arXiv:2602.18905v2 [cs.LG] 5 Jun 2026

TRUE: A Trustworthy Unified Explanation Framework for Large Language Model Reasoning

Yujiao Yang
Dalian University of Technology
Dalian, China
yjyang@mail.dlut.edu.cn

Abstract

Large language models (LLMs) have demonstrated strong capabilities in complex reasoning tasks, yet their decision-making processes remain difficult to interpret. Existing explanation methods often lack trustworthy structural insight and are limited to single-instance analysis, failing to reveal reasoning stability and systematic failure mechanisms. To address these limitations, we propose the Trustworthy Unified Explanation Framework (TRUE), which integrates executable reasoning verification, feasible-region directed acyclic graph (DAG) modeling, and causal failure mode analysis. At the instance level, we redefine reasoning traces as executable process specifications and introduce blind execution verification to assess operational validity. At the local structural level, we construct feasible-region DAGs via structure-consistent perturbations, enabling explicit characterization of reasoning stability and the executable region in the local input space. At the class level, we introduce a causal failure mode analysis method that identifies recurring structural failure patterns and quantifies their causal influence using Shapley values. Extensive experiments across multiple reasoning benchmarks demonstrate that the proposed framework provides multi-level, verifiable explanations, including executable reasoning structures for individual instances, feasible-region representations for neighboring inputs, and interpretable failure modes with quantified importance at the class level. These results establish a unified and principled paradigm for improving the interpretability and reliability of LLM reasoning systems.

1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities on complex reasoning tasks spanning mathematical problem solving, logical inference, and knowledge-intensive decision-making. Understanding *how* LLMs arrive at their decisions is crucial for revealing reasoning boundaries, building user trust, and enabling reliable deployment Lanham et al. [2023], Paul et al. [2024], Tutek et al. [2025]. However, due to the scale and opacity of modern LLMs, traditional perturbation- and attention-based attribution methods fail to provide structurally meaningful explanations Agarwal et al. [2024]. Prompting-based methods instead elicit step-by-step reasoning traces Cahlik et al. [2025], yet prior work shows these traces often serve as post-hoc rationalizations rather than faithful reflections of the decision process Turpin et al. [2023], Barez et al. [2025]. Moreover, existing local explanation methods treat each instance in isolation, failing to expose structural patterns or systematic failure modes at the problem-class level.

To address these limitations, we propose the **Trustworthy Unified Explanation Framework (TRUE)**, a multi-level framework that elevates explanations from descriptive text to verifiable, executable, and structurally grounded reasoning representations. TRUE consists of three components: (i) *executable explanations*, in which reasoning traces are redefined as process specifications and validated by an independent blind execution verifier that recovers the answer solely from the steps,

Preprint.

establishing a direct link between explanation executability and reasoning correctness; (ii) *local feasible-region DAGs*, constructed via structure-preserving perturbations, which characterize the stability and executability of reasoning across semantically similar inputs; and (iii) *cluster-level failure mode analysis*, which aggregates counterfactual perturbations across semantically related samples and uses Shapley attribution to identify and quantify recurring structural failure modes at the class level.

We conduct extensive experiments on six representative reasoning benchmarks. The results show that executable explanations reliably capture the validity of reasoning structures at the instance level, feasible-region DAGs effectively model structural stability, and cluster-level analysis identifies interpretable and quantifiable systematic deficiencies. Together, these components constitute a unified paradigm for interpretable and verifiable LLM reasoning analysis.

2 Related Works

Perturbation-Based Local Explanation. Perturbation-based methods estimate input importance by applying controlled modifications and observing output changes, forming one of the most widely used model-agnostic paradigms Ribeiro et al. [2016]. For LLMs, tokens, context segments, or prompts are selectively removed or replaced to analyze their influence on reasoning behavior Turpin et al. [2023], Madani et al. [2025], Ortu et al. [2024]. Shapley-value methods provide a theoretically grounded framework by quantifying marginal contributions Lundberg and Lee [2017]; recent extensions adapt them to LLM token- and context-level attribution Horovicz and Goldshmidt [2024], Xiao et al. [2025], Liu et al. [2024], Patel et al. [2025], as well as data attribution Tan et al. [2025], Naudot et al. [2025] for analyzing the influence of training data or in-context examples. Despite their rigor, these methods identify influential components in isolation without modeling the structural organization of reasoning, and therefore cannot verify executability, characterize stability across related inputs, or identify failure modes.

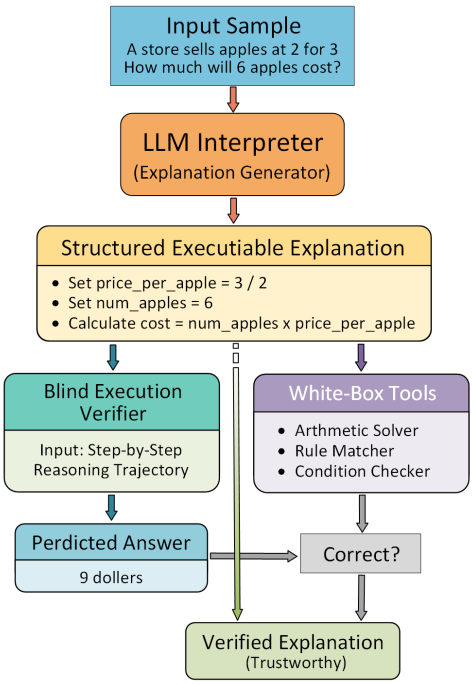

Figure 1: Generation and verification of executable explanations. Given an input instance, the LLM interpreter produces a structured executable explanation consisting of explicit reasoning steps. The explanation is executed by a blind verifier to obtain a predicted answer, while white-box tools independently verify correctness of each reasoning step.

Prompting-Based Explanation. Prompting-based methods elicit intermediate reasoning steps as natural-language traces. Chain-of-Thought (CoT) Wei et al. [2022] and its extensions—Zero-Shot CoT Kojima et al. [2022], Plan-and-Solve Wang et al. [2023a], Self-Refine Madaan et al. [2023]—produce coherent step-by-step reasoning. These approaches make the reasoning process more observable and provide a practical interface for analyzing model behavior. Subsequent studies analyze how such traces reflect model behavior via demonstration editing and contradictory mappings Han et al. [2023], Wei et al. [2023], perturbation/attribution on CoT Wang et al. [2023b], Chuang et al. [2025], and counterfactual prompting Madaan et al. [2022], Zhou et al. [2023]. However, faithfulness concerns persist: generated traces may act as post-hoc rationalizations uncorrelated with the true decision process Turpin et al. [2023], Lanham et al. [2023]. Overall, prompting-based methods treat explanations as descriptive text without verifying whether they constitute complete and executable reasoning procedures, operate only at the single-instance level, and do not model shared reasoning structures or systematic failure modes, which motivates our framework.

3 Methodology

3.1 Executable Explanations for Single-Instance Reasoning

We begin by studying *trustworthy explanations* at the single-instance level. Since CoT-style traces often act as post-hoc rationalizations rather than faithful decision records, we redefine explanations as *executable reasoning specifications* whose validity can be verified by external execution (Figure 1).

Definition 1 (Executable Explanation). Given an input problem x, an executable explanation is defined as a structured sequence of reasoning steps

$$E = (e_1, e_2, \ldots, e_T), \tag{1}$$

where each step e_t specifies an executable operation with its symbolic variables and a brief description. The explanation is a process specification, not a narrative, and must not reveal intermediate numerical values or the final answer. In practice, candidate explanations are generated via fixed prompting strategies (e.g., CoT, Plan-and-Solve) and uniformly treated as executable explanations.

Definition 2 (Trustworthy Explanation). E is *trustworthy* if an independent executor can recover the correct answer without access to the original problem x. We implement this via a *blind execution verifier* V that combines LLM-based step interpretation with white-box components (arithmetic solvers, rule-based matchers) to prevent external knowledge injection: the LLM only interprets the provided steps while white-box modules ensure operational correctness. The verifier receives only E and produces

$$\hat{y} = V(E). \tag{2}$$

If $\hat{y}$ matches the ground truth, the explanation encodes a complete and executable solution structure and is deemed trustworthy.

3.2 Local Feasible-Region Modeling via Structure-Preserving Perturbations

Executable explanations verify single-instance validity but do not characterize stability under nearby input variations. To assess whether the model preserves correct reasoning across semantically similar inputs, we propose a *local feasible-region modeling* framework that characterizes the feasible reasoning structure in the input neighborhood (Figure 2).

Structure-Preserving Perturbation Generation. Given a target instance x, the LLM identifies perturbable factors and modifies them *without changing the overall solution structure*, producing a neighborhood set

$$\mathcal{N}(x) = \{x^{(0)}, x^{(1)}, \ldots, x^{(K)}\}, \tag{3}$$

where $x^{(0)}$ denotes the original instance. Perturbations follow predefined types (parameter variations, entity substitutions, condition adjustments), and labels for perturbed instances are obtained by combining LLM reasoning with white-box tools (e.g., numerical calculators).

Reasoning Behavior Evaluation on Perturbations. For each instance in $\mathcal{N}(x)$, we obtain reasoning steps and predicted answers, then quantify for each step s_i: (i) *semantic correctness* $C_i \in \{0, 1\}$, determined by an LLM semantic verifier $g(s_i, s_i^{\text{ref}})$ against the reference

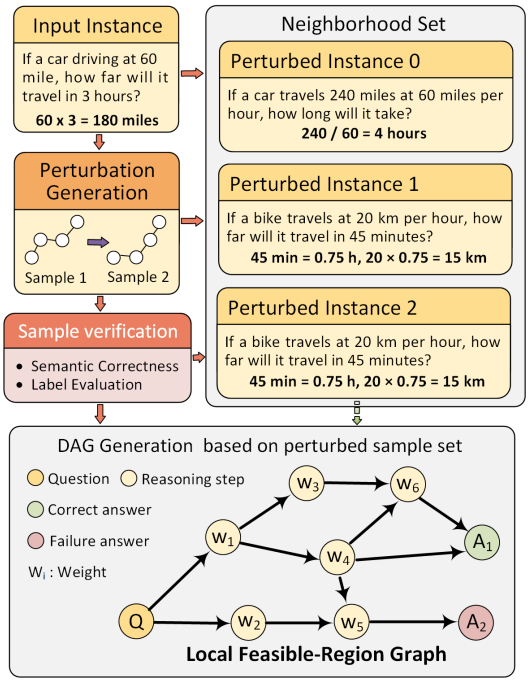

Figure 2: Construction of the local feasible-region graph via structure-preserving perturbations. Verified perturbed samples are aggregated into a directed acyclic graph, where nodes represent reasoning steps and edges encode their dependencies.

step; and (ii) *blind executability* $R_i = n_i^{\text{exec}}/|\mathcal{N}(x)|$, the fraction of perturbed instances for which s_i executes successfully. The step reliability weight is $W_i = C_i \cdot R_i$, jointly capturing semantic correctness and robustness to local perturbations.

Local Feasible-Region Graph Construction. We aggregate all reasoning trajectories observed across $\mathcal{N}(x)$ into a directed acyclic graph $G = (S, E)$, where each node S_i is a merged reasoning step weighted by $W_i \in [0, 1]$ and edges encode observed step dependencies. This graph provides a structured representation of the reasoning paths the LLM may exhibit on nearby inputs, together with their correctness and execution likelihoods, enabling local decision-behavior analysis.

3.3 Cluster-Level Failure Mode Analysis

Local feasible-region modeling captures stability within individual-sample neighborhoods but does not directly reveal class-level failure mechanisms. We therefore introduce a cluster-level failure mode analysis module that automatically discovers recurring reasoning deficiencies within a problem class and quantifies their influence on prediction outcomes (Figure 3).

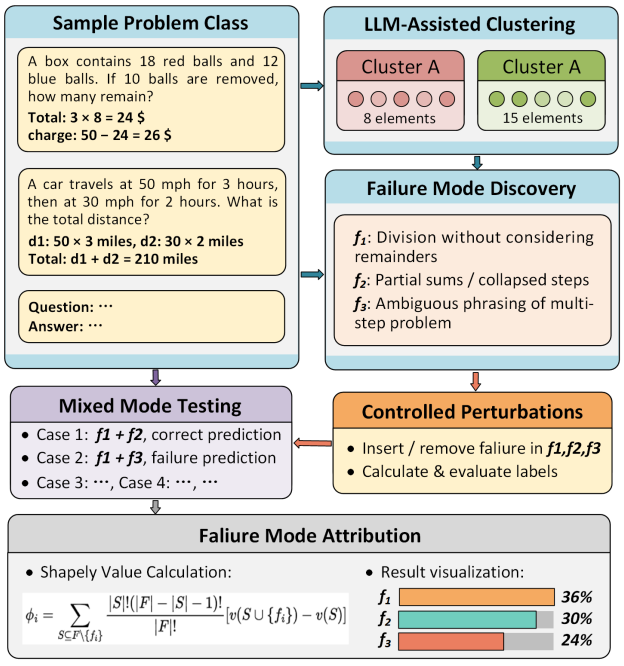

Figure 3: Overview of the proposed failure mode discovery and attribution framework. Given a sample problem class, semantically similar instances are grouped via LLM-assisted clustering, followed by systematic failure mode discovery. Controlled perturbations and mixed-mode testing are then performed to evaluate reasoning behavior under failure combinations. Finally, Shapley value attribution quantifies the causal impact of each failure mode on prediction outcomes.

Failure Mode Discovery. We partition the dataset into small-scale clusters $C = \{x_1, \ldots, x_n\}$ via LLM-assisted inductive summarization, where each cluster contains samples with comparable reasoning structures.

We define a failure mode as a minimal structural condition whose presence increases the likelihood of incorrect prediction, represented as a binary indicator $f_i : (x, E) \to \{0, 1\}$. Each mode must be identifiable from the reasoning trace or input structure, manipulable via controlled perturbations, and locally minimal in influence. Failure modes are automatically extracted by an LLM-driven procedure: for incorrectly predicted samples, generated traces are aligned with reference solutions to identify divergent steps, which the LLM then abstracts into candidate failure descriptions (affected components, input fragments, hypothesized cause). Candidates are normalized and merged via semantic clustering into a compact failure mode set $F = \{f_1, \ldots, f_K\}$.

Failure Importance Estimation. To quantify each mode's influence, we construct counterfactuals by injecting selected failure modes into correctly solved samples and removing/simplifying them in incorrectly solved ones; ground-truth labels are recomputed when problem parameters change. The perturbed samples form an augmented cluster C' on which we measure prediction correctness under different failure configurations. We estimate the contribution of each mode via Shapley attribution:

$$\phi_i = \sum_{S \subseteq F \setminus \{f_i\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} \left[v(S \cup \{f_i\}) - v(S) \right] \tag{4}$$

Table 1: Executable Explanation Evaluation across six benchmarks (%). Base model: GPT-4o-mini, N=200 per dataset.

	MATH				MMLU				GSM8K			
Method	EA	OA	EC	ERR	EA	OA	EC	ERR	EA	OA	EC	ERR
CoT	73.0	76.0	96.1	0.0	69.5	72.5	95.9	0.0	92.0	93.0	97.8	14.3
Self-Refine	72.5	75.0	96.7	0.0	63.0	66.0	91.7	7.4	90.5	92.0	98.4	0.0
Zero-Shot-CoT	74.5	78.5	93.6	4.7	59.0	69.0	83.3	4.8	93.5	95.0	97.9	10.0
Plan-and-Solve	70.0	73.0	95.2	1.9	56.0	72.5	72.4	12.7	92.5	93.0	98.4	14.3
Avg.	72.5	75.6	95.4	1.7	61.9	70.0	85.8	6.2	92.1	93.2	98.1	9.6
	BBH				**CRUXEval**				**HotpotQA**			
Method	EA	OA	EC	ERR	EA	OA	EC	ERR	EA	OA	EC	ERR
CoT	68.5	71.5	95.8	0.0	46.5	48.0	96.9	0.0	14.5	32.0	42.2	1.5
Self-Refine	46.5	60.5	69.4	11.4	39.0	42.0	89.3	2.6	5.0	15.0	30.0	0.6
Zero-Shot-CoT	46.0	68.5	62.0	11.1	43.5	43.0	97.7	2.6	11.0	25.0	40.0	1.3
Plan-and-Solve	36.0	70.0	45.7	13.3	45.0	46.5	95.7	0.9	11.5	24.0	41.7	2.0
Avg.	49.3	67.6	68.2	9.0	43.5	44.9	94.9	1.5	10.5	24.0	38.5	1.4

where $v(S)$ denotes prediction correctness under configuration S. The resulting ϕ_i values yield a cluster-level failure profile that highlights dominant error sources and provides an interpretable characterization of reasoning robustness within the problem class.

4 Experiments

4.1 Executable Explanation Reliability Evaluation

Setup. We evaluate on six benchmarks (N=200 per dataset, fixed random seed): **MATH** Hendrycks et al. [2021a], **GSM8K** Cobbe et al. [2021], **BBH** Suzgun et al. [2023], **MMLU** Hendrycks et al. [2021b], **CRUXEval** Gu et al. [2024], and **HotpotQA** Yang et al. [2018], spanning mathematical, logical, knowledge, code, and open-domain reasoning. We vary (i) prompting strategy (CoT, Zero-Shot-CoT, Self-Refine, Plan-and-Solve) with GPT-4o-mini fixed, and (ii) base model (7 APIs from 4 providers) with CoT fixed. GPT-4o-mini serves as the blind executor throughout to decouple execution quality from model capability.

Metrics. Let N_{exec}, N_{orig}, N_{joint}, N_{rec} denote counts of samples correct under blind execution, original method, both, and recovery-only, respectively. We report:

$$EA = \frac{N_{\text{exec}}}{N}, \quad OA = \frac{N_{\text{orig}}}{N}, \quad EC = \frac{N_{\text{joint}}}{N_{\text{orig}}}, \quad ERR = \frac{N_{\text{rec}}}{N - N_{\text{orig}}}. \tag{5}$$

Execution Accuracy (EA) measures explanation self-sufficiency; Original Accuracy (OA) is the standard task accuracy of the prompting method; Execution Consistency (EC) quantifies what fraction of correct predictions remain verifiable under blind execution; Execution Recovery Rate (ERR) measures recovery from originally wrong predictions.

Results. Table 1 reveals a clear executability hierarchy across task types. On structured tasks, EA closely tracks OA: GSM8K achieves avg. EC = 98.1% and MATH avg. EC = 95.4%, confirming that arithmetic reasoning explanations are highly self-contained. CRUXEval shows a distinctive pattern: despite moderate OA (44.9%), EC remains 94.9%, reflecting the inherently procedural nature of code traces. In contrast, knowledge-intensive tasks show larger EA–OA gaps; HotpotQA is the most extreme (avg. EA = 10.5% vs. OA = 24.0%, EC = 38.5%), as multi-hop QA requires external factual retrieval that cannot be encoded in reasoning steps alone. ERR is near zero across most datasets, indicating blind execution rarely corrects originally wrong predictions. These results establish a three-tier hierarchy: mathematical/code (EC $>$ 94%), knowledge-intensive (EC $\approx$ 68–86%), and open-domain QA (EC $\approx$ 39%).

Table 2: Executable Explanation Evaluation across seven model APIs on four benchmarks (CoT, N=200, verifier: GPT-4o-mini). Models grouped by provider; horizontal rules separate provider families. Upper block: MATH and MMLU; lower block: BBH and CRUXEval.

		MATH				**MMLU**			
Provider	**Model**	EA	OA	EC	ERR	EA	OA	EC	ERR
OpenAI	GPT-4o-mini	71.5	74.0	95.9	1.9	70.0	72.0	95.8	3.6
	GPT-4.1-nano	65.5	70.5	90.1	6.8	66.0	69.0	91.3	9.7
	GPT-5-nano	93.0	97.0	95.9	0.0	73.5	75.0	98.0	0.0
Google	Gemini-2.0-Flash	46.0	48.0	95.8	0.0	65.0	74.5	81.2	17.6
Alibaba	Qwen3.5-35B	98.0	98.5	99.0	33.3	80.5	81.5	98.8	0.0
	Qwen3.5-Plus	98.0	98.5	99.5	0.0	83.0	84.5	98.2	0.0
DeepSeek	DeepSeek-Chat	88.5	92.0	96.2	0.0	78.0	78.5	98.1	4.7
		BBH				**CRUXEval**			
Provider	**Model**	EA	OA	EC	ERR	EA	OA	EC	ERR
OpenAI	GPT-4o-mini	68.0	70.5	96.5	0.0	46.5	47.5	97.9	0.0
	GPT-4.1-nano	63.5	67.5	82.2	24.6	46.5	49.5	93.9	0.0
	GPT-5-nano	77.5	80.5	95.7	2.6	52.5	53.5	98.1	0.0
Google	Gemini-2.0-Flash	63.5	60.5	91.7	20.3	48.0	49.5	94.9	2.0
Alibaba	Qwen3.5-35B	88.5	88.5	100.0	0.0	53.5	53.5	100.0	0.0
	Qwen3.5-Plus	86.5	89.5	96.6	0.0	52.5	53.0	99.0	0.0
DeepSeek	DeepSeek-Chat	79.5	80.0	98.8	2.5	53.0	53.0	100.0	0.0

Table 2 shows that executable explanation quality consistently correlates with model capability across all four provider families. Stronger models yield higher EA: GPT-5-nano reaches 93.0% on MATH and 77.5% on BBH, while Qwen3.5-35B and Qwen3.5-Plus both attain 98.0% on MATH. EC remains above 90% for most combinations; the main exceptions—GPT-4.1-nano on BBH (82.2%, ERR = 24.6%) and Gemini-2.0-Flash on MMLU (81.2%, ERR = 17.6%)—suggest that weaker models occasionally produce structurally recoverable explanations despite incorrect original predictions. On CRUXEval, all models achieve EC $>$ 93.9% regardless of capability level, confirming that code-execution reasoning produces inherently self-contained explanations. These findings validate the generality of the TRUE framework across diverse model families.

4.2 Structural Validity Evaluation of Local Feasible Regions

Setup. We experiment on MATH and MMLU-CF, constructing five clusters per dataset (10–20 instances each) via LLM-assisted grouping. For each cluster, an anchor instance is perturbed under three regimes (mild/moderate/aggressive) to build a feasible-region DAG; reference solutions for perturbed instances are verified with external tools. We assess two properties: (i) *success-rate prediction*: an LLM is given the DAG plus a CoT trace and asked to predict execution success; cross-entropy (CE) is compared against a same-budget baseline that uses anchor-only CoT sampling instead of perturbations; (ii) *trajectory coverage*: reasoning steps are semantically aligned to DAG nodes and the fraction of matched steps (Pret./GT) is reported across perturbation regimes.

Results. The DAG-based predictor substantially outperforms the anchor-only baseline on both datasets (Table 3). On MATH, average CE drops from 4.63 to 0.85 (-81.6%), with the largest gains in Prealgebra (ΔCE = +10.46) and Counting & Probability (+6.97). On MMLU-CF, average CE falls from 4.90 to 0.97, with marked improvements in Economics and Health domains. Pert. SR closely tracks Test SR across clusters, confirming that the perturbations faithfully model the local data distribution. Trajectory coverage (Table 4) increases monotonically with perturbation strength; under aggressive perturbations, MATH reaches 96.4% for both Pret. and GT, while MMLU-CF achieves 66.6% and 61.3%. Together these results confirm that the feasible-region DAG captures sufficient structural information to characterize cluster-level reasoning behavior.

Table 3: Success Rate Prediction Results on MATH and MMLU-CF. Lower CE indicates better calibration. ΔCE = Baseline CE $-$ DAG CE.

Category	Pert. SR	Test SR	DAG CE↓	Baseline CE↓	ΔCE
MATH					
Prealgebra	50.0%	87.5%	**1.05**	11.51	+10.46
Number Theory	83.3%	75.0%	**0.75**	1.39	+0.64
Intermediate Alg.	33.3%	50.0%	**0.30**	0.75	+0.45
Counting & Prob.	50.0%	75.0%	**1.72**	8.69	+6.97
Precalculus	16.7%	37.5%	**0.43**	0.79	+0.36
Average	46.7%	65.0%	**0.85**	4.63	+3.78
MMLU-CF					
Classification and Rules	83.3%	65.0%	1.00	**0.83**	-0.17
Health and Medicine	100.0%	80.0%	**0.53**	4.51	+3.98
History and Politics	66.6%	75.0%	**0.67**	4.51	+3.84
Science and Technology	83.3%	65.0%	0.98	**0.83**	-0.15
Economics and Finance	100.0%	40.0%	**1.69**	13.82	+12.13
Average	86.6%	65.0%	**0.97**	4.90	+3.93

Table 4: DAG trajectory coverage results on MATH and MMLU-CF datasets. Nodes and Edges denote the size of the feasible-region DAG. Pret.(%) represents the average trajectory coverage rate of perturbed instances, while GT(%) denotes the coverage rate of ground-truth reasoning trajectories.

	MATH Dataset				MMLU-CF Dataset			
Perturbation	**Nodes**	**Edges**	**Pret.(%)**	**GT(%)**	**Nodes**	**Edges**	**Pret.(%)**	**GT(%)**
Mild	9	35	51.9	34.8	18	29	50.8	48.3
Moderate	10	35	74.2	62.9	23	30	58.8	53.1
Aggressive	13	31	96.4	96.4	26	32	66.6	61.3

4.3 Cluster-Level Failure Mode Evaluation

Setup. We evaluate on MATH, BBH (multiple-choice subset), and CRUXEval, constructing two representative clusters per dataset (six total) via GPT-4o-mini-assisted inductive summarization. For each cluster, five high-impact failure modes are selected and counterfactual variants are generated by (a) injecting failure conditions into correctly predicted instances and (b) removing them from incorrect ones; ground-truth labels are recomputed where needed. Shapley values ϕ_i are estimated from the characteristic function $v(S)$ defined by prediction correctness under each failure configuration (Eq. 4). Stability is assessed by repeating the full pipeline on subsampled clusters of size $n \in \{5, 10, 20, 40\}$, evaluating mode-set overlap (Jaccard) and ranking consistency (Kendall's τ).

Results. Figure 4 shows that reasoning failures are not uniformly distributed but concentrated in domain-specific structural patterns. On MATH, procedural errors dominate: Area Calculation (ϕ=0.37) and Application of Formulas (ϕ=0.32) are the top modes in Prealgebra and Algebra respectively. On BBH, constraint-management failures lead—Overlooking Constraints (ϕ=0.25) and Logical Consistency Errors (ϕ=0.22) in Logical Reasoning, while Spatial-Temporal shows a more diffuse distribution (max ϕ=0.18). On CRUXEval, the Simple Functions cluster exhibits three equal-weight low-level errors (ϕ=0.13 each), while Loop & Recursion is dominated by Unconditional Update (ϕ=0.27), reflecting difficulty in tracking mutable state. These domain-specific profiles confirm the Shapley framework's ability to pinpoint fine-grained structural weaknesses.

Table 5 reports failure mode discovery stability measured by Jaccard similarity and Kendall's τ across six clusters from three datasets as a function of subsample size.

Across all clusters, Jaccard similarity is consistently moderate to high (≥ 0.59), indicating that the set of discovered failure modes is robust even with small sample sizes. CRUXEval clusters exhibit the highest mode overlap stability (Jaccard ≥ 0.67 at all sample sizes), likely because code execution

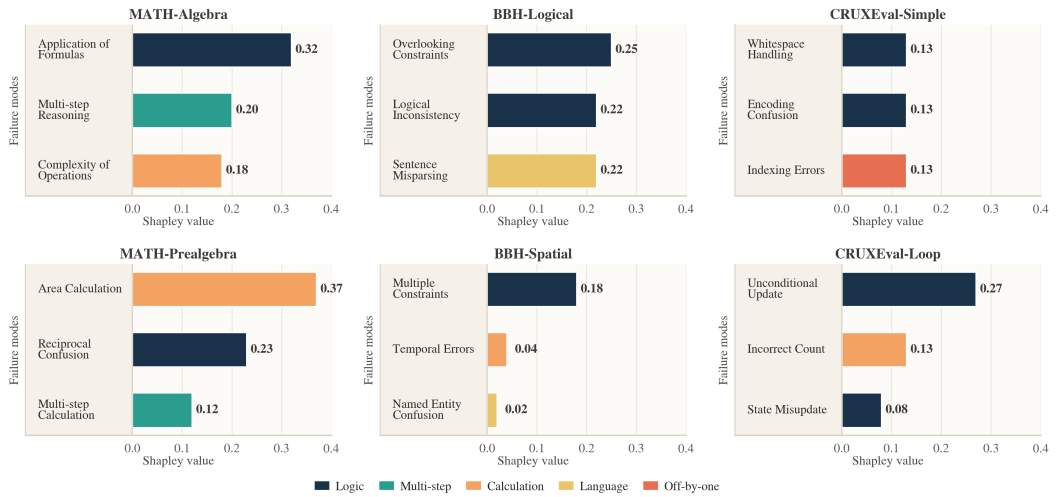

Figure 4: Cluster-level failure mode profiles on MATH, BBH, and CRUXEval. Top-3 failure modes per cluster ranked by Shapley value ϕ. Reasoning failures concentrate in domain-specific structural patterns rather than distributing uniformly.

Table 5: Failure mode discovery stability under subsampling (mean $\pm$ std over 3 trials). Jac: Jaccard; τ: Kendall's τ.

	MATH-Algebra		**MATH-Prealgebra**		**BBH-Logical**		**BBH-Spatial**		**CRUX-Simple**		**CRUX-Loop**	
n	Jac	τ	Jac	τ	Jac	τ	Jac	τ	Jac	τ	Jac	τ
5	$.59_{\pm.11}$	$.22_{\pm.88}$	$.70_{\pm.23}$	$.22_{\pm.57}$	$.59_{\pm.11}$	$.67_{\pm.27}$	$.89_{\pm.16}$	$.49_{\pm.13}$	$.89_{\pm.16}$	$.18_{\pm.14}$	$.67_{\pm.00}$	$.00_{\pm.27}$
10	$.70_{\pm.23}$	$.71_{\pm.28}$	$.59_{\pm.11}$	$.56_{\pm.16}$	$.70_{\pm.23}$	$.51_{\pm.35}$	$.89_{\pm.16}$	$.49_{\pm.13}$	$.89_{\pm.16}$	$.13_{\pm.34}$	$.78_{\pm.16}$	$.07_{\pm.29}$
20	$.67_{\pm.00}$	$.67_{\pm.27}$	$.67_{\pm.00}$	$.67_{\pm.00}$	$.59_{\pm.11}$	$.67_{\pm.27}$	$.67_{\pm.00}$	$.56_{\pm.16}$	$.89_{\pm.16}$	$.44_{\pm.26}$	$1.00_{\pm.00}$	$.47_{\pm.47}$
40	$.59_{\pm.11}$	$.44_{\pm.42}$	$.78_{\pm.16}$	$.44_{\pm.31}$	$.67_{\pm.00}$	$.33_{\pm.00}$	$.78_{\pm.16}$	$.36_{\pm.03}$	$.89_{\pm.16}$	$.33_{\pm.25}$	$.89_{\pm.16}$	$.38_{\pm.33}$

errors fall into well-defined structural categories. The CRUXEval Loop & Recursion cluster achieves perfect mode overlap (Jaccard = 1.00) at n=20.

Ranking stability (Kendall's τ) shows more variation. MATH clusters achieve moderate ranking consistency ($\tau \approx 0.44$–0.71 for $n \geq 10$), while BBH Spatial-Temporal maintains stable rankings across all sizes ($\tau \approx 0.36$–0.56). CRUXEval clusters show lower stability at small sample sizes ($\tau \approx 0.00$–0.18 at n=5), but improve with larger samples ($\tau \approx 0.33$–0.47 at $n \geq 20$).

These results demonstrate that the proposed failure mode discovery framework achieves reliable mode identification even with limited samples (n=10–20), confirming its practical applicability for cluster-level reasoning analysis.

4.4 Ablation Study and Error Analysis

Impact of Prompting Strategy on Faithfulness. We evaluate four widely-used prompting strategies—CoT [Wei et al., 2022], Zero-Shot-CoT [Kojima et al., 2022], Self-Refine [Madaan et al., 2023], and Plan-and-Solve [Wang et al., 2023a]—to assess how prompting complexity affects reasoning faithfulness as measured by the Hidden Failure Rate (HFR $= 1 -$ EC). Table 6 reports OA (%) and HFR (%) for each method on all six benchmarks.

Two key findings emerge. First, *more elaborate prompting does not guarantee more faithful reasoning.* Plan-and-Solve achieves similar OA to CoT on BBH (70.0% vs. 71.5%) but incurs a dramatically higher HFR (54.3% vs. 4.2%): over half of its apparently correct solutions fail blind execution verification. Similarly, Zero-Shot-CoT on MMLU reaches 69.0% OA while producing 16.7% hidden failures. Second, *faithfulness degrades sharply with task complexity.* On structured-output tasks (MATH, GSM8K), all methods achieve HFR $\leq 6.4\%$, while on open-ended multi-hop QA (HotpotQA) HFR exceeds 57% for every method—indicating that the model's stated reasoning rarely fully encodes its answer derivation for complex QA.

Table 6: Ablation over prompting strategies: Original Accuracy (OA%) and Hidden Failure Rate (HFR% $= 1-$EC). Lower HFR indicates more faithful reasoning. Best HFR per dataset in **bold**.

	MATH		GSM8K		MMLU		BBH		CRUXEval		HotpotQA	
Method	OA	HFR	OA	HFR	OA	HFR	OA	HFR	OA	HFR	OA	HFR
CoT	76.1	**4.0**	93.0	2.2	72.5	**4.1**	71.5	**4.2**	48.0	3.1	32.0	57.8
Zero-Shot-CoT	78.5	6.4	95.0	2.1	69.0	16.7	68.5	38.0	43.0	**2.3**	25.0	60.0
Self-Refine	75.2	3.3	92.0	**1.6**	66.0	8.3	60.5	30.6	42.0	10.7	15.0	70.0
Plan-and-Solve	73.0	4.8	93.0	**1.6**	72.5	27.6	70.0	54.3	46.5	4.3	24.0	**58.3**

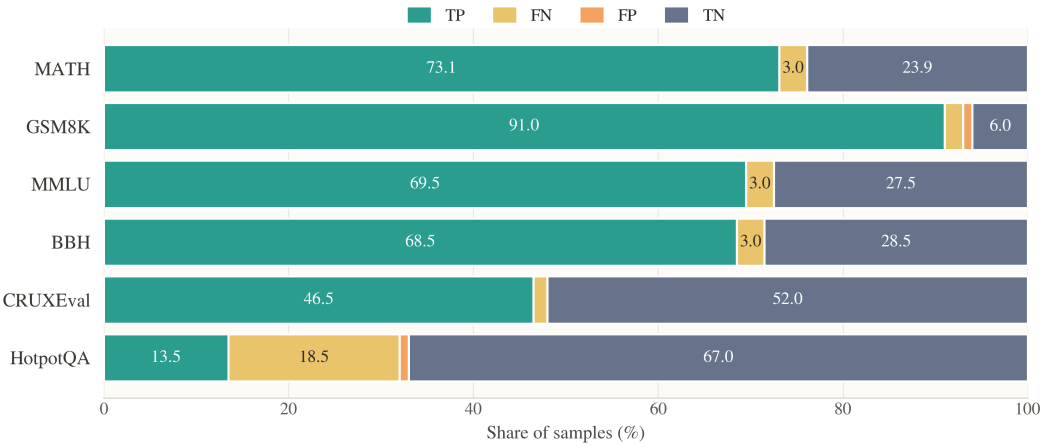

Figure 5: Error breakdown (% of n=200 samples) for CoT. FP $\approx$ 0 across all tasks; FN (hidden failures) dominates on HotpotQA.

Error Breakdown Analysis. Figure 5 details the four-way error decomposition for CoT across all datasets. *True Positive (TP)*: both CoT answer and blind-executed answer are correct; *False Negative (FN)*: CoT answer correct but execution fails (hidden failure); *False Positive (FP)*: execution succeeds but CoT answer is wrong (rare, indicates lucky execution); *True Negative (TN)*: both fail.

The FP rate is near zero across all benchmarks, confirming that blind execution does not fabricate correct answers and the verifier is conservative. The dominant failure mode is FN (hidden failures), which is negligible on arithmetic tasks (MATH: 3.0%, GSM8K: 2.0%) but substantial on HotpotQA (18.5%). The high TN rate on CRUXEval (52.0%) and HotpotQA (67.0%) reflects the inherent difficulty of those tasks rather than a failure of the TRUE framework itself. Together these results confirm the central claim of TRUE: OA systematically overestimates reasoning reliability, and EC (or equivalently HFR) provides an executable audit of faithfulness that OA cannot capture.

5 Conclusion

We propose TRUE, a unified framework for analyzing LLM reasoning through executable verification and perturbation-driven structural analysis. Its three components—executable explanations with blind-execution verification, perturbation-guided feasible-region DAGs, and Shapley-based cluster-level failure mode analysis—jointly characterize reasoning behavior at instance, local, and class levels. Experiments on multiple reasoning benchmarks confirm that executable explanations reliably encode solution procedures, feasible-region DAGs improve execution success prediction, and cluster-level analysis identifies stable, interpretable failure modes even under limited samples. Together, these results establish a principled paradigm for interpretable and verifiable LLM reasoning analysis.

Limitations. Two limitations warrant attention. First, the blind execution verifier V relies on an LLM for step interpretation, bounding its reliability to the quality of the underlying model. Second, exact Shapley attribution over all 2^K subsets limits the framework to small K (we use K=5); efficient Shapley approximation algorithms would be needed to scale to larger mode sets.

References

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, et al. Measuring faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2307.13702*, 2023.

Debjit Paul, Robert West, Antoine Bosselut, and Boi Faltings. Making reasoning matter: Measuring and improving faithfulness of chain-of-thought reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 15012–15032, 2024.

Martin Tutek, Fateme Hashemi Chaleshtori, Ana Marasović, and Yonatan Belinkov. Measuring chain of thought faithfulness by unlearning reasoning steps. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 9935–9960, 2025.

Chirag Agarwal, Sree Harsha Tanneru, and Himabindu Lakkaraju. Faithfulness vs. plausibility: On the (un) reliability of explanations from large language models. *arXiv preprint arXiv:2402.04614*, 2024.

Vojtech Cahlik, Rodrigo Alves, and Pavel Kordik. Reasoning-grounded natural language explanations for language models. In *World Conference on Explainable Artificial Intelligence*, pages 3–18. Springer, 2025.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems*, 36:74952–74965, 2023.

Fazl Barez, Tung-Yu Wu, Iván Arcuschin, Michael Lan, Vincent Wang, Noah Siegel, Nicolas Collignon, Clement Neo, Isabelle Lee, Alasdair Paren, et al. Chain-of-thought is not explainability. In *ICLR 2025 Workshop on Foundation Models in the Wild*, 2025.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

Mohammad Reza Ghasemi Madani, Aryo Pradipta Gema, Gabriele Sarti, Yu Zhao, Pasquale Minervini, and Andrea Passerini. Noiser: Bounded input perturbations for attributing large language models. *arXiv preprint arXiv:2504.02911*, 2025.

Francesco Ortu, Zhijing Jin, Diego Doimo, Mrinmaya Sachan, Alberto Cazzaniga, and Bernhard Schölkopf. Competition of mechanisms: Tracing how language models handle facts and counterfactuals. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8420–8436, 2024.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

Miriam Horovicz and Roni Goldshmidt. Tokenshap: Interpreting large language models with monte carlo shapley value estimation. In *Proceedings of the 1st Workshop on NLP for Science (NLP4Science)*, pages 1–8, 2024.

Yingtai Xiao, Yuqing Zhu, Sirat Samyoun, Wanrong Zhang, Jiachen T Wang, and Jian Du. Tokenshapley: Token level context attribution with shapley value. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 3882–3894, 2025.

Fengyuan Liu, Nikhil Kandpal, and Colin Raffel. Attribot: A bag of tricks for efficiently approximating leave-one-out context attribution. *arXiv preprint arXiv:2411.15102*, 2024.

Sara Patel, Mingxun Zhou, and Giulia Fanti. Maxshapley: Towards incentive-compatible generative search with fair context attribution. *arXiv preprint arXiv:2512.05958*, 2025.

Haoru Tan, Sitong Wu, Xiuzhe Wu, Wang Wang, Bo Zhao, Zeke Xie, Gui-Song Xia, and Xiaojuan Qi. Understanding data influence with differential approximation. *arXiv preprint arXiv:2508.14648*, 2025.

Filip Naudot, Tobias Sundqvist, and Timotheus Kampik. llmshap: A principled approach to llm explainability. *arXiv preprint arXiv:2511.01311*, 2025.

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

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35: 22199–22213, 2022.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 2609–2634, 2023a.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594, 2023.

Xiaochuang Han, Daniel Simig, Todor Mihaylov, Yulia Tsvetkov, Asli Celikyilmaz, and Tianlu Wang. Understanding in-context learning via supportive pretraining data. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12660–12673, 2023.

Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, et al. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*, 2023.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023b.

Yung-Sung Chuang, Benjamin Cohen-Wang, Shannon Zejiang Shen, Zhaofeng Wu, Hu Xu, Xi Victoria Lin, James Glass, Shang-Wen Li, and Wen-tau Yih. Selfcite: Self-supervised alignment for context attribution in large language models. *arXiv preprint arXiv:2502.09604*, 2025.

Aman Madaan, Shuyan Zhou, Uri Alon, Yiming Yang, and Graham Neubig. Language models of code are few-shot commonsense learners. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1384–1403, 2022.

Wenxuan Zhou, Sheng Zhang, Hoifung Poon, and Muhao Chen. Context-faithful prompting for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14544–14556, 2023.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021a.

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

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, 2023.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021b.

Alex Gu, Baptiste Rozière, Hugh Leather, Armando Solar-Lezama, Gabriel Synnaeve, and Sida I. Wang. CRUXEval: A benchmark for code reasoning, understanding and execution. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2369–2380, 2018.