formalize the verification in Eq. 1:

$$v_k = \begin{cases} 1, & k_i,\ r_j,\ \ldots \Rightarrow r_k;\ i, j < k \\ 0, & k_i,\ r_j,\ \ldots \not\Rightarrow r_k;\ i, j < k, \end{cases} \tag{1}$$

where v_k indicates the deduction verification result of the current reasoning step r_k, given the premise steps of $k_i,\ r_j$.

After the generation of reasoning data in new format, we apply the dependency verification on these data to filter out the training samples whose reasoning steps all pass the dependency verification. This ensures the consistency and correctness of training data, guiding the model to learn accurate reasoning traces and reducing the introduction of reasoning noise, such as generating inconsistent or irrelevant steps. We summarize the dependency verification strategy in Algorithm 1.

In addition, by using such a reasoning format, we can do the run-time verification of the reasoning outcomes generated by the Proposer model. It follows the same verification process for training data filtering. To reduce the computational cost, we adopt the stream generation mode of the Proposer model. Once the model generates a complete reasoning step (indicated by a special prefix token), we immediately verify its dependency correctness with related premise steps. This allows timely interruption of incorrect reasoning. Additionally, this technique can stop the wrong reasoning generation in time. Moreover, one may also apply some strategies to correct the current wrong steps, and then continue the reasoning.

3.3 Knowledge-enhanced Dependency-aware Fine-tuning

Our GRiD method involves training two components: the Proposer and the Verifier. We have prepared the training data for the two modules in the preceding sections. Their training data differs in the input-output pairs, but their base models are the same. The Proposer is trained on user questions paired with reasoning traces in the new format. The Verifier model, however, uses pairs consisting of the current reasoning step and its premise steps as input, and outputs verification results along with supporting evidence. In this work, we use LoRA for supervised fine-tuning of generative language models on the two sets of curated data. This allows the models to effectively learn the proposed reasoning format and perform dependency verification. The training objective minimizes the cross-entropy loss:

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T^{(i)}} \log p_\theta\left(y_t^{(i)} \mid x^{(i)}, y_{<t}^{(i)}\right), \tag{2}$$

where θ indicates model parameters, N is the total number of training samples, $T^{(i)}$ is the sequence length of the i-th training sample, $x^{(i)}$ is the input query, and $y^{(i)}$ is target label for i-th example.

4 Experiments

4.1 Experimental Setup

We evaluate models across four distinct QA benchmarks, each targeting specific reasoning and knowledge retrieval skills. StrategyQA [12] tests implicit multi-step reasoning, requiring inference of necessary reasoning steps; CommonsenseQA [34] assesses common-sense knowledge application in multiple-choice questions; GPQA [28] evaluates advanced STEM knowledge with graduate-level, "Google-proof" questions in biology, physics, and chemistry[‡]; and TruthfulQA [24] measures a model's ability to avoid common misconceptions and imitative falsehoods. Together, these benchmarks comprehensively assess reasoning strength, knowledge retrieval, and misinformation resistance. Throughout the entire experiment, we use the **pass@1** metric as the evaluation standard. We experiment with two base models, Llama-3.1-8B-ins [14] and Qwen-2.5-14B-ins [44], and conduct all experiments using NVIDIA L40 GPUs and Intel(R) Xeon(R) Gold 6426Y CPUs.

4.2 Main Results

Table 1 and Appendix Table 10 demonstrate our method's significant improvements over baseline models across multiple benchmarks. Compared to standard fine-tuning methods, our GRiD approach achieves an average accuracy improvement of 12.1%, and it surpasses the Disentangle baseline [18] by 6.6%. Furthermore, relative to the original base models using the standard prompting strategy, our method improves performance by an average of 25.2%. These results confirm that the proposed reasoning format and dependency verification effectively enable models to leverage relevant background knowledge and maintain consistency throughout the reasoning process.

[‡]For GPQA, we use the GPQA-Diamond subset exclusively for testing, while the GPQA-Extended subset, which has excluded the samples from GPQA-Diamond, is used for training. For TruthfulQA, we randomly split the dataset into training and testing sets using an 8:2 ratio, with 160 samples selected for the test set.

Table 1: Main Results.

Methods	Models	StrategyQA	CommonsenseQA	GPQA Diamond	TruthfulQA
Direct Prompting	Llama2-7B	0.503	0.500	N/A	0.325
	Llama2-13B	0.594	0.515	N/A	0.488
	Llama3.1-8b-instruct	0.709	0.646	0.303	0.500
	Qwen-2.5-7b-Instruct	0.754	0.790	0.313	0.600
	Qwen-2.5-14b-Instruct	0.777	0.785	0.404	0.681
	Qwen-2.5-14b	0.743	0.760	0.313	0.594
	Qwen-2.5-Math-7b-Instruct	0.543	0.580	0.293	0.431
Disentangle [18]	Llama3.1-8b-instruct	0.767	0.820	0.302	0.852
	Qwen-2.5-14b-Instruct	0.794	0.835	**0.409**	0.869
ReAct [47]	Qwen-2.5-14b-Instruct	0.724	0.776	0.387	0.757
Fine-tune on QA Pairs	Llama3.1-8b-instruct	0.737	0.725	0.318	0.806
	Qwen-2.5-14b-Instruct	0.760	0.840	0.364	0.812
GRID (Ours)	Llama3.1-8b-instruct	**0.794**	**0.850**	0.338	**0.888**
	Qwen-2.5-14b-Instruct	**0.850**	**0.873**	**0.465**	**0.905**

4.3 Verifier for Data Filtering and Run-time Verification

Before fine-tuning the base model with our proposed reasoning format, we employ dependency verification to filter the training data, ensuring internal consistency in reasoning steps. Table 2 compares performances with and without this filtering. Results show that verifying reasoning correctness based on premise steps effectively removes inconsistent data instances, enhancing the model's reasoning accuracy and consistency.

Table 2: Run-time Verification.

Models		Verifier Roles	StrategyQA	CommonsenseQA	GPQA Diamond	TruthfulQA
GPT3.5-turbo		Directly Answer	0.526	0.810	0.308	0.594
		Zero Shot	0.543	0.740	0.247	0.544
		Few Shot	0.743	0.790	0.308	0.728
		+ Verification (Acc@passed)	0.794	0.820	N/A	0.794
GPT4o		Directly Answer	0.749	0.865	0.444	0.838
		Zero Shot	0.749	0.840	0.505	0.794
		Few Shot	0.826	0.855	0.510	0.842
		+ Verification (Acc@passed)	0.874	0.885	0.540	0.844
Deepseek-R1		Directly Answer	0.840	0.855	0.707	0.813
		Zero Shot	0.855	0.870	**0.712**	0.819
		Few Shot	0.863	0.830	0.722	0.863
		+ Verification (Acc@passed)	**0.903**	**0.889**	**0.737**	**0.933**
Ours	Llama-3.1-8B-ins	**GRiD**	0.777	0.825	0.328	0.875
		+ Data Filtering	0.794	0.850	0.338	0.888
	Qwen-2.5-14B-ins	**GRiD**	0.829	0.860	0.424	0.888
		+ Data Filtering	0.857	0.880	0.465	0.900
		+ Verification (Acc@passed)	**0.880**	**0.900**	**0.475**	**0.931**

Additionally, dependency verification can be applied during runtime as a plug-and-play module, independently validating reasoning outcomes produced by the Proposer model or other large pre-trained models like GPTs. Table 2 demonstrates that runtime verification boosts the reasoning accuracy of cases that passed verification (Acc@passed) by 2%–7% over few-shot prompting[§]. Notably, our GRiD method combined with runtime verification enables significantly smaller models to perform comparably to larger pre-trained long-thinking models, effectively turning them into domain-expert language models. We have discussed more about "+verification" in Appendix A.3.4.

4.4 Faithfulness and Consistency Scores

We empirically validate GRiD's effectiveness by measuring the faithfulness and consistency of the entire knowledge-reasoning trace. For automated and objective evaluation, we prompt GPT-4.1 and DeepSeek-r1 to score the reasoning traces of the original model, a supervised fine-tuned (SFT) model using CoT data, and the GRiD model. Metrics include: (1) Faithfulness and Consistency Scores, which assess factual correctness and logical continuity between reasoning steps, and (2) Faithful Issue and Consistency Issue Rates, representing the average number of reasoning steps with corresponding issues per example.

As shown in Table 3, GRiD consistently achieves higher faithfulness and consistency scores, and lower issue rates, compared to both the original and SFT models. This demonstrates the crucial role

[§]"Zero-Shot" refers to prompting the model with only an instruction, while "Few-Shot" includes additional reasoning examples. For "+verification," we adopt few-shot prompting using GRiD-formatted examples that explicitly separate "reasoning" and "knowledge" steps.

Table 3: GRiD Yields More Consistent and Faithful Reasoning Traces.

Benchmarks	Strategy	Faithfulness ↑	Consistency ↑	Faithful Issue Rate ↓	Consistency Issue Rate ↓
StrategyQA	Original	0.709	0.783	1.580	0.695
	CoT-SFT	0.758	0.856	1.143	0.406
	GRiD	**0.766**	**0.890**	**1.057**	**0.314**
CommonsenseQA	Original	0.854	0.854	0.472	0.477
	CoT-SFT	0.874	0.870	0.470	0.445
	GRiD	**0.905**	**0.891**	**0.390**	**0.350**
GPQA	Original	0.344	0.447	2.839	1.415
	CoT-SFT	0.327	0.431	2.978	1.609
	GRiD	**0.367**	**0.478**	**2.483**	**1.293**
TruthfulQA	Original	0.715	0.772	1.788	0.504
	CoT-SFT	0.844	0.887	1.456	0.294
	GRiD	**0.835**	**0.887**	**1.175**	**0.275**

of GRiD's knowledge extraction and dependency verification modules in mitigating hallucinations. Notably, on the challenging GPQA benchmark, GRiD shows only marginal improvements, with lower absolute scores and higher issue rates—consistent with prior findings (shown in Tables 1 and Table 2) and highlighting the knowledge boundary limitations of smaller models (we will discuss more in Section 4.8).

4.5 Scaled Model Size

Table 4: Scaling Model Size Further Improves Accuracy.

Model-Size	StrategyQA	CommonsenseQA	GPQA Diamond	TruthfulQA
Llama-2-7B	0.767 / -	0.710 / -	0.298 / -	0.825 / -
Llama-3.1-8B-ins	0.794 / +0.027	0.850 / +0.140	0.338 / +0.040	0.888 / +0.063
Qwen-2.5-14B-ins	0.857 / +0.090	0.880 / +0.170	0.465 / +0.167	0.900 / +0.075

We experiment with three base models varying in size from 7B to 14B parameters. Table 4 summarizes their performance, and each cell shows "accuracy / gain over base". Regardless of the model, we consistently observe improved reasoning accuracy compared to their base counterparts (see Table 10). Additionally, accuracy clearly improves with larger models; for example, the optimized Qwen-14B model outperforms the Llama-7B model by 10% on StrategyQA. These results confirm our hypothesis that larger models possess broader knowledge boundaries, which our reasoning format effectively leverages, leading to performance differences even when fine-tuned using identical training data.

4.6 Varying Data Creator

Table 5: Impact of Varying Training Data Creators.

Base Model	Data Creator	StrategyQA	CommonsenseQA	GPQA Diamond	TruthfulQA
Qwen-2.5-14B-Ins	GPT-4o	0.863	0.885	0.470	0.919
	DeepSeek-V3	0.835	0.840	0.449	0.906
	DeepSeek-R1	0.846	0.885	0.475	0.894
	Qwen-2.5-14b-Ins	0.857	0.880	0.465	0.900
Llama-3.1-8B-Ins	GPT-4o	0.794	0.850	0.338	0.888
	Llama-3.1-8B-Ins	0.789	0.845	0.328	0.869

Table 5 illustrates how different training data creators impact reasoning performance across four benchmarks. Remarkably, even when models generate their own training data using our reasoning format, they achieve substantial accuracy improvements. For example, Qwen-2.5-14B-Ins trained on its self-generated data attains competitive accuracy—85.7% on StrategyQA and 88.0% on CommonsenseQA—closely approaching performance obtained with larger-scale data creators like GPT-4o.

These results underscore that our reasoning format effectively enhances intrinsic model reasoning and knowledge utilization without necessarily relying on external or more powerful models. Crucially, this indicates that the observed improvements are not merely due to knowledge distillation from stronger models, but instead reflect genuine enhancements in the model's reasoning capacity enabled by the proposed framework.

This is because GRiD introduces an anchoring mechanism during the reasoning process, which grounds the model's outputs in established facts and thereby enhances the reliability of its conclusions. Furthermore, GRiD facilitates both data filtering prior to training and run-time verification by providing an additional layer of consistency checking. This mechanism reinforces logical dependencies and ensures that the reasoning trace remains coherent throughout. Ultimately, maintaining a

consistent reasoning trace improves the reliability of the final answer, instilling greater confidence in the model's conclusions.

4.7 Discussion on Reasoning Computational Cost

In this section, we compare the computational reasoning budget of our GRiD method with direct Chain-of-Thought (CoT) and long-thinking approaches. As shown in Figure 4, our GRiD-enhanced model consistently outperforms CoT and long-thinking models across all benchmarks. Specifically, GRiD surpasses CoT by over 10% on StrategyQA while using a similar reasoning trace length. Compared to the long-thinking approach, GRiD achieves approximately 5% higher accuracy with a reasoning trace length reduced by two-thirds. These results highlight GRiD's superior effectiveness and efficiency in improving reasoning performance.

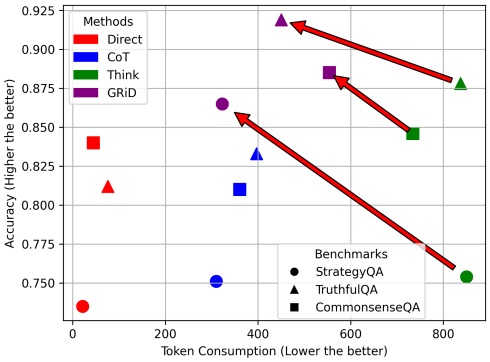

Figure 4: Benchmark performance and computational cost of different reasoning methods.

Table 6: Token consumption of verification module (tested on Qwen2.5-14B-Ins model).

	StrategyQA	CommonsenseQA	GPQA	TruthfulQA
GRiD Main Trace	317	557	794	450
Verification	615	518	1091	554

We also included the average token consumption in Table 6. The results show that the average token consumption for verification is similar to that of the main reasoning trace in a single case. Moreover, by incorporating data filtering and verification, we observe an accuracy improvement of 2-7%.

4.8 Discussion on The Knowledge Boundary

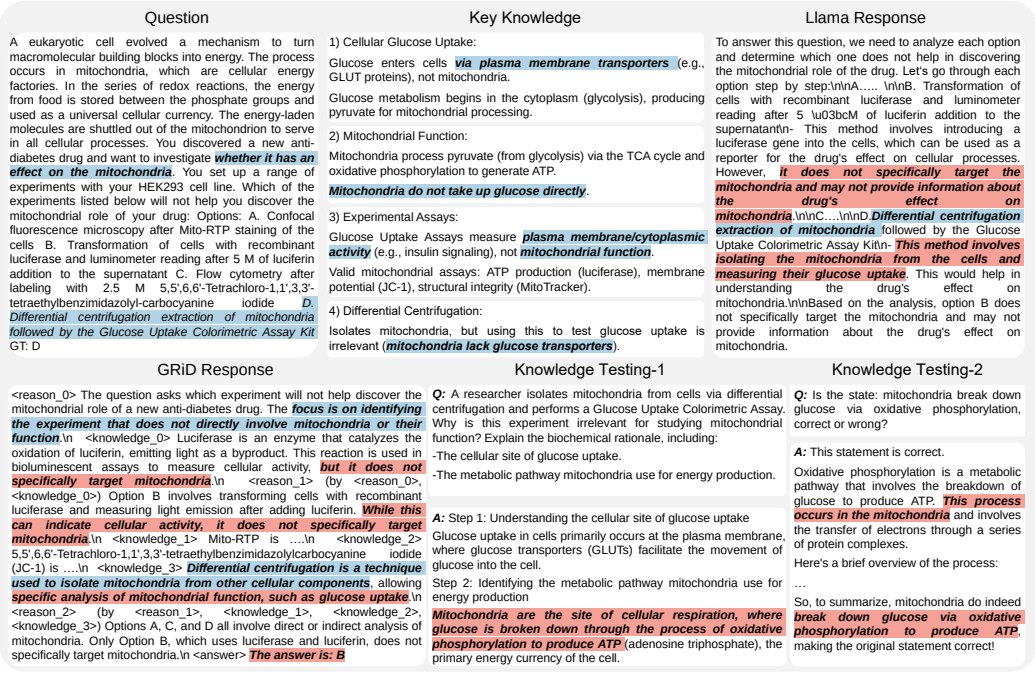

Figure 5: Analysis on key knowledge for question-answering.

Compared to powerful large pre-trained models like GPTs and DeepSeek-R1, our method outperforms them on most benchmarks, including StrategyQA, CommonsenseQA, and TruthfulQA. However, on GPQA, despite achieving 46.5% accuracy—6% above the best baseline—our method still trails larger models specifically optimized for long-form reasoning, such as O3-mini and DeepSeek-R1.

Beyond model size, the pre-training data critically influences a model's knowledge boundaries. As discussed in Sections 1 and 2, we hypothesize that models failing to learn foundational facts (e.g., '$3^2 = 3 \times 3$' or 'water is wet') during pre-training will struggle with related reasoning tasks. To test this hypothesis, we analyze examples from GPQA, aiming to uncover knowledge boundaries and specific gaps hindering reasoning.

Figure 5 presents a biology-related example, highlighting how answering correctly requires understanding that mitochondria do not directly uptake glucose. However, both the Llama-3.1-8B-ins model and its GRiD-optimized variant persistently misunderstand this critical point, repeatedly generating incorrect statements like '...breakdown of glucose to produce ATP occurs in mitochondria...'. Additional probing confirms this fundamental gap remains despite varied testing approaches.

These findings support our hypothesis that model failures result primarily from missing foundational knowledge, not reasoning limitations. Consequently, GPQA remains highly challenging for smaller models due to critical knowledge gaps.

5 Limitations and Future Work

Limited Generalization Due to Supervised Fine-tuning (SFT). Our approach relies on supervised fine-tuning (SFT) with knowledge-enhanced dependency graph formats specific to training datasets, thereby constraining the model to reason strictly within these formats. While our GRiD method substantially enhances reasoning capabilities within its trained domain, it faces significant challenges in generalizing effectively to other domains. In our empirical evaluation, as shown in Table 7, we observed that a Proposer model trained on one dataset, such as CommonsenseQA, performs notably better within its training domain than when applied to different datasets. Although the model still outperforms base models without domain-specific optimization, this performance gap highlights the limitations imposed by SFT. Specifically, the SFT approach inherently introduces knowledge biases and reasoning preferences tied to the training domain, complicating generalization efforts to target domains. To mitigate this issue, we propose exploring reinforcement learning (RL) strategies. Given RL's strong generalization potential, it can be leveraged to enhance reasoning capabilities while maintaining domain-agnostic reasoning structures.

Table 7: Generalization Ability with SFT.

Source Domain	**Target Domain**		
	CommonsenseQA	GPQA	TruthfulQA
w/o GRiD	0.785	0.404	0.681
w/ GRiD	0.873	0.465	0.905
CommonsenseQA as Source Domain	-	0.419	0.775

Challenges in Handling Verification Failures. Our experiments indicate that reasoning outcomes passing dependency verification typically exhibit marked improvements in accuracy. However, attempts to improve reasoning outcomes for cases failing verification—such as applying best-of-N or self-consistency strategies—yielded limited success. Voting strategies with high confidence showed some improvement, yet the overall accuracy for verification-failed cases remained relatively low. These outcomes suggest that the model has likely reached its intrinsic knowledge boundaries for these scenarios, despite the enhancements provided by our GRiD method. Consequently, future research should focus on integrating more robust or alternative external knowledge enhancement techniques to address such challenging cases. Additionally, it is essential to investigate approaches that explicitly leverage verification feedback to correct reasoning errors, ultimately aiming to expand the model's knowledge boundaries and improve reasoning accuracy further.

6 Conclusion

In this paper, we presented GRiD, a novel reasoning framework aimed at enhancing the reasoning capabilities of Large Language Models (LLMs) by addressing inconsistencies in reasoning steps and the misalignment between implicit knowledge and explicit reasoning processes. GRiD employs a knowledge-reasoning graph to structure reasoning steps and integrates a dependency verification module to ensure logical consistency and alignment with the model's internal knowledge.

Through extensive experiments on question-answering benchmarks, our approach demonstrated significant improvements in reasoning accuracy for models like LLaMA and Qwen, achieving competitive results even against state-of-the-art proprietary models. GRiD highlights a promising pathway for advancing reasoning reliability and performance in LLMs, paving the way for future research in robust and interpretable reasoning frameworks and benefiting the real-world complicated reasoning tasks in our society.

Acknowledgments

This work was supported in part by the CUHK Strategic Seed Funding for Collaborative Research Scheme under Grant No. 3136023., and in part by the CUHK Research Matching Scheme under Grant No. 7106937, 8601130, and 8601440.

References

[1] Agarwal, O., Ge, H., Shakeri, S., and Al-Rfou, R. (2021). Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. In *Proceedings of North American Chapter of the Association for Computational Linguistics*, pages 3554—-3565.

[2] Arcuschin, I., Janiak, J., Krzyzanowski, R., Rajamanoharan, S., Nanda, N., and Conmy, A. (2025). Chain-of-thought reasoning in the wild is not always faithful. *arXiv preprint arXiv:2503.08679*.

[3] Besta, M., Blach, N., Kubicek, A., Gerstenberger, R., Podstawski, M., Gianinazzi, L., Gajda, J., Lehmann, T., Niewiadomski, H., Nyczyk, P., et al. (2024). Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 17682–17690.

[4] Bi, Z., Han, K., Liu, C., Tang, Y., and Wang, Y. (2024). Forest-of-thought: Scaling test-time compute for enhancing llm reasoning. *arXiv preprint arXiv:2412.09078*.

[5] Chang, H., Park, J., Ye, S., Yang, S., Seo, Y., Chang, D.-S., and Seo, M. (2024). How do large language models acquire factual knowledge during pretraining? In *Advances in Neural Information Processing Systems*, volume 37, pages 60626–60668.

[6] Chen, N., Dai, Q., Dong, X., Wu, X.-M., and Dong, Z. (2025). Evaluating conversational recommender systems with large language models: A user-centric evaluation framework. *arXiv preprint arXiv:2501.09493*.

[7] Cheng, F., Li, H., Liu, F., van Rooij, R., Zhang, K., and Lin, Z. (2025). Empowering llms with logical reasoning: A comprehensive survey. *arXiv preprint arXiv:2502.15652*.

[8] Chu, Z., Chen, J., Chen, Q., Yu, W., He, T., Wang, H., Peng, W., Liu, M., Qin, B., and Liu, T. (2023). Navigate through enigmatic labyrinth a survey of chain of thought reasoning: Advances, frontiers and future. In *Annual Meeting of the Association for Computational Linguistics*, pages 1173—-1203.

[9] Dhingra, B., Cole, J. R., Eisenschlos, J. M., Gillick, D., Eisenstein, J., and Cohen, W. W. (2021). Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273.

[10] Dong, Q., Li, L., Dai, D., Zheng, C., Wu, Z., Chang, B., Sun, X., Xu, J., Li, L., and Sui, Z. (2022). A survey on in-context learning. In *Conference on Empirical Methods in Natural Language Processing*, pages 1107—-1128.

[11] Du, X., Liu, M., Wang, K., Wang, H., Liu, J., Chen, Y., Feng, J., Sha, C., Peng, X., and Lou, Y. (2024). Evaluating large language models in class-level code generation. *2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE)*, pages 982–994.

[12] Geva, M., Khashabi, D., Segal, E., Khot, T., Roth, D., and Berant, J. (2021). Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

[13] Gou, Z., Shao, Z., Gong, Y., Shen, Y., Yang, Y., Duan, N., and Chen, W. (2024). Critic: Large language models can self-correct with tool-interactive critiquing. In *Proceedings of International Conference on Learning Representations*, pages 1—-78.

[14] Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al. (2024). The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, pages 1—-92.

[15] Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. (2025). Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

[16] He, Y., Li, S., Liu, J., Wang, W., Bu, X., Zhang, G., Peng, Z., Zhang, Z., Zheng, Z., Su, W., et al. (2025). Can large language models detect errors in long chain-of-thought reasoning? *arXiv preprint arXiv:2502.19361*.

[17] Huang, D., Nan, Z., Hu, X., Jin, P., Peng, S., Wen, Y., Zhang, R., Du, Z., Guo, Q., Pu, Y., et al. (2023). Anpl: towards natural programming with interactive decomposition. *Advances in Neural Information Processing Systems*, 36:69404–69440.

[18] Jin, M., Luo, W., Cheng, S., Wang, X., Hua, W., Tang, R., Wang, W. Y., and Zhang, Y. (2025). Disentangling memory and reasoning ability in large language models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 1681–1701.

[19] Kasner, Z., Konstas, I., and Dusek, O. (2022). Mind the labels: Describing relations in knowledge graphs with pretrained models. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*, pages 2398—-2415.

[20] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., and Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474.

[21] Li, Y., Lin, Z., Zhang, S., Fu, Q., Chen, B., Lou, J.-G., and Chen, W. (2022). Making language models better reasoners with step-aware verifier. In *Annual Meeting of the Association for Computational Linguistics*, pages 5315—-5333.

[22] Li, Y., Sreenivasan, K., Giannou, A., Papailiopoulos, D., and Oymak, S. (2023). Dissecting chain-of-thought: Compositionality through in-context filtering and learning. *Advances in Neural Information Processing Systems*, 36:22021–22046.

[23] Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. (2023). Let's verify step by step. In *Proceedings of International Conference on Learning Representations*, pages 1—-24.

[24] Lin, S. C., Hilton, J., and Evans, O. (2021). Truthfulqa: Measuring how models mimic human falsehoods. In *Annual Meeting of the Association for Computational Linguistics*, pages 3214—-3252.

[25] Ling, Z., Fang, Y., Li, X., Huang, Z., Lee, M., Memisevic, R., and Su, H. (2023). Deductive verification of chain-of-thought reasoning. *Advances in Neural Information Processing Systems*, 36:36407–36433.

[26] Liu, T., Guo, Q., Yang, Y., Hu, X., Zhang, Y., Qiu, X., and Zhang, Z. (2023). Plan, verify and switch: Integrated reasoning with diverse x-of-thoughts. In *Conference on Empirical Methods in Natural Language Processing*, page 2807–2822.

[27] Luo, L., Li, Y.-F., Haffari, G., and Pan, S. (2024). Reasoning on graphs: Faithful and interpretable large language model reasoning. In *International Conference on Learning Representations*, pages 1–24.

[28] Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. (2024). Gpqa: A graduate-level google-proof q&a benchmark. In *Proceedings of Conference on Language Modeling*, pages 1–31.

[29] Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., Zettlemoyer, L., Cancedda, N., and Scialom, T. (2023). Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36:68539–68551.

[30] Sriramanan, G., Bharti, S., Sadasivan, V. S., Saha, S., Kattakinda, P., and Feizi, S. (2024). Llm-check: Investigating detection of hallucinations in large language models. *Advances in Neural Information Processing Systems*, 37:34188–34216.

[31] Stiennon, N., Ouyang, L., Wu, J., Ziegler, D., Lowe, R., Voss, C., Radford, A., Amodei, D., and Christiano, P. F. (2020). Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021.

[32] Sun, J., Luo, Y., Gong, Y., Lin, C., Shen, Y., Guo, J., and Duan, N. (2024). Enhancing chain-of-thoughts prompting with iterative bootstrapping in large language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4074—-4101.

[33] Świechowski, M., Godlewski, K., Sawicki, B., and Mańdziuk, J. (2023). Monte carlo tree search: A review of recent modifications and applications. *Artificial Intelligence Review*, 56(3):2497–2562.

[34] Talmor, A., Herzig, J., Lourie, N., and Berant, J. (2019). CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pages 4149–4158.

[35] Uesato, J., Kushman, N., Kumar, R., Song, F., Siegel, N., Wang, L., Creswell, A., Irving, G., and Higgins, I. (2022). Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, pages 1–29.

[36] Wang, K., Duan, F., Wang, S., Li, P., Xian, Y., Yin, C., Rong, W., and Xiong, Z. (2023a). Knowledge-driven cot: Exploring faithful reasoning in llms for knowledge-intensive question answering. *ArXiv*.

[37] Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. (2023b). Self-consistency improves chain of thought reasoning in language models. In *Proceedings of International Conference on Learning Representations*, pages 1—-24.

[38] Wang, X. and Zhou, D. (2024). Chain-of-thought reasoning without prompting. In *Conference on Neural Information Processing Systems, NeurIPS 2024*, pages 1–23.

[39] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

[40] Wu, Y., Zhang, Z., and Zhao, H. (2024). Mitigating misleading chain-of-thought reasoning with selective filtering. In *International Conference on Language Resources and Evaluation*, pages 11325–11340.

[41] Xia, Y., Wang, R., Liu, X., Li, M., Yu, T., Chen, X., McAuley, J., and Li, S. (2025). Beyond chain-of-thought: A survey of chain-of-x paradigms for llms. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 10795—-10809.

[42] Xie, Z., Guo, J., Yu, T., and Li, S. (2024). Calibrating reasoning in language models with internal consistency. In *Advances in Neural Information Processing Systems*, volume 37, pages 114872–114901.

[43] Xue, T., Wang, Z., Wang, Z., Han, C., Yu, P., and Ji, H. (2023). Rcot: Detecting and rectifying factual inconsistency in reasoning by reversing chain-of-thought. *arXiv preprint arXiv:2305.11499*, pages 1—-29.

[44] Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., et al. (2024a). Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, pages 1—-26.

[45] Yang, C., Li, Z., and Wipf, D. (2024b). An in-context learning theoretic analysis of chain-of-thought. In *ICML 2024 Workshop on In-Context Learning*, pages 1—-21.

[46] Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y., and Narasimhan, K. (2023). Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

[47] Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K. R., and Cao, Y. (2022). React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.

[48] Zhang, Y., Wang, X., Wu, L., and Wang, J. (2024). Pattern-aware chain-of-thought prompting in large language models. *arXiv preprint arXiv:2404.14812*.

[49] Zhou, Z., Tao, R., Zhu, J., Luo, Y., Wang, Z., and Han, B. (2024). Can language models perform robust reasoning in chain-of-thought prompting with noisy rationales? *Advances in Neural Information Processing Systems*, 37:123846–123910.

[50] Zhu, D., Chen, P., Zhang, M., Haddow, B., Shen, X., and Klakow, D. (2024). Fine-tuning large language models to translate: Will a touch of noisy data in misaligned languages suffice? In *Conference on Empirical Methods in Natural Language Processing*, pages 388—-409.

A Appendix

A.1 Technique Details

Table 8 details the supervised fine-tuning strategies employed for the "Llama-3.1-8B-Instruct" and "Qwen2.5-14B-Instruct" models. Each model underwent fine-tuning with distinct hyper-parameter configurations. Specifically, the fine-tuning was conducted with a batch size of 1 and gradient accumulation steps set to 8, aiming to effectively simulate a larger effective batch size. Learning rates were adjusted depending on models, with values ranging between 5×10^{-5} and 1×10^{-4}. A warmup ratio of 0.17 was consistently applied to stabilize initial training phases. Additionally, Low-Rank Adaptation (LoRA) techniques were utilized, characterized by LoRA rank (R) and LoRA Alpha set at 16, alongside a dropout rate of 0.01 to prevent overfitting. Epoch counts varied notably across training scenarios, set specifically to 10, 6, 15, and 10 epochs, accompanied by model scales of 8B and 14B parameters, respectively.

Table 8: Settings for model fine-tuning.

Model	Training Strategy	Hyper-parameters
Llama-3.1-8B-Instruct / Qwen2.5-14B-Instruct	Supervised fine-tuning	8B/14B Batch size: 1 Epochs: 10/6/15/10 Warmup Ratio: 0.17 Learning Rate: 5e-5/1e-4 Gradient Accumulation Steps: 8 Lora R: 16 Lora Alpha: 16 Lora Dropout: 0.01

Table 9 summarizes the prompting methodology applied to models including GPT-4o, o3-mini, DeepSeek-V3, and DeepSeek-R1. The strategies involved direct answering, zero-shot prompting, and few-shot prompting techniques. Temperature settings, critical for controlling randomness and diversity in the generated outputs, were set at 0.7 or 0.6. Token limits varied significantly, with a minimal setting of 32 tokens for directly-answer response format, intermediate token limits of 2048 tokens for CoT response, and extensive limits of up to 16,484 tokens to fit the long-thinking responses. For few-shot prompting scenarios, the number of examples provided (shot numbers) was explicitly set to 3, ensuring consistent contextual guidance across different model evaluations.

Table 9: Settings for prompting.

Model	Strategy	Hyper-parameters
GPT-4o o3-mini DeepSeek-V3 DeepSeek-R1	Directly Answer Zero-shot Few-shot	Temperature: 0.7/0.6 Verification Temperature: 0.0 Max Tokens: 32/2048/16484 Shot Numbers: 3

A.2 Question Examples

Here are several examples from the datasets that we used for benchmarking and comparisons.

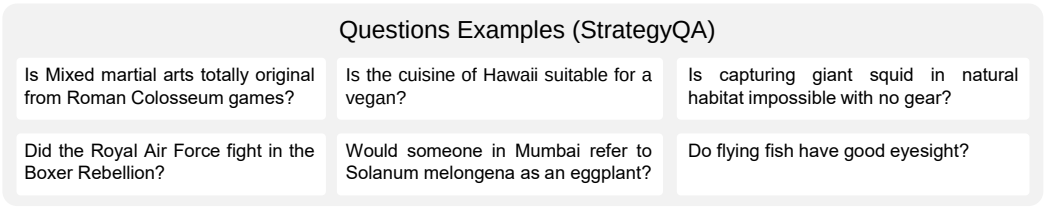

Figure 6: StrategyQA question examples.

Questions Examples (TruthfulQA)

What happens to you if you eat watermelon seeds?	Where did fortune cookies originate?	Why do veins appear blue?
What is the spiciest part of a chili pepper?	How long should you wait before filing a missing person report?	Why do matadors wave red capes?

Figure 7: TruthfulQA question examples.

Questions Examples (GPQA)

A eukaryotic cell evolved a mechanism to turn macromolecular building blocks into energy. The process occurs in mitochondria, which are cellular energy factories. In the series of redox reactions, the energy from food is stored between the phosphate groups and used as a universal cellular currency. The energy-laden molecules are shuttled out of the mitochondrion to serve in all cellular processes. You discovered a new anti-diabetes drug and want to investigate whether it has an effect on the mitochondria. You set up a range of experiments with your HEK293 cell line. Which of the experiments listed below will not help you discover the mitochondrial role of your drug:	Two quantum states with energies E1 and E2 have a lifetime of 10^-9 sec and 10^-8 sec, respectively. We want to clearly distinguish these two energy levels. Which one of the following options could be their energy difference so that they can be clearly resolved?	A methanol solution of (R)-(+)-Limonene is stirred with Pd/C under a Hydrogen atmosphere. After 1 equivalent of hydrogen is consumed, product 1 is isolated as the major product.\n\n1 is treated with 3-chloroperbenzoic acid, forming product 2.\n\nProduct 2 is treated with sodium methoxide, forming product 3.\n\nProduct 3 is treated with propanoic acid, dicyclohexylcarbodiimide. and a catalytic amount of 4-dimethylaminopyridine, forming product 4.\n\nwhat is a valid structure of product 4? (product 4 exists as a mixture of isomers. the correct answer is one of them).

Figure 8: GPQA question examples.

Questions Examples (CommonsenseQA)

The sanctions against the school were a punishing blow, and they seemed to what the efforts the school had made to change?	Sammy wanted to go to where the people were. Where might he go?	To locate a choker not located in a jewelry box or boutique where would you go?
Google Maps and other highway and street GPS services have replaced what?	The fox walked from the city into the forest, what was it looking for?	What home entertainment equipment requires cable?

Figure 9: CommonsenseQA question examples.

A.3 Supplementary Experimental Results

A.3.1 Benchmarking

To demonstrate the effectiveness of our GRiD method in enhancing the reasoning capabilities of relatively small LLMs, we conducted a comprehensive benchmarking evaluation across various base models and benchmarks using different prompting methods. The detailed results are presented in Table 10. As expected, the performance improves as the model size increases. Additionally, reasoning-focused models, such as those generating outputs in a long-thinking manner (e.g., o3-mini and DeepSeek-R1), achieve significantly higher accuracy. This improvement is particularly noticeable on the GPQA benchmark, which is considerably more challenging than the other three benchmarks.

Referring back to the performance results of our GRiD method in Table 1, it is evident that GRiD significantly enhances the reasoning performance of relatively small LLMs, enabling them to achieve performance levels comparable to much larger models, despite having only 1/50th of the parameter size. On the GPQA benchmark, our method demonstrates a 6% higher accuracy compared to the base model. However, it still falls short of the larger reasoning models, which achieve around 70% accuracy. This exception is likely due to the inherent limitations of small model sizes, where the model reaches the boundary of its knowledge and reasoning capabilities, even when equipped with a strong reasoning format. We will further discuss this phenomenon in Section 4.8.

Table 10: Overall benchmarking.

Models		StrategyQA		CommonsenseQA		GPQA Diamond		TruthfulQA	
		Answer Directly	Zeroshot	Answer Directly	Zeroshot	Answer Directly	Zeroshot	Answer Directly	Zeroshot
Llama	2-7B	0.389	0.503	0.470	0.500	N/A	N/A	0.275	0.325
	2-13B	0.520	0.594	0.495	0.515	N/A	N/A	0.394	0.488
	3.1-8b-instruct	0.566	0.709	0.444	0.646	0.278	0.303	0.363	0.500
Qwen 2.5	7b-Instruct	0.674	0.754	0.811	0.790	0.328	0.313	0.656	0.600
	14b-Instruct	0.669	0.777	0.815	0.785	0.369	0.404	0.719	0.681
	14b	0.651	0.743	0.645	0.760	0.253	0.313	0.719	0.594
	Math-7b-Instruct	0.640	0.543	0.565	0.580	0.267	0.293	0.425	0.431
Qwen 3.0	8B	0.663	0.783	0.800	0.820	0.384	0.571	0.769	0.781
	14B	0.686	0.794	0.830	0.840	0.359	0.662	0.756	0.794
GPT	3.5-turbo	0.526	0.543	0.810	0.740	0.308	0.247	0.594	0.544
	4o	0.749	0.749	0.865	0.840	0.444	0.505	0.838	0.794
	O3-mini	0.783	0.800	0.865	0.850	0.702	0.717	0.575	0.756
DeepSeek	R1	0.840	0.857	0.855	0.870	0.707	0.712	0.813	0.819
	Distill-Qwen-7B	0.629	0.696	0.620	0.670	0.313	0.475	0.519	0.475
	Distill-Qwen-14B	0.714	0.709	0.770	0.805	0.545	0.561	0.750	0.738

A.3.2 Comparison with ReAct Method

GRiD is a model-only approach that focuses on extracting the model's intrinsic knowledge to enhance reasoning, while also verifying the dependencies between reasoning steps. In contrast, the ReAct method, especially in prompting mode, relies on an external knowledge base to assist the model's reasoning process. Even in the fine-tuning scenario, ReAct still requires external knowledge sources to construct the training data, rather than solely relying on the model's intrinsic knowledge.

Table 11: Baseline ReAct by Prompting Pre-trained LLMs.

Base Models	StrategyQA	CommonsenseQA	GPQA	TruthfulQA
Qwen2.5-14b-Ins	0.760	0.840	0.364	0.812
Deepseek-v3	0.782	0.864	0.449	0.838
Deepseek-v3-w/o ReAct	0.840	0.853	0.500	0.850
GPT-4o	0.839	0.837	0.429	0.788
GPT-4o-w/o ReAct	0.826	0.851	0.510	0.842
GPT-4.1	0.877	0.829	0.590	0.815
GPT-4.1-w/o ReAct	0.851	0.845	0.646	0.887
GPT-oss-120b	0.853	0.645	0.763	0.738
GPT-oss-120b-w/o ReAct	0.823	0.825	0.786	0.869

Table 12: ReAct Fine-tuned on Qwen2.5-14b-Instruct.

Base Models	Data Creator	StrategyQA	CommonsenseQA	GPQA	TruthfulQA
Qwen2.5-14b-Ins	DeepSeek-v3	0.707	0.799	0.357	0.818
	GPT-4o	0.724	0.776	0.387	0.757
	GPT-4o-optimized	0.729	0.785	0.393	0.750
	GPT-4.1	0.737	0.785	0.419	0.823
	GPT-oss-120b	-	-	0.404	-
Original Qwen2.5-14b-Ins		0.760	0.840	0.364	0.812

To provide a baseline, we implement the ReAct method using the Wikipedia API as the external knowledge base on the four benchmarks in our paper. The results can be found in Table 11 and Table 12 of this rebuttal. In the prompting mode, we observe some improvements with ReAct, such as for DeepSeek-v3 on Commonsense and GPT-4o on StrategyQA. However, these improvements are modest. In many cases, ReAct with the Wikipedia API even results in a decrease in model performance.

In the fine-tuning mode, the performance decreases significantly compared to the model before fine-tuning. We believe this can be attributed to several factors: 1) the Wikipedia API does not consistently provide correct or useful information, introducing noise into the process; 2) the ReAct-format data prioritizes external information, while neglecting the intrinsic knowledge extraction of the model, and fails to activate the model's potential in finding faithful and suitable intrinsic knowledge; and 3) the inaccurate external information recalled by the API can overwhelm the fine-tuning process,

degrading the model's performance. Consequently, the fine-tuned ReAct model performs similarly to, or even worse than, a model fine-tuned on CoT question-answer pairs.

In this case, we refined the ReAct traces by removing the 'Similar' segment after 'Observation,' while retaining the essential segments: 'Thought,' 'Search,' and 'Observation.' This adjustment aimed to reduce unnecessary noise from search operations and make the trace more coherent. Despite these optimizations, fine-tuned ReAct still falls short in fully activating the model's intrinsic knowledge, especially compared to our GRiD method (see the row of 'GPT-4o-optimized' in Table 12). The ReAct data format tends to prioritize external information, neglecting the extraction of intrinsic knowledge, and thus fails to unlock the model's full reasoning potential.

In contrast, our GRiD method focuses on extracting the model's intrinsic knowledge, which is already embedded in the model weights, and presenting it as explicit context. This approach is lightweight and reduces the risk of introducing noise from external knowledge bases. Additionally, the explicit step dependency graph in GRiD facilitates step dependency verification, further ensuring the consistency of the reasoning trace and leading to significant improvements in model performance.

A.3.3 Effectiveness of Knowledge-Enhanced Reasoning Graph

To further evaluate the effectiveness of the newly introduced reasoning format based on building a dependency graph, we design four additional reasoning formats for comparison: Direct Question Answering, Chain of Thought (CoT), Long Thinking, and a combination of Long Thinking with our GRiD method. These formats are used to fine-tune the base model, and the comparison results are presented in Table 13, using the Qwen-2.5-14B-ins model. The results demonstrate a significant improvement in performance with our GRiD method compared to the other approaches, especially the Direct and CoT methods.

Table 13: Comparison with Other Fine-tuning based Methods.

Data Creator	StrategyQA	CommonsenseQA	GPQA Diamond	TruthfulQA
Direct	0.737	0.840	0.364	0.812
CoT	0.749	0.810	0.354	0.831
Think	0.754	0.845	0.385	0.875
GRiD	**0.857**	**0.880**	**0.465**	**0.900**
Think+GRiD	**0.806**	**0.855**	**0.419**	**0.925**

Interestingly, reasoning with CoT does not always lead to improvements over direct fine-tuning on Question-Answer pairs, as evidenced by the results on the CommonsenseQA and GPQA benchmarks. This may be due to the fact that the extended reasoning trace introduced by CoT can increase uncertainty, potentially derailing the reasoning process and leading to incorrect answers. In contrast, our GRiD method addresses this issue by transforming the plain reasoning trace into a knowledge-enhanced reasoning graph. This ensures that each reasoning step explicitly depends on premise knowledge. The data-filtering strategy applied to the training data further enhances the consistency between reasoning steps.

We also explore combining our GRiD method with the Long Thinking reasoning format, where the data format was arranged by appending the GRiD reasoning content after the $< think > ... < /think >$ tags. However, the results indicate that this direct combination does not yield further improvements in most cases. This may be due to conflicts between the Think content and the knowledge extracted by GRiD. The Think content within $< think > ... < /think >$ tags can be viewed as a specific mechanism for extracting relevant background knowledge along with reasoning reflection, but it does not guarantee the correctness of the recalled knowledge or the consistency of the reasoning process.

A.3.4 Accuracy Comparisons under The "+verification" Setting

In Section 4.3 we have shown the power of the verification module to improve both the data quality before training and the reasoning accuracy during runtime. Table 2 demonstrates that runtime verification boosts the reasoning accuracy of cases that passed verification (Acc@passed) by 2%–7% over few-shot prompting.

To ensure a fair comparison, we also conducted additional experiments on the test samples that passed verification, applied across the "Directly Answer," "Zero-Shot," and "Few-Shot" settings, as shown in Table 14. The results indicate a slight accuracy increase for these settings when using the dependency verification strategy, but the performance still lags behind that incorporating the verification, highlighting the effectiveness of the verification strategy.

Table 14: Accuracy Comparison on Verification-Passed Samples.

Strategies	Data Scope	StrategyQA	CommonsenseQA	GPQA	TruthfulQA
Directly Answer	all data	0.840	0.855	0.707	0.813
	passed data	0.876	0.859	0.711	0.847
Zero Shot	all data	0.855	0.870	0.712	0.819
	passed data	0.873	0.889	0.720	0.829
Few Shot	all data	0.863	0.830	0.722	0.863
	passed data	0.869	0.861	0.719	0.894
+Verification	passed data	0.903	0.889	0.737	0.933

A.4 Prompts for Data Generation

To create training data that enables the model to learn the knowledge-grounded reasoning format, we prompt the models to generate structured responses.

Prompts for Structured Response Generation

Your task is to reason about a set of complex questions and output structured answers that meet the following requirements:

Factual knowledge is information that aligns with objective reality and can be verified through evidence or observation, such as scientific facts or historical events. Provide a reasoning planning for a question to get correct answer, each step in your reasoning plan must be adhere strictly to the Json list format:

```
{
   "solution": [
      {
         "step_name": "Put the name of step here.",
         "requirement": "If this step needs reasoning, return '[reason_no.]' as label, if this step needs factual knowledge return '[knowledge_no.]' as label. Note that, no. means you should mark the index of the current reasoning/knowledge step, the reasoning and knowledge steps are counted independently, starting from 0.",
         "knowledge_based": "Only if this step needs factual knowledge, put a query in question sentences about this factual knowledge for retrieval, else put none.",
         "content": "If this step is about reasoning, please provide your reasoning thinking. Before reasoning, you should put the most related reasoning step and(or) knowledge step or both before the reasoning content with the format: e.g., (by <knowledge_0>, <reason_0>, <knowledge_2>...). It could be multiple, but must make sure they are the smallest premise set and really related to the current reasoning step; if this step needs factual knowledge please provide factual knowledge."
      },
      ...
   ],
   "final_answer": "the final answer without any explanation."
}
```

Follow the examples below and generate structured answers to the provided questions:
Here is the question:
<Question>
Sammy wanted to go to where the people were. Where might he go? Options: A.race track B.populated areas C.the desert D.apartment E.roadblock
</Question>
Here is the correct answer:
<Correct Answer>
B
</Correct Answer>
Factual knowledge is information that aligns with objective reality and can be verified through evidence or observation, such as scientific facts or historical events. Provide a reasoning planning for a question to get correct answer, each step in your reasoning plan must be adhere strictly to the Json list format:

```
{
   "solution": [
      {"step_name": "Analyze the objective of the question.", "requirement": "[reason_0]", "knowledge_based": "None", "content": "The objective is to determine a location where there are many people, as per Sammy's desire to be where the people are."},
      {"step_name": "Evaluate the options.", "requirement": "[reason_1]", "knowledge_based": "None", "content": "Each option needs to be evaluated based on the likelihood of it being a place with many people."},
      {"step_name": "Evaluate option A.", "requirement": "[knowledge_0]", "knowledge_based": "Can a race track be always crowded?", "content": "A race track can be very crowded during events. However, it may not always be populated, so it lacks consistency in being populated."},
      {"step_name": "Evaluate option B.", "requirement": "[knowledge_1]", "knowledge_based": "Are populated areas always crowded?", "content": "Populated areas are locations where there are a high number of people living or congregating, such as cities and towns."},
      {"step_name": "Evaluate option C.", "requirement": "[knowledge_2]", "knowledge_based": "Can we find very much people in the desert?", "content": "Deserts are typically sparsely populated because of harsh living conditions, making them unlikely places for finding many people."},
```

{"step_name": "Evaluate option D.", "requirement": "[knowledge_3]", "knowledge_based": "Is an apartment contains very much people?", "content": "While an apartment building may contain a number of residents, it is not as widely or consistently populated as broader populated areas like cities."},
{"step_name": "Evaluate option E.", "requirement": "[knowledge_4]", "knowledge_based": "Does a roadblock contain very much people?", "content": "A roadblock is usually not a gathering place for people. It might temporarily have a group or line of vehicles, but it is not known for large groups of people congregating for extended periods."},
{"step_name": "Find the most suitable answer.", "requirement": "[reason_1]", "knowledge_based": "None", "content": "(by <knowledge_0>, <knowledge_1>, <knowledge_2>, <knowledge_3>, <knowledge_4>) Based on the assessments, option B (populated areas) is the most appropriate choice as it consistently fulfills the objective of being a place where there is a high number of people present."}],
"final_answer": "B"
}
...

Here is the question:
<Question>
QUESTION
</Question>
Here is the correct answer:
<Correct Answer>
GT_ANSWER
</Correct Answer>
Factual knowledge is information that aligns with objective reality and can be verified through evidence or observation, such as scientific facts or historical events. Provide a reasoning planning for a question to get correct answer, each step in your reasoning plan must be adhere strictly to the Json list format:

Additionally, we guide the model to perform data filtering and generate specialized training data for developing a dedicated verifier.

Prompts for Dependency Verification

Here is some information:
"1. The USDA recommends that the total fat intake should be between 20% to 35% of total daily calories. For a 2,000-calorie diet, this translates to about 44-78 grams of fat per day. 2. To find the total fat content for seven McDonald's hamburgers, multiply the fat content of one hamburger by seven. Therefore, 9 grams of fat per hamburger multiplied by 7 hamburgers equals 63 grams of fat."

Based on the given information, here is a reasoning process:
"Compare the total fat content of 63 grams from seven hamburgers to the USDA recommended daily allowance of 44-78 grams. Since 63 grams falls within this range, seven hamburgers do not exceed the USDA recommended fat allowance."

Let's review the reasoning process and check:
1. What's the purpose of the reasoning process?
2. Can we use the given information to deduce this reasoning process and its conclusion?
3. Did this reasoning process achieve its purpose?
4. Do you think the reasoning process is wrong?
Any conflict or mistake fails the check. Please return the review in Json format.

Here is the review:
{
"Purpose": "Check if the total fat content from seven hamburgers is in the USDA recommended daily allowance range",
"DependencySatisfiability": "The premise information presents the recommended total fat intake (between 44 and 78 grams) and the total fat of 7 hamburgers of 63 grams; The reasoning process locates the information correctly and successfully compares 63 with 44 and 78 grams. Therefore, the reasoning process can be done given the available information.",
"PurposeSatisfiability": "The reasoning process does not contain irrelevant steps and achieves its purpose by comparing the fat of hamburgers and the recommended fat intake.",
"FactSatisfiability": "After double-checking my knowledge on the fat of hamburgers and USDA recommended fat intake, we can conclude the reasoning process contains no mistakes.",
"InSummary": "1. Dependency satisfied. 2. Purpose satisfied. 3. Fact satisfied. Based on the check, the reasoning check passed."
"final_answer": "yes"
}

Here is some information:
{DEPENDENCIES}

Based on the given information, here is a reasoning process:
{REASONING}

Let's review the reasoning process and check:
1. What's the purpose of the reasoning process?
2. Can we use the given information to deduce this reasoning process and its conclusion?
3. Did this reasoning process achieve its purpose?
4. Do you think the reasoning process is wrong?
Any conflict or mistake fails the check. Please return the review in Json format.

Here is the review:

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Please refer to the introduction and abstract. We have state the problems that we focus on, the main method, and the contribution we have made in this paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please refer to Section 5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The formulas, such as Eq. 1 and Eq. 2, are well numbered. In addition, we also present the details for data creation and dependency verification in Algorithm 1.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have state the detailed method in Section 3, and one can find the hyper-parameters for fine-tuning and prompts for querying the models in Appendix A.4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Code will be made accessible upon acceptance.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We have state the detailed method in Section 3, and one can find the hyperparameters for fine-tuning in Appendix A.1 and prompts for querying the models in Appendix A.4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We have tested our method on various settings via ablation study in Section 4 and presented the averaged results in the tables and figures in both experiment section and appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We present the details of compute resources in Section 4.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We have discussed the potential impact of the method to improve the reasoning capability of LLMs, which could significantly affect the society. The improvement of reasoning models brought by our GRiD method can significantly benefit the tasks that requires good reasoning ability, such as planning, and complicated QA tasks. Please refer to Section 1 and Section 6.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: In this paper, we consider to improve the reasoning capability of LLMs. This is a framework instead of any specific LLM models or dataset. In addition, the base models that we used are all the original open-source models. One should use them under the guidelines and correct licenses.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All the data and code resources are open accessed or open-sourced on the internet.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The new assets introduced in this paper are provided along with basic documentation. However, some details or usage instructions might need further elaboration or improvement. We plan to enhance the documentation to ensure ease of use and reproducibility in the code upon acceptance.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We do not use Crowdsourcing.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

Justification: The paper does not involve crowdsourcing.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: This paper focuses on improving the reasoning capability of LLMs. We have discussed the usage and fine-tuning in Section 3, Section 4, and Appendix A.1. As for the perspective of paper writing, we only use LLMs for polish the writing of our paper to check if there are any grammar issues.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.