Mining Logic under Uncertainty: Probabilistic Soft Logic with Energy-Based Inference for Chain-of-Thought Verification

Jiang Yu[1]* **Jinlong Tian**[1]* **Kewei Cheng**[2] **Yue He**[3] **Haoxuan Li**[4]
Haotian Wang[1] **Yunhai Wang**[3] **Wenjing Yang**[1] **Zhouchen Lin**[4] **Shixuan Liu**[1]†

[1]College of Computer Science and Technology, National University of Defense Technology
[2]Microsoft
[3]Renmin University of China
[4]Peking University
yujiang12321@gmail.com szftandy@hotmail.com

Abstract

Chain-of-Thought reasoning improves the explainability of large language models, yet verifying such reasoning remains difficult when intermediate steps express epistemic uncertainty rather than deterministic entailment. Neuro-symbolic verifiers typically translate reasoning steps into rigid first-order or SMT-style constraints. This forces each proposition into a binary truth regime and creates a *Rigidity–Ambiguity Mismatch*: uncertainty-marked claims such as *may*, *likely*, or *suggests* are either over-hardened into deterministic assertions or rejected as unverifiable. We propose PROBVERI, a probabilistic soft-logic framework for formal verification under uncertainty. The key idea is to represent uncertainty-marked reasoning steps as weighted soft logical constraints over continuous truth assignments, so that partial evidential support remains expressible. Building on this soft-logic representation, we formulate verification as a contrastive energy minimization problem: a step is considered supported when its natural interpretation has lower verification energy than its negated counterpart under the same evidence, soft rules, and assumption regularization. Probabilistic Soft Logic defines the logical energy through weighted hinge-loss constraints. We introduce a support-grounded assumption penalty that discourages unsupported latent bridge assumptions between context and claim. The resulting energy gap provides a confidence-scored verification signal that separates unsupported steps from weakly but consistently supported inferences. Experiments on ProofWriter, BioASQ, and LegalBench-SARA show that PROBVERI recovers more verified-correct reasoning traces than rigid verifiers while reducing over-acceptance compared with soft-only PSL. These results suggest that uncertainty-aware verification benefits from soft logical formalization and contrastive comparison between natural and negated interpretations.

1 Introduction

Large language models increasingly rely on Chain-of-Thought reasoning to make intermediate reasoning steps visible [1, 2, 3, 4, 5]. Recent work further improves reasoning by exploring, refining, or optimizing reasoning paths [6, 7, 8]. This transparency is useful, but it does not in itself guarantee

*Equal contribution.
†Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

reliability. A generated reasoning chain may appear coherent while containing unsupported claims, implicit assumptions, or locally invalid transitions. In high-stakes domains such as medicine, law, and scientific reasoning, the central challenge is therefore not only to generate a plausible reasoning trace, but to verify whether each intermediate step is justified by the available context.

A natural response is to combine language models with external verifiers. Existing neuro-symbolic approaches extract logical structure from generated reasoning and check it using formal solvers, theorem provers, executable programs, or satisfiability engines [9, 10, 11, 12, 13, 14]. These methods provide strong guarantees when a reasoning step can be faithfully translated into deterministic symbolic form. In such cases, verification can be reduced to checking whether the premises entail the claim, contradict the claim, or fail to translate into the solver language. However, this verification regime implicitly assumes that the proposition being checked has a binary semantic status.

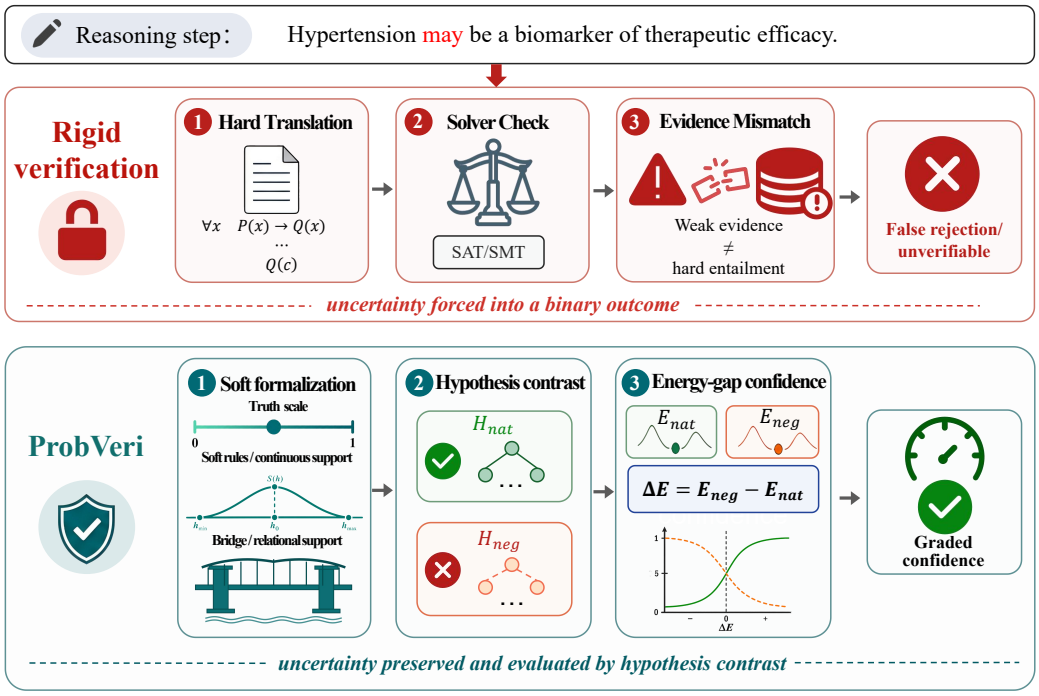


Figure 1: **From rigid verification to uncertainty-aware contrastive verification.** Rigid verification translates an uncertainty-marked step into hard symbolic representation and applies a binary solver check. This can mistake weak but valid evidential support for an evidence mismatch, producing either an unverifiable outcome ($\perp$) or a false rejection. In contrast, PROBVERI preserves uncertainty through soft formalization, compares the natural and negated interpretations under shared evidence, and uses the energy gap $\Delta E = E_{\text{neg}} - E_{\text{nat}}$ to produce a confidence-scored verification signal.

This assumption breaks down for many natural-language CoT steps. Reasoning traces often contain epistemic markers such as *may*, *likely*, *suggests*, *could indicate*, or *is associated with*. These expressions are not stylistic noise; they indicate that the step expresses weak or partial evidential support rather than deterministic entailment. For example, the statement "Hypertension may be a biomarker of therapeutic efficacy" does not assert the hard fact $\texttt{IsBiomarker}(\texttt{HTN}, \texttt{Efficacy})$. It instead states that the available evidence may support such a relation. A rigid verifier has only unsatisfactory options: it can harden the statement into a deterministic predicate, thereby changing its meaning, or it can reject the step as untranslatable. In the latter case, the solver effectively returns an unverifiable outcome $\perp$, which may appear as a false rejection even when the original step is plausibly but weakly supported.

We call this failure mode the *Rigidity–Ambiguity Mismatch*: rigid verifiers require binary truth assignments. However, many reasoning steps in natural language are not that definite. When a step uses words such as *may*, *likely*, or *suggests*, it usually expresses partial evidence rather than a hard claim. The key insight is that such ambiguity should not be treated as a translation error. Instead,

the verifier should keep the uncertainty and reason about it directly. An uncertainty-marked step should first be represented in a soft logical form that preserves partial support. The verifier then asks a contrastive question: is the original interpretation better supported by the evidence than its negation? This shifts verification from binary proof checking to a comparison of which interpretation is more compatible with the available evidence and constraints.

Building on this insight, we propose PROBVERI, a probabilistic soft-logic framework for verifying reasoning under uncertainty. Deterministic steps are still handled by a rigid symbolic path, but uncertainty-marked or weakly grounded steps are handled by a probabilistic soft-logic path. In this path, Probabilistic Soft Logic (PSL) represents reasoning steps as weighted soft constraints over continuous truth assignments [15, 16]. Importantly, the weight of each rule is determined by how well the surrounding context supports it, rather than by the LLM's own opaque confidence. To prevent the soft path from validating claims by inventing unsupported missing premises, PROBVERI adds a support-grounded assumption penalty over latent bridge assumptions. Verification is then performed by comparing the optimized energy of the step's natural interpretation with the optimized energy of its negated interpretation. The difference between these two energies gives a graded confidence score for how well the original step is supported.

Figure 1 summarizes this contrast: rigid verification forces uncertain support into binary checking, whereas PROBVERI preserves uncertainty and evaluates the natural interpretation against its negation through an energy gap. Empirically, this formulation addresses two opposite failure modes. Rigid symbolic verification is precise but often rejects uncertainty-marked steps because they cannot be faithfully translated into hard entailment. Soft-only verification covers more cases, but it can also over-accept weakly supported claims. Across logical, biomedical, and legal reasoning benchmarks, PROBVERI verifies more correct reasoning traces than rigid verifiers, while reducing over-acceptance compared with soft-only PSL, suggesting that uncertainty-aware verification requires both soft logical formalization to preserve graded support, and natural-versus-negated contrast to prevent weak claims from being accepted too easily.

Our contributions are threefold:

- We identify the *Rigidity–Ambiguity Mismatch* in verifying uncertainty-marked CoT reasoning, where weak evidential support is hardened into binary entailment or rejected as unverifiable.
- We introduce a probabilistic soft-logic formulation that represents uncertain steps as weighted soft constraints and verifies them through natural-versus-negated energy minimization.
- We evaluate PROBVERI across logical, biomedical, and legal reasoning benchmarks with multiple reasoning backbones. The results show that PROBVERI accepts more correct reasoning traces than rigid verifiers, but does not become as overly permissive as soft-only verification.

2 Problem Formulation

A reasoning trace is a sequence of natural-language steps, where each step is a local inference toward the final answer. For example,

s_1 : "The patient has hypertension."

s_2 : "Hypertension may indicate treatment efficacy."

s_3 : "Therefore, the treatment is likely effective."

Here, s_1 states a relatively determinate fact, while s_2 and s_3 express uncertain inferences marked by words such as *may* and *likely*.

Formally, let Q be the query and C be the available context, such as retrieved documents, premises, or case facts. A language model produces a reasoning trace $S = \{s_1, \ldots, s_T\}$. At step t, the verifier observes the accumulated history $\mathcal{H}_{t-1} = C \cup \{s_1, \ldots, s_{t-1}\}$ and decides whether the candidate step s_t is supported by this history. For example, when verifying s_2, the verifier uses C and s_1 to judge whether the evidence supports the uncertain claim that hypertension may indicate treatment efficacy.

Standard symbolic verification treats this as hard entailment. It translates s_t into a logical proposition q_t, and checks whether q_t follows from, contradicts, or cannot be verified from $\mathcal{H}_{t-1}$ [9, 10, 14, 17, 18]. We write the hard verification outcome as

$$\mathcal{V}_{\mathrm{hard}}(s_t, \mathcal{H}_{t-1}) \in \{1, 0, \bot\},$$

where 1 means verified, 0 means contradicted, and $\perp$ means untranslatable or unverifiable. This works for deterministic steps, but becomes brittle when a step expresses uncertainty.

Many CoT steps in biomedical, legal, and scientific reasoning contain epistemic markers such as *may*, *likely*, *suggests*, or *could indicate*. These markers are not stylistic noise; they change the meaning of the claim. For instance, s_t: "Hypertension may indicate treatment efficacy" does not assert the hard proposition $\texttt{Indicates}(\texttt{HTN}, \texttt{Efficacy})$. Instead, it expresses partial evidential support. We represent such an uncertainty-marked step as $s_t = (p_t, u_t)$, where p_t is the predicate-level content and u_t is the uncertainty condition or evidential force. In this example, p_t is the relation between hypertension and efficacy, while $u_t =$ *may* indicates weak or partial support.

This leads to the *Rigidity–Ambiguity Mismatch*: rigid verification expects binary truth values, while uncertainty-marked reasoning expresses graded support. A rigid verifier must either harden the uncertain step into a deterministic proposition, thereby changing its meaning, or return $\perp$, thereby rejecting a plausibly supported step as unverifiable.

Our goal is to verify both deterministic and uncertainty-marked steps. The verifier outputs a step-level confidence score $\kappa_t \in [0, 1]$ and a binary verification label $\hat{y}_t \in \{0, 1\}$. For deterministic steps, hard symbolic checking may be sufficient. For uncertainty-marked steps, the verifier should preserve graded support, penalize unsupported assumptions, and distinguish weakly supported inferences from hallucinations. The next section describes how PROBVERI implements these requirements using probabilistic soft logic and contrastive energy-based verification.

3 Method

PROBVERI verifies CoT reasoning with a dual-path design. Deterministic steps are handled by a rigid symbolic verifier, preserving the precision of solver-based verification when the step can be faithfully formalized as a hard claim. In contrast, uncertainty-marked steps are verified by a probabilistic soft-logic path. The main methodological contribution lies in the soft path: For an uncertain step, it first represents the step with soft logical rules that allow partial truth. It then penalizes unsupported missing links between the context and the claim. Finally, it compares the original interpretation with its negation. A step is assigned higher confidence when the original interpretation fits the evidence better than the negated one.

3.1 Overview

Given a query Q, context C, and reasoning trace $S = \{s_1, \ldots, s_T\}$, PROBVERI verifies the trace step by step. At step t, it extracts a local support context $P_t \subseteq \mathcal{H}_{t-1}$ and routes s_t to one of two paths. Deterministic steps are checked by an SMT-style rigid verifier [19], while steps with epistemic markers or weak evidential support are checked by the PSL soft-logic path. A support gap occurs when a step is plausible but requires an additional bridge assumption beyond the local context. The two paths are then combined into unified step-level and trace-level verdicts.

3.2 Routing and Soft Formalization

The router decides which verification path should be used for each reasoning step. The router uses both lexical and contextual signals. Words such as *may*, *likely*, *suggests*, *could indicate*, and *is associated with* indicate that the step should not be treated as a hard claim. The verifier also checks whether the local context P_t directly supports the step or whether a missing bridge is needed. If s_t can be faithfully written as a deterministic claim q_t, the rigid path checks whether q_t follows from or contradicts P_t. If translation or solver execution fails, it returns $\perp$. For uncertainty-marked steps, however, $\perp$ may only mean that the step cannot be expressed as hard logic, so non-contradictory cases can be rerouted to the soft path.

For a soft-routed step, PROBVERI separates the claim content from its support strength. For example,

"Hypertension may be a biomarker of therapeutic efficacy"

is mapped to

$$\texttt{IsBiomarker}(\texttt{HTN}, \texttt{Efficacy}),$$

while *may* is kept as the uncertainty condition. The local context P_t is converted into observed facts $\mathcal{O}_t$, and a small set of candidate bridge assumptions A_t is generated to capture possible missing links

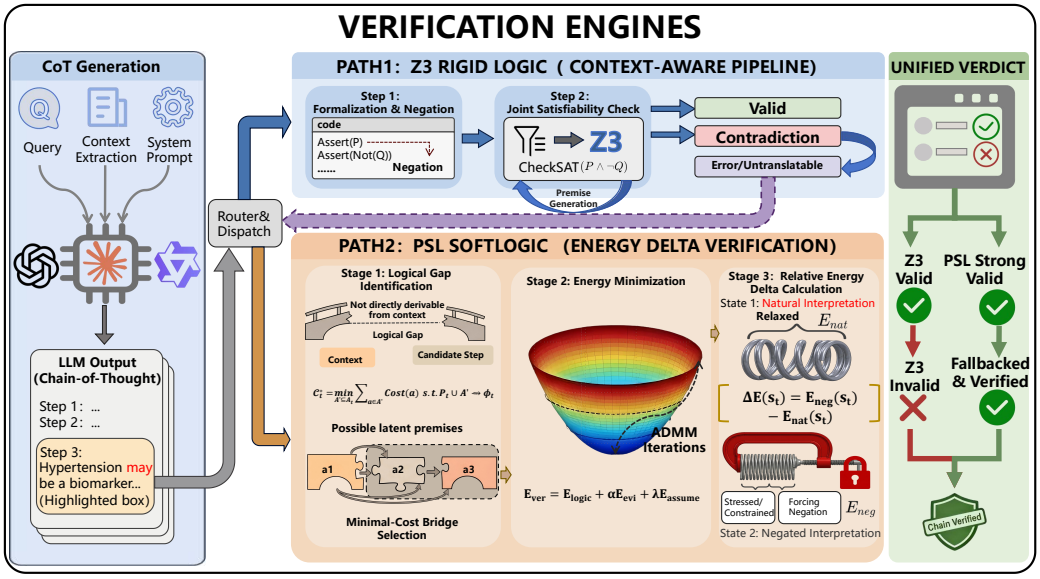


Figure 2: **Overall framework of PROBVERI.** A generated CoT trace is verified step by step through a router-and-dispatch mechanism. Deterministic steps are sent to a rigid symbolic path, while uncertainty-marked or support-gap steps are sent to a PSL soft-logic path. The soft path constructs support-grounded weighted rules, regularizes bridge assumptions, and compares natural and negated interpretations through an energy gap. The two paths produce unified step- and trace-level verdicts.

between the context and the claim. Bridge candidates are treated as *proposals* rather than accepted facts. A candidate is retained only if it preserves entity alignment, matches an allowed bridge type, and does not contradict the local context or previously verified steps; otherwise it is rejected or assigned high cost. The bridge generator cannot by itself verify a step, but only defines a bounded candidate set that is later penalized and tested by the contrastive energy objective.

3.3 Support-Grounded PSL Energy

The soft path checks whether the claim is supported without allowing the verifier to freely invent missing premises. Bridge assumptions are allowed, but they have costs. Bridge costs are automatically determined by evidence source, semantic specificity, structural complexity, and contradiction status, using fixed coefficients rather than per-example manual annotations. The minimum-cost search considers up to eight candidate bridges and subsets containing at most two assumptions. Let $A_t = \{a_{t,1}, \ldots, a_{t,K}\}$ be the candidate bridge assumptions. We define the minimum cost needed to connect the context to the claim as

$$\begin{aligned} \mathcal{C}_t^* &= \min_{A' \subseteq A_t} \sum_{a \in A'} \mathrm{Cost}(a) \quad \text{s.t.} \quad P_t \cup A' \rightsquigarrow \phi_t, \\ w_t &= \exp(-\eta\, \mathcal{C}_t^*)\,. \end{aligned} \tag{1}$$

Here, $\rightsquigarrow$ means approximate support, not strict entailment. If the claim is well supported, the required cost is low and w_t is high. If it depends on weak or broad assumptions, the cost is high and w_t is low. Using this support weight, PROBVERI builds a weighted PSL program over continuous truth values $\mathbf{z} \in [0,1]^m$ [15, 16]. For a soft rule $B \to H$, the violation is

$$d_r(\mathbf{z}) = \max(0, I(B) - I(H)).$$

The full verification energy is

$$E_{\mathrm{ver}}(\mathbf{z}; s_t) = \underbrace{\sum_{r \in \mathcal{R}_t} w_r \big(d_r(\mathbf{z})\big)^p}_{E_{\mathrm{logic}}(\mathbf{z};s_t),\ p\in\{1,2\}} + \alpha \underbrace{\sum_{o \in \mathcal{O}_t} \rho_o \big(z_o - y_o\big)^2}_{E_{\mathrm{evi}}(\mathbf{z};s_t)} + \lambda \underbrace{\sum_{a_{t,i} \in A_t} \mathrm{Cost}(a_{t,i})\, z_{a_{t,i}}}_{E_{\mathrm{assume}}(\mathbf{z};s_t)}. \tag{2}$$

The proof follows from the convexity of PSL hinge-loss potentials, squared evidence penalties, and nonnegative linear assumption penalties; details are given in Appendix B. The three terms respectively measure rule violation, disagreement with observed evidence, and reliance on costly assumptions. E_{logic} penalizes violations of grounded soft rules, E_{evi} keeps inferred truth values close to observed atoms $o \in \mathcal{O}_t$, and E_{assume} penalizes weakly supported or costly latent bridges. Thus, a low-energy interpretation must fit the soft rules, agree with evidence, and avoid unsupported assumptions. We optimize this hinge-loss objective with standard PSL inference using ADMM [20]; solver details are given in Appendix I.

3.4 Contrastive Verification and Trace Verdict

Soft verification should not accept a claim merely because its natural interpretation can be made low-energy. If a claim is underconstrained, its negation may also achieve low energy. In that case, the evidence does not clearly support one interpretation over the other. This is the key difference between PROBVERI and a soft-only PSL verifier. A soft-only verifier may accept a step when the natural interpretation has low energy, even if the negated interpretation fits the same evidence equally well. In contrast, PROBVERI compares the two interpretations directly. It accepts a step only when the natural interpretation fits the shared evidence, rules, bridge candidates, and penalties better than the negated alternative. Let $\mathcal{Z}_{\text{nat}}(s_t)$ be the feasible space that softly enforces the target relation in ϕ_t, and let $\mathcal{Z}_{\text{neg}}(s_t)$ be the feasible space that enforces the complementary condition that the relation is not supported. Both use the same evidence atoms, PSL rules, bridge assumptions, and penalty parameters. We define the contrastive energy gap as

$$\Delta E(s_t) = \underbrace{\min_{\mathbf{z} \in \mathcal{Z}_{\text{neg}}(s_t)} E_{\text{ver}}(\mathbf{z}; s_t)}_{E_{\text{neg}}(s_t)} - \underbrace{\min_{\mathbf{z} \in \mathcal{Z}_{\text{nat}}(s_t)} E_{\text{ver}}(\mathbf{z}; s_t)}_{E_{\text{nat}}(s_t)} . \tag{3}$$

A positive $\Delta E(s_t)$ means that the natural interpretation has lower energy than the negated interpretation under the same evidence and assumptions. In other words, the original step explains the available information better than its negation. This follows the standard energy-based view of inference, where more compatible configurations receive lower energy [21]. The confidence score κ_t comes from this contrast, rather than from the language model's own reported confidence.

$$\kappa_t = \sigma(\gamma \Delta E(s_t)) . \tag{4}$$

Finally, PROBVERI combines the outputs from the two paths. Rigidly verified steps are accepted, contradicted steps are rejected, and soft-routed steps are accepted according to their confidence κ_t. At the trace level, PROBVERI aggregates the step-level decisions and confidences. In our main experiments, a reasoning trace is counted as verified only when all evaluated steps pass the verification threshold and no step is contradicted. Full pseudocode and solver details are provided in Appendix A.

4 Experiments

We evaluate whether PROBVERI improves formal verification under uncertainty. The experiments focus on three aspects: verified-correct recovery across datasets and reasoning backbones, the contribution of each proposed component, and verification behavior under different support conditions.

4.1 Experimental Setup

Benchmarks. We evaluate on three benchmarks that cover complementary verification regimes. ProofWriter tests controlled logical entailment [22], BioASQ tests biomedical reasoning where intermediate steps often express epistemic uncertainty or association [23], and LegalBench-SARA tests statutory reasoning with conditional or weakly grounded steps [24, 25]. We generate CoT traces using three reasoning backbones: GPT-5.1, Claude 3.5, and Qwen3-8B, all with temperature 0.0. The verifier does not use the backbone's self-reported confidence; all soft verification scores are computed from the proposed contrastive energy objective.

Baselines. We compare against representative verification baselines that produce explicit verification labels. *(1) LLM-as-a-Judge* is an implemented neural-judgment baseline inspired by verifier-style evaluation and process-supervision work [26, 27]. *(2) Explanation-Refiner-style Verification* represents theorem-prover-guided explanation refinement adapted as a verification baseline without revising the final answer [13]. *(3) Direct SMT Checking* directly translates each step into hard symbolic constraints and verifies it through satisfiability checking [19]. *(4) Rigid Verification w/o Premise Generation* removes explicit premise or bridge generation from the rigid neuro-symbolic pipeline. *(5) VeriCoT-style Rigid Verification* represents a stronger rigid neuro-symbolic verifier with premise extraction and formal consistency checking [14]. *(6) Soft-only PSL Verification* replaces hard symbolic checking with soft logical constraints [15, 16] but does not use natural-versus-negated contrast or assumption regularization. Answer-level methods such as self-consistency do not produce step-level verification labels and are therefore not included in the main verification table.

Comparison fairness. All verification methods are evaluated on the same generated CoT traces and source context under consistent backbone and decoding settings. Importantly, Soft-only PSL and PROBVERI share the same local support contexts P_t, observed facts $\mathcal{O}_t$, candidate bridge assumptions A_t, bridge costs, and grounded PSL rules. Within the soft verification path, the two methods differ only in assumption regularization and natural-versus-negated energy comparison. This controlled comparison isolates the contribution of the proposed verification mechanisms rather than differences in the underlying soft-logic formalization resources.

Metrics. We report Pass Rate (PR.), Precision (Prec.), and Verified Correct Answer Rate (VCAR) for each verifier. VCAR is defined as $\frac{\#\mathrm{VerifiedCorrect}}{\#\mathrm{Total}}$. Equivalently, when Pass Rate and Precision are written as fractions, $\mathrm{VCAR} = \mathrm{PassRate} \times \mathrm{Precision}$, and is our primary metric because it jointly reflects coverage and reliability. Task Accuracy (TA.) measures the correctness of the underlying generated answers. Since all verifiers are applied to the same generated CoT traces and do not revise final answers, TA. is reported once for each dataset–backbone block rather than repeated for every verification method.

Computational overhead. Under the same evaluation protocol, Soft-only PSL and PROBVERI require an average of 6.61 and 7.56 LLM calls, respectively, with average runtimes of 33.1 and 37.6 seconds. These results indicate that the additional verification mechanisms introduce a moderate computational overhead while improving verified-correct recovery.

4.2 Main Results across Backbones

Table 1 reports verification performance across three benchmarks and three CoT-generating backbones. Since the verifier does not change the final answer, Task Accuracy is fixed within each backbone block; the main comparison is therefore between verification coverage, measured by Pass Rate, and verification reliability, measured by Precision and VCAR.

The results show a consistent coverage–reliability trade-off. Direct SMT Checking and rigid verification variants tend to achieve high precision, but their pass rates are low, especially on BioASQ and LegalBench-SARA, where intermediate reasoning often contains uncertainty, association, or assumption-dependent support. Soft-only PSL Verification substantially improves coverage, but its lower precision suggests that purely soft relaxation can accept some weakly supported reasoning bridges.

PROBVERI achieves the highest VCAR in every benchmark–backbone block. Its advantage does not come from maximizing Pass Rate alone, nor from being as selective as rigid checkers. Instead, it preserves much of the coverage benefit of PSL while using rigid consistency checks and contrastive energy comparison to reject unsupported soft matches. This is why Soft-only PSL remains a strong and close baseline, but PROBVERI consistently yields a better verified-correct rate across different reasoning backbones.

4.3 Ablation Study

Table 2 reports ablations on GPT-5.1-generated CoT traces under the same test examples and validation-selected thresholds. Parsing, formalization, solver, and timeout failures are counted as not passed. Soft-only PSL is included as a reference baseline, while w/o Soft Verification Path

Table 1: **Main verification results across reasoning backbones.** Bold numbers indicate the best VCAR within each benchmark and reasoning-backbone block.

Method	ProofWriter			BioASQ			LegalBench-SARA		
	PR.	**Prec.**	**VCAR**	**PR.**	**Prec.**	**VCAR**	**PR.**	**Prec.**	**VCAR**
GPT-5.1-generated CoT traces	(TA.: ProofWriter 78.1, BioASQ 81.7, LegalBench-SARA 80.4)								
LLM-as-a-Judge	53.8	79.1	42.6	38.7	66.1	25.6	31.2	61.5	19.2
Explanation-Refiner-style Verification	18.1	80.7	14.6	3.1	77.4	2.4	8.4	89.3	7.5
Direct SMT Checking	10.9	92.7	10.1	6.4	73.4	4.7	5.3	88.7	4.7
Rigid Verification w/o Premise Gen.	5.8	91.4	5.3	3.4	64.7	2.2	1.2	58.3	0.7
VeriCoT-style Rigid Verification	47.3	93.0	44.0	27.2	86.4	23.5	18.1	88.4	16.0
Soft-only PSL Verification	63.5	75.4	47.9	46.8	67.7	31.7	38.1	60.9	23.2
PROBVERI	63.1	84.0	**53.0**	46.9	78.9	**37.0**	36.4	77.5	**28.2**
Claude 3.5-generated CoT traces	(TA.: ProofWriter 77.0, BioASQ 80.9, LegalBench-SARA 80.3)								
LLM-as-a-Judge	52.2	78.0	40.7	37.9	64.1	24.3	28.6	61.5	17.6
Explanation-Refiner-style Verification	14.5	82.1	11.9	1.8	77.8	1.4	7.1	88.7	6.3
Direct SMT Checking	10.3	93.2	9.6	5.7	73.7	4.2	4.7	91.5	4.3
Rigid Verification w/o Premise Gen.	3.6	91.7	3.3	2.5	60.0	1.5	0.7	42.9	0.3
VeriCoT-style Rigid Verification	45.0	93.8	42.2	25.8	83.7	21.6	15.5	86.5	13.4
Soft-only PSL Verification	60.5	74.4	45.0	44.4	64.6	28.7	35.1	57.0	20.0
PROBVERI	61.3	82.5	**50.6**	44.8	73.9	**33.1**	33.1	71.5	**23.7**
Qwen3-8B-generated CoT traces	(TA.: ProofWriter 73.6, BioASQ 76.8, LegalBench-SARA 76.1)								
LLM-as-a-Judge	47.2	72.5	34.2	32.5	58.2	18.9	25.7	52.9	13.6
Explanation-Refiner-style Verification	12.7	77.2	9.8	1.3	69.2	0.9	5.2	82.7	4.3
Direct SMT Checking	9.3	90.3	8.4	4.6	73.9	3.4	4.1	82.9	3.4
Rigid Verification w/o Premise Gen.	2.5	92.0	2.3	1.9	57.9	1.1	0.6	50.0	0.3
VeriCoT-style Rigid Verification	40.3	90.8	36.6	19.5	79.0	15.4	10.9	81.7	8.9
Soft-only PSL Verification	56.9	69.6	39.6	39.8	57.3	22.8	32.2	48.1	15.5
PROBVERI	57.5	77.9	**44.8**	38.8	69.8	**27.1**	28.7	66.4	**19.0**

corresponds to the rigid-path limit in Table 1. Each block reports Pass Rate, Precision, VCAR, and absolute VCAR change relative to the full model.

Table 2: **Ablation study on GPT-5.1-generated CoT traces.** We report Pass Rate (PR.), Precision (Prec.), VCAR, and absolute VCAR change (Δ) relative to the full model. Soft-only PSL is included as a reference baseline.

Variant	ProofWriter				BioASQ				LegalBench-SARA			
	PR.	**Prec.**	**VCAR**	Δ	**PR.**	**Prec.**	**VCAR**	Δ	**PR.**	**Prec.**	**VCAR**	Δ
PROBVERI	63.1	84.0	**53.0**	–	46.9	78.9	**37.0**	–	36.4	77.5	**28.2**	–
w/o Soft Verification Path	47.3	93.0	44.0	-9.0	27.2	86.4	23.5	-13.5	18.1	88.4	16.0	-12.2
w/o Negated-Energy Comparison	60.7	76.0	46.1	-6.9	43.2	68.7	29.7	-7.3	34.7	67.4	23.4	-4.8
w/o Assumption Penalty	66.4	71.2	47.3	-5.7	54.2	60.7	32.9	-4.1	42.5	55.8	23.7	-4.5
w/o Support-Grounded Weighting	58.8	78.1	45.9	-7.1	42.4	70.0	29.7	-7.3	34.2	63.4	21.7	-6.5
Direct LLM Confidence as Weight	57.6	78.5	45.2	-7.8	41.5	67.5	28.0	-9.0	33.9	62.2	21.1	-7.1
Soft-only PSL Verification (reference)	63.5	75.4	47.9	-5.1	46.8	67.7	31.7	-5.3	38.1	60.9	23.2	-5.0

The ablations clarify that the gains do not come from soft relaxation alone. Removing the soft verification path substantially reduces recovery, especially on BioASQ and LegalBench-SARA. However, removing constraints such as the assumption penalty can increase Pass Rate while lowering Precision, indicating that higher coverage may come from over-accepting weakly grounded bridges. The full model achieves the best VCAR by preserving much of the recovery benefit of PSL while improving reliability through contrastive verification and assumption regularization.

Table 3: Representative cases of uncertainty-aware verification. The examples illustrate how PROBVERI distinguishes deterministic entailment, supported uncertainty, and unsupported inference.

Dataset / Case	Reasoning Step	Path	Verification Outcome
ProofWriter	Fiona is blue; all blue animals are kind; therefore, Fiona is kind.	Rigid	**Accepted.** The conclusion follows from explicit premises.
BioASQ	ADHD might be associated with the Dp71 genotype.	PSL	**Accepted.** Partial contextual evidence supports the association.
LegalBench-SARA	The taxpayer is likely eligible under the exception if the statutory condition holds.	PSL	**Accepted.** A contextual bridge supports the conditional inference.
BioASQ	The treatment likely improves long-term survival.	PSL	**Rejected.** Short-term response evidence does not establish long-term survival benefit.

4.4 Case Study: Verification under Different Support Conditions

To illustrate how PROBVERI distinguishes deterministic entailment, partial evidential support, and unsupported inferences, we examine four representative verification cases from ProofWriter, BioASQ, and LegalBench-SARA. Table 3 summarizes their reasoning steps, verification paths, and resulting decisions. These cases complement the aggregate results in Tables 1–2 by illustrating the behavior of the dual-path verification mechanism.

Analysis. The cases illustrate three complementary verification behaviors. First, deterministic entailment remains under rigid symbolic verification, preserving the ability to validate logically supported conclusions. Second, uncertainty-marked biomedical and legal inferences can be evaluated through soft logical constraints without requiring deterministic entailment. Third, soft verification does not automatically accept every plausible inference: when supporting evidence is insufficient, the contrastive energy comparison and assumption penalty can prevent acceptance.

Together, these examples clarify the role of the dual-path architecture. Rigid verification handles explicit logical entailment, while the PSL path evaluates graded evidential support and distinguishes supported uncertainty from unsupported extrapolation.

5 Related Work

Process-level reasoning verification. Chain-of-Thought prompting makes intermediate reasoning visible [1, 2], but fluent steps are not necessarily supported by the given context. Answer-level sampling and reranking methods improve reliability by selecting more consistent or better-scored reasoning paths [3, 26]. Process-level supervision and reward modeling further encourage intermediate steps to be locally correct rather than only final-answer accurate [27, 28]. Another line of work checks explanations through entailment models, natural-language inference, or structured explanation refinement [29, 30, 31]. Self-refinement and LLM-as-a-judge methods use model feedback to revise or evaluate reasoning traces [6, 7, 32, 33]. Recent analyses show that LLM self-verification and critic models can still struggle to identify invalid reasoning steps, especially in logical reasoning, planning, and long-CoT settings [34, 35, 36]. These methods are useful for answer selection and broad error detection, but they usually rely on learned semantic judgments or answer-level agreement rather than explicit formal constraints. Our work targets a different regime, where a step may be plausible and evidence-dependent without being deductively entailed.

Rigid neuro-symbolic verification. Another line of work translates natural-language reasoning into programs, first-order constraints, theorem-prover statements, or SMT formulas [9, 10]. These formal objects can then be checked for entailment, consistency, or executability using external solvers or symbolic execution engines [11, 12]. Recent verifier systems further combine premise extraction, formal consistency checking, explanation refinement, or autoformalization to improve symbolic verification of reasoning traces [13, 14, 37]. Related work also studies faithful formalization and automatic translation from language to formal representations [17, 38]. Benchmarks and analyses

show that logical reasoning in language models remains brittle across first-order, symbolic, and multi-step reasoning settings [39, 18, 40]. Such methods provide strong guarantees when the premises are explicit and the target claim can be faithfully represented as a hard symbolic statement. However, most rigid verifiers assume binary truth values and strict entailment. When a CoT step contains markers such as *may*, *likely*, *suggests*, or *is associated with*, hard translation either removes the uncertainty or fails to produce a faithful formal object. This rigidity motivates a verifier that can preserve graded evidential support.

Soft logic and energy-based inference. Soft probabilistic logics provide a natural tool for reasoning with noisy evidence, partial observability, and relational structure [41, 42]. Probabilistic graphical and relational models have also been used to combine uncertain evidence with structured dependencies [43]. Probabilistic Soft Logic (PSL) relaxes Boolean truth values to $[0, 1]$ and represents weighted first-order rules as hinge-loss potentials [15, 16]. Related structured-learning work has studied relational prediction and heterogeneous information networks [44], as well as generalizable inference in structured dynamical systems [45]. Prior PSL-style models have been used for information extraction, entity resolution, knowledge-base completion, collective classification, and credibility modeling. Energy-based inference similarly treats reasoning as compatibility minimization, assigning lower energy to configurations that better satisfy evidence and constraints [21]. Recent neuro-symbolic frameworks further explore flexible logical composition and inference. NeSyCoCo introduces neuro-symbolic concept composition for compositional generalization, while NePTune explores neuro-Pythonic compositional reasoning in vision-language settings [46, 47]. NeuPSL integrates neural predicates with probabilistic soft logic through an energy-based framework for joint learning and inference [48]. Unlike these approaches, PROBVERI focuses on post-hoc verification of uncertainty-marked natural-language CoT steps, using support-grounded assumption regularization and natural-versus-negated energy comparison to assess whether a generated inference is supported by its context. Complementary research on imperfect supervision has examined correlations among labels in label distribution learning [49], biased label-distribution annotations [50], and instance-dependent annotation noise [51]. These studies focus on learning from uncertain labels rather than verifying the evidential support of generated reasoning steps.

6 Conclusion and Limitations

We presented PROBVERI, an energy-based framework for verifying Chain-of-Thought reasoning under epistemic uncertainty. Motivated by the Rigidity–Ambiguity Mismatch, PROBVERI avoids forcing uncertainty-marked steps into binary entailment and instead compares natural and negated interpretations under shared evidence, weighted soft rules, and assumption regularization. Experiments across logical, biomedical, and legal reasoning benchmarks show improved verified-correct recovery over neural, rigid symbolic, and soft-only baselines while retaining selectivity. The framework remains limited by approximate bridge-assumption search, predicate-skeleton extraction errors, and the fact that the confidence score is a verification signal rather than a fully calibrated probability. Future work should improve assumption filtering, domain-adaptive predicate extraction, and scalable uncertainty-aware verification for longer traces. The resulting step-level verification signals may also support future applications such as targeted reasoning correction, candidate-trace reranking, and verifier-guided reasoning exploration.

Acknowledgments and Disclosure of Funding

Funding. This work was supported by the National Natural Science Foundation of China (NSFC) under Grant No. 72601425 and the Science Challenge Project (No. TZ2025005). Z. Lin was supported by the Beijing Natural Science Foundation (No. L257007), the Shanghai Artificial Intelligence Laboratory (No. 2026ZD0125500), the National Natural Science Foundation of China (No. 62276004), and the Beijing Major Science and Technology Project (No. Z251100008425006). H. Li was supported by the Shanghai General AI Foundation Models Program (No. 2025SHZDZX026D35) and the CCF-Kuaishou Large Model Explorer Fund (No. CCF-KuaiShou 2026018).

Competing Interests. The authors declare no competing interests.

References

[1] 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.

[2] 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.

[3] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

[4] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022.

[5] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

[6] 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.

[7] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

[8] Xuan Zhang, Chao Du, Tianyu Pang, Qian Liu, Wei Gao, and Min Lin. Chain of preference optimization: Improving chain-of-thought reasoning in llms. *Advances in Neural Information Processing Systems*, 37:333–356, 2024.

[9] Zhan Ling, Yunhao Fang, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, and Hao Su. Deductive verification of chain-of-thought reasoning. *Advances in Neural Information Processing Systems*, 36:36407–36433, 2023.

[10] Liangming Pan, Alon Albalak, Xinyi Wang, and William Wang. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3806–3824, 2023.

[11] Theo Olausson, Alex Gu, Ben Lipkin, Cedegao Zhang, Armando Solar-Lezama, Joshua Tenenbaum, and Roger Levy. Linc: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5153–5176, 2023.

[12] Xi Ye, Qiaochu Chen, Isil Dillig, and Greg Durrett. Satlm: Satisfiability-aided language models using declarative prompting. *Advances in Neural Information Processing Systems*, 36: 45548–45580, 2023.

[13] Xin Quan, Marco Valentino, Louise A Dennis, and André Freitas. Verification and refinement of natural language explanations through llm-symbolic theorem proving. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 2933–2958, 2024.

[14] Yu Feng, Nathaniel Weir, Kaj Bostrom, Sam Bayless, Darion Cassel, Sapana Chaudhary, Benjamin Kiesl-Reiter, and Huzefa Rangwala. Vericot: Neuro-symbolic chain-of-thought validation via logical consistency checks. *arXiv preprint arXiv:2511.04662*, 2025.

[15] Angelika Kimmig, Stephen Bach, Matthias Broecheler, Bert Huang, Lise Getoor, Vikash Mansinghka, Daniel Roy, and Noah Goodman. A short introduction to probabilistic soft logic. In *Proceedings of the NIPS workshop on probabilistic programming: foundations and applications*, pages 1–4, 2012.

[16] Stephen H Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. Hinge-loss markov random fields and probabilistic soft logic. *Journal of Machine Learning Research*, 18(109):1–67, 2017.

[17] Jundong Xu, Hao Fei, Liangming Pan, Qian Liu, Mong-Li Lee, and Wynne Hsu. Faithful logical reasoning via symbolic chain-of-thought. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13326–13365, 2024.

[18] Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, et al. Folio: Natural language reasoning with first-order logic. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22017–22031, 2024.

[19] Leonardo De Moura and Nikolaj Bjørner. Z3: An efficient smt solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340. Springer, 2008.

[20] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations Trends in Machine Learning*, 3(1):1–122, 2010.

[21] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, Fujie Huang, et al. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

[22] Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. Proofwriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634, 2021.

[23] Anastasios Nentidis, Georgios Katsimpras, Anastasia Krithara, Salvador Lima-López, Eulàlia Farré-Maduell, Martin Krallinger, Natalia Loukachevitch, Vera Davydova, Elena Tutubalina, and Georgios Paliouras. Overview of bioasq 2024: the twelfth bioasq challenge on large-scale biomedical semantic indexing and question answering. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 3–27. Springer, 2024.

[24] Neel Guha, Julian Nyarko, Daniel Ho, Christopher Ré, Adam Chilton, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel Rockmore, Diego Zambrano, et al. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models. *Advances in neural information processing systems*, 36:44123–44279, 2023.

[25] Nils Holzenberger, Andrew Blair-Stanek, and Benjamin Van Durme. A dataset for statutory reasoning in tax law entailment and question answering. *arXiv preprint arXiv:2005.05257*, 2020.

[26] 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, 2021. *URL https://arxiv. org/abs/2110.14168*, 9, 2021.

[27] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The twelfth international conference on learning representations*, 2023.

[28] Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439, 2024.

[29] Bhavana Dalvi, Peter Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark. Explaining answers with entailment trees. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pages 7358–7370, 2021.

[30] Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. Entailer: Answering questions with faithful and truthful chains of reasoning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2078–2093, 2022.

[31] Robert Vacareanu, Anurag Pratik, Evangelia Spiliopoulou, Zheng Qi, Giovanni Paolini, Neha Anna John, Jie Ma, Yassine Benajiba, and Miguel Ballesteros. General purpose verification for chain of thought prompting. *arXiv preprint arXiv:2405.00204*, 2024.

[32] Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. Chain-of-verification reduces hallucination in large language models. In *Findings of the association for computational linguistics: ACL 2024*, pages 3563–3578, 2024.

[33] Alon Jacovi, Yonatan Bitton, Bernd Bohnet, Jonathan Herzig, Or Honovich, Michael Tseng, Michael Collins, Roee Aharoni, and Mor Geva. A chain-of-thought is as strong as its weakest link: A benchmark for verifiers of reasoning chains. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4615–4634, 2024.

[34] Ruixin Hong, Hongming Zhang, Xinyu Pang, Dong Yu, and Changshui Zhang. A closer look at the self-verification abilities of large language models in logical reasoning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 900–925, 2024.

[35] Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. On the self-verification limitations of large language models on reasoning and planning tasks. *arXiv preprint arXiv:2402.08115*, 2024.

[36] Yancheng He, Shilong Li, Jiaheng Liu, Weixun Wang, Xingyuan Bu, Ge Zhang, Zy Peng, Zhaoxiang Zhang, Zhicheng Zheng, Wenbo Su, et al. Can large language models detect errors in long chain-of-thought reasoning? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18468–18489, 2025.

[37] Jinlong Tian, Jiang Yu, Kewei Cheng, Fengxiang Cheng, Yue He, Yunfei Wang, Haotian Wang, Haoxuan Li, Wenjing Yang, and Shixuan Liu. Logicsage: Neuro-symbolic reasoning with socratic-guided enhancement. In *Forty-third International Conference on Machine Learning*, 2026.

[38] Abhinav Lalwani, Tasha Kim, Lovish Chopra, Christopher Hahn, Zhijing Jin, and Mrinmaya Sachan. Autoformalizing natural language to first-order logic: A case study in logical fallacy detection. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pages 132–147, 2025.

[39] Abulhair Saparov and He He. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. *arXiv preprint arXiv:2210.01240*, 2022.

[40] Mihir Parmar, Nisarg Patel, Neeraj Varshney, Mutsumi Nakamura, Man Luo, Santosh Mashetty, Arindam Mitra, and Chitta Baral. Logicbench: Towards systematic evaluation of logical reasoning ability of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13679–13707, 2024.

[41] Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine learning*, 62(1): 107–136, 2006.

[42] Lise Getoor and Ben Taskar. *Introduction to statistical relational learning*. MIT press, 2007.

[43] Matthias Brocheler, Lilyana Mihalkova, and Lise Getoor. Probabilistic similarity logic. *arXiv preprint arXiv:1203.3469*, 2012.

[44] Shixuan Liu, Changjun Fan, Kewei Cheng, Yunfei Wang, Peng Cui, Yizhou Sun, and Zhong Liu. Inductive meta-path learning for schema-complex heterogeneous information networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):10196–10209, 2024.

[45] Shixuan Liu, Yue He, Haotian Wang, Wenjing Yang, Yunfei Wang, Peng Cui, and Zhong Liu. Environment inference for learning generalizable dynamical system. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[46] Danial Kamali, Elham J. Barezi, and Parisa Kordjamshidi. Nesycoco: a neuro-symbolic concept composer for compositional generalization. AAAI'25/IAAI'25/EAAI'25. AAAI Press, 2025. ISBN 978-1-57735-897-8. doi: 10.1609/aaai.v39i4.32439. URL https://doi.org/10.1609/aaai.v39i4.32439.

[47] Danial Kamali and Parisa Kordjamshidi. Neptune: A neuro-pythonic framework for tunable compositional reasoning on vision-language. In *International Conference on Learning Representations*, volume 2026, pages 86124–86148, 2026.

[48] Connor Pryor, Charles Dickens, Eriq Augustine, Alon Albalak, William Yang Wang, and Lise Getoor. Neupsl: Neural probabilistic soft logic. In Edith Elkind, editor, *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 4145–4153. International Joint Conferences on Artificial Intelligence Organization, 8 2023. doi: 10.24963/ijcai.2023/461. URL https://doi.org/10.24963/ijcai.2023/461. Main Track.

[49] Zhiqiang Kou, Jing Wang, Jiawei Tang, Yuheng Jia, Boyu Shi, and Xin Geng. Exploiting multi-label correlation in label distribution learning. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 4326–4334, 8 2024.

[50] Zhiqiang Kou, Si Qin, Hailin Wang, Jing Wang, Mingkun Xie, Shuo Chen, Yuheng Jia, Tongliang Liu, Masashi Sugiyama, and Xin Geng. Label distribution learning with biased annotations assisted by multi-label learning. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, 8 2025.

[51] Zhiqiang Kou, Jing Wang, Yuheng Jia, Biao Liu, and Xin Geng. Instance-dependent inaccurate label distribution learning. *IEEE Transactions on Neural Networks and Learning Systems*, 36 (1):1425–1437, 2025. doi: 10.1109/TNNLS.2023.3329870.

A Full Algorithmic Details

This appendix provides the full execution details of PROBVERI. The main paper describes the conceptual pipeline, while this section specifies the step-level decision procedure, router, rigid path, soft-path contrastive energy verifier, and trace-level aggregation rule.

A.1 Overall Verification Procedure

Algorithm 1 PROBVERI: Energy-Based Chain-of-Thought Verification

Input: query Q, context C, reasoning trace $S = \{s_1, \ldots, s_T\}$
Output: step-level scores $\{\kappa_t\}_{t=1}^{T}$, labels $\{\hat{y}_t\}_{t=1}^{T}$, trace score κ_{trace}, trace label $\hat{y}_{\text{trace}}$
$\mathcal{H}_0 \leftarrow C, \quad \mathcal{V} \leftarrow \emptyset$
for $t = 1, \ldots, T$ **do**
  Extract local support context $P_t \subseteq \mathcal{H}_{t-1}$
  $r_t \leftarrow \textsc{Route}(s_t, P_t)$
  if $r_t = \texttt{Rigid}$ **then**
    $o_t \leftarrow \textsc{RigidCheck}(s_t, P_t)$
    if $o_t = \texttt{Verified}$ **then**
      $\kappa_t \leftarrow 1.0, \quad \hat{y}_t \leftarrow 1$
    else if $o_t = \texttt{Contradicted}$ **then**
      $\kappa_t \leftarrow 0.0, \quad \hat{y}_t \leftarrow 0$
    else
      $r_t \leftarrow \texttt{Soft}$ ▷ reroute untranslatable or underdetermined steps
    end if
  end if
  if $r_t = \texttt{Soft}$ **then**
    $(\phi_t, u_t) \leftarrow \textsc{ExtractSkeleton}(s_t)$
    $\mathcal{O}_t \leftarrow \textsc{ExtractEvidenceAtoms}(P_t)$
    $A_t \leftarrow \textsc{GenerateBridgeCandidates}(P_t, \phi_t)$
    $\mathcal{C}_t^* \leftarrow \textsc{ApproxRepairCost}(P_t, \phi_t, A_t)$
    $w_t \leftarrow \exp(-\eta \mathcal{C}_t^*)$
    $\mathcal{R}_t \leftarrow \textsc{BuildPSLRules}(\phi_t, w_t, \mathcal{O}_t, A_t)$
    $E_{\text{nat}}(s_t) \leftarrow \min_{\mathbf{z} \in \mathcal{Z}_{\text{nat}}(s_t)} E_{\text{ver}}(\mathbf{z}; s_t)$
    $E_{\text{neg}}(s_t) \leftarrow \min_{\mathbf{z} \in \mathcal{Z}_{\text{neg}}(s_t)} E_{\text{ver}}(\mathbf{z}; s_t)$
    $\Delta E_t \leftarrow E_{\text{neg}}(s_t) - E_{\text{nat}}(s_t)$
    $\kappa_t \leftarrow \sigma(\gamma \Delta E_t)$
    $\hat{y}_t \leftarrow \mathbb{I}[\kappa_t > \tau_{\text{verify}}]$
  end if
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{(s_t, \kappa_t, \hat{y}_t, r_t)\}$
  $\mathcal{H}_t \leftarrow \mathcal{H}_{t-1} \cup \{s_t\}$
end for
$\kappa_{\text{trace}} \leftarrow \min_t \kappa_t$
$\hat{y}_{\text{trace}} \leftarrow \mathbb{I}[\forall t,\ \hat{y}_t = 1]$
return $\mathcal{V}, \kappa_{\text{trace}}, \hat{y}_{\text{trace}}$

The verifier proceeds sequentially because later reasoning steps may rely on earlier verified or rejected steps. In the main experiments, a trace is counted as verified only when all evaluated steps are verified and no step is contradicted. This conservative aggregation rule is chosen to match the goal of process-level reliability: a correct final answer should not be treated as fully verified if it depends on an unsupported intermediate step.

A.2 Router and Dispatch

Algorithm 2 Router and Dispatch

Input: step s_t, support context P_t
Output: route $r_t \in \{\texttt{Rigid}, \texttt{Soft}\}$
$m_t \leftarrow \textsc{DetectUncertaintyMarker}(s_t)$
$f_t \leftarrow \textsc{CheckFormalizability}(s_t)$
$g_t \leftarrow \textsc{EstimateSupportGap}(P_t, s_t)$
if $m_t = 0$ and $f_t = 1$ and $g_t \leq \tau_{\text{gap}}$ **then**
    $r_t \leftarrow \texttt{Rigid}$
else
    $r_t \leftarrow \texttt{Soft}$
end if
return r_t

The router is intentionally conservative. A step is sent to the rigid path only when it is directly formalizable and does not require uncertainty-aware repair. Steps with explicit epistemic markers, support gaps, or failed but non-contradictory formalizations are handled by the soft path. This design prevents the verifier from treating translation failure as semantic rejection when the original step is plausibly but weakly supported.

A.3 Rigid-Path Verification

Algorithm 3 Rigid Z3 Verification

Input: step s_t, support context P_t
Output: $o_t \in \{\texttt{Verified}, \texttt{Contradicted}, \bot\}$
Translate P_t into symbolic premises $\Phi(P_t)$
Translate s_t into target claim q_t
if translation fails **then**
    return $\bot$
end if
if $\Phi(P_t) \cup \{\neg q_t\}$ is UNSAT **then**
    return `Verified`
else if $\Phi(P_t) \cup \{q_t\}$ is UNSAT **then**
    return `Contradicted`
else
    return $\bot$
end if

The rigid path follows the standard refutation pattern: a deterministic claim is verified when the premises and the negation of the claim are jointly unsatisfiable. If neither the claim nor its negation can be ruled out, the rigid path returns $\bot$, which may trigger soft-path rerouting when the step is not directly contradicted. This is especially important for uncertainty-marked statements whose linguistic force cannot be faithfully represented as a hard Boolean proposition.

A.4 Soft-Path Contrastive Verification

Algorithm 4 Soft-Path Contrastive Energy Verification

Input: step s_t, support context P_t, bridge candidate budget K
Output: confidence score κ_t, label $\hat{y}_t$
Extract predicate skeleton ϕ_t and uncertainty condition u_t
Extract evidence atoms $\mathcal{O}_t$ from P_t
Generate candidate bridge set $A_t = \{a_1, \ldots, a_K\}$
Score and filter candidates using support, specificity, contradiction, and complexity
Estimate $\mathcal{C}_t^* = \min_{\substack{A' \subseteq A_t,\ |A'| \leq 2 \\ P_t \cup A' \rightsquigarrow \phi_t}} \sum_{a_i \in A'} \mathrm{Cost}(a_i)$
Compute $w_t = \exp(-\eta \mathcal{C}_t^*)$
Construct weighted PSL rules $\mathcal{R}_t$ and evidence atoms $\mathcal{O}_t$
Solve natural-state optimization to obtain $E_{\mathrm{nat}}(s_t)$
Solve negated-state optimization to obtain $E_{\mathrm{neg}}(s_t)$
Compute $\Delta E_t = E_{\mathrm{neg}}(s_t) - E_{\mathrm{nat}}(s_t)$
Compute $\kappa_t = \sigma(\gamma \Delta E_t)$
$\hat{y}_t \leftarrow \mathbb{I}[\kappa_t > \tau_{\mathrm{verify}}]$
return $\kappa_t, \hat{y}_t$

The soft path differs from a soft-only PSL verifier in the final contrastive step. A soft-only verifier accepts a claim if the natural interpretation can be assigned low energy. PROBVERI instead compares the natural and negated interpretations under the same evidence, rules, bridge candidates, and penalty parameters. This controls for underconstrained cases in which both interpretations can appear locally plausible.

B Mathematical Derivations and Proofs

B.1 PSL Relaxation

Let $I(a) \in [0, 1]$ be a soft truth assignment for each grounded atom a. For a rule $r : A_1 \wedge \cdots \wedge A_n \rightarrow H$, the Lukasiewicz relaxation evaluates the body as

$$I(A_1 \wedge \cdots \wedge A_n) = \max\left(0, \sum_{i=1}^{n} I(A_i) - n + 1\right), \tag{5}$$

and the distance to satisfaction is

$$d_r(I) = \max\left(0, I(A_1 \wedge \cdots \wedge A_n) - I(H)\right). \tag{6}$$

For a single-body implication $B \rightarrow H$, this reduces to $d_r(I) = \max(0, I(B) - I(H))$. The hinge form penalizes violations but assigns zero energy to satisfied soft implications.

B.2 Convexity of the PSL Logical Energy

Proposition 1 (Convexity of logical energy)**.** *If $w_r \geq 0$ for all $r \in \mathcal{R}$ and $d_r(\mathbf{z}) = \max(0, \ell_r(\mathbf{z}))$ with affine ℓ_r, then*

$$E(\mathbf{z}) = \sum_{r \in \mathcal{R}} w_r (d_r(\mathbf{z}))^p$$

is convex over $[0, 1]^m$ for $p = 1$ or $p = 2$.

Proof. The function $d_r(\mathbf{z}) = \max(0, \ell_r(\mathbf{z}))$ is convex because it is the pointwise maximum of an affine function and the constant zero function. If $p = 1$, $w_r d_r(\mathbf{z})$ is convex for $w_r \geq 0$. If $p = 2$, $x \mapsto x^2$ is convex and nondecreasing on $x \geq 0$, while $d_r(\mathbf{z}) \geq 0$, so $(d_r(\mathbf{z}))^2$ is convex by composition. A nonnegative weighted sum of convex functions is convex. □

B.3 Convexity of the Relaxed Verification Objective

Proposition 2 (Convexity of the relaxed verifier)**.** *Suppose E_{logic} is a PSL hinge-loss energy, E_{evi} is a nonnegative weighted squared evidence penalty, and E_{assume} is a nonnegative linear penalty over latent assumption variables. Then $E_{\text{ver}}(\mathbf{z}; s)$ is convex over the relaxed feasible domain.*

Proof. The logical energy is convex by the previous proposition. The evidence term is a nonnegative weighted sum of convex quadratic functions:

$$E_{\text{evi}}(\mathbf{z}; s) = \sum_{j \in \mathcal{O}_s} \rho_j (z_j - \hat{z}_j)^2, \qquad \rho_j \geq 0.$$

The assumption penalty is linear with nonnegative coefficients:

$$E_{\text{assume}}(\mathbf{z}; s) = \sum_i c_i a_i, \qquad c_i \geq 0.$$

Since $\alpha, \lambda \geq 0$, the full energy is a nonnegative weighted sum of convex functions and is convex. □

B.4 Rigid Verification as a Limiting Case

Proposition 3 (Rigid verification limit)**.** *If truth assignments are restricted to $\{0, 1\}$ and all violated rules receive infinite penalty, then finite-energy assignments of E_{ver} coincide with assignments satisfying the corresponding hard constraints.*

Proof. Under binary truth assignments, each rule is either satisfied or violated. If a rule is violated, its distance-to-satisfaction is positive. Sending the corresponding rule weight to infinity assigns infinite energy to any assignment violating that rule. Therefore, finite-energy assignments are exactly assignments that satisfy all hard constraints. This recovers rigid symbolic verification. □

B.5 Monotonicity of Support-Grounded Weight

Proposition 4 (Monotonic support weighting)**.** *Let $w_t = \exp(-\eta \mathcal{C}_t^*)$ with $\eta > 0$. Then w_t is strictly decreasing in the repair cost $\mathcal{C}_t^*$.*

Proof.

$$\frac{\partial w_t}{\partial \mathcal{C}_t^*} = -\eta \exp(-\eta \mathcal{C}_t^*) < 0.$$

Thus larger repair cost always yields smaller rule strength. □

B.6 Assumption Penalty Discourages Free Repair

Proposition 5 (Penalty against unsupported repair)**.** *For fixed E_{logic} and E_{evi}, increasing λ increases the energy of any solution requiring positive assumption energy.*

Proof. For any assignment with $E_{\text{assume}}(\mathbf{z}; s) > 0$,

$$\frac{\partial E_{\text{ver}}(\mathbf{z}; s)}{\partial \lambda} = E_{\text{assume}}(\mathbf{z}; s) > 0.$$

Therefore, increasing λ makes solutions relying on latent bridge assumptions less favorable. □

B.7 Interpretation of the Contrastive Gap

By definition,

$$\Delta E(s_t) = E_{\text{neg}}(s_t) - E_{\text{nat}}(s_t).$$

If $\Delta E(s_t) > 0$, then $E_{\text{nat}}(s_t) < E_{\text{neg}}(s_t)$, meaning that the natural interpretation admits a lower-energy explanation than the negated interpretation under the same evidence and constraints. If $\Delta E(s_t) \approx 0$, the evidence does not distinguish the two interpretations. If $\Delta E(s_t) < 0$, the negated interpretation is more compatible. This is why the contrastive gap is preferable to using E_{nat} alone: an underconstrained natural interpretation can have low energy even when it is not positively supported.

B.8 Why the Gap Is Not a Raw Probability

The score

$$\kappa_t = \sigma(\gamma \Delta E_t)$$

is a confidence-scored verification signal rather than a model-reported probability. The sigmoid transformation maps the contrastive energy gap into $[0, 1]$, while γ controls the sharpness of the mapping. A larger positive gap indicates stronger relative support for the natural interpretation; a larger negative gap indicates stronger relative support for the negated interpretation. We therefore use κ_t for ranking and thresholded verification decisions, rather than interpreting it as a calibrated probability.

C Energy Objective: Term-by-Term Specification

The full verification energy is

$$E_{\text{ver}}(\mathbf{z}; s_t) = E_{\text{logic}}(\mathbf{z}; s_t) + \alpha E_{\text{evi}}(\mathbf{z}; s_t) + \lambda E_{\text{assume}}(\mathbf{z}; s_t). \tag{7}$$

Logical energy. For each weighted soft rule $r \in \mathcal{R}_t$, let $d_r(\mathbf{z}) = \max(0, \ell_r(\mathbf{z}))$. We use

$$E_{\text{logic}}(\mathbf{z}; s_t) = \sum_{r \in \mathcal{R}_t} w_r [d_r(\mathbf{z})]^p, \qquad p \in \{1, 2\}. \tag{8}$$

In the experiments, $p = 2$ is used by default because it penalizes large violations more strongly while keeping the relaxed objective convex.

Evidence energy. Observed evidence atoms are either clamped or softly penalized. The soft evidence penalty is

$$E_{\text{evi}}(\mathbf{z}; s_t) = \sum_{a \in \mathcal{O}_t} \rho_a (z_a - y_a)^2, \tag{9}$$

where $y_a \in [0, 1]$ is the evidence-derived target value and ρ_a is the evidence confidence. High-confidence extracted facts use larger ρ_a; weakly extracted or paraphrased evidence uses smaller values.

Assumption energy. Let $A_t = \{a_1, \ldots, a_K\}$ be candidate bridge assumptions. We use

$$E_{\text{assume}}(\mathbf{z}; s_t) = \sum_{i=1}^{K} c_i z_{a_i}, \tag{10}$$

where $c_i = \text{Cost}(a_i)$ is the cost of introducing assumption a_i. Unsupported broad assumptions receive larger costs than context-grounded assumptions.

Support-grounded rule weight. The repair cost is

$$\mathcal{C}_t^* = \min_{A' \subseteq A_t} \sum_{a \in A'} \text{Cost}(a) \quad \text{s.t.} \quad P_t \cup A' \rightsquigarrow \phi_t, \tag{11}$$

and the corresponding rule weight is

$$w_t = \exp(-\eta \mathcal{C}_t^*). \tag{12}$$

This separates the cost of assuming missing support from the inference-time assumption penalty. The parameter η controls how repair cost affects rule strength, while λ controls how much latent assumption usage is penalized during inference.

Natural and negated energy. The two energies are computed with identical evidence, identical rule templates, and identical assumption penalties. Only the target interpretation is changed:

$$E_{\text{nat}}(s_t) = \min_{\mathbf{z} \in \mathcal{Z}_{\text{nat}}(s_t)} E_{\text{ver}}(\mathbf{z}; s_t), \qquad E_{\text{neg}}(s_t) = \min_{\mathbf{z} \in \mathcal{Z}_{\text{neg}}(s_t)} E_{\text{ver}}(\mathbf{z}; s_t). \tag{13}$$

This controls for the difficulty of the context and makes the energy gap interpretable as a relative compatibility score.

D Natural-versus-Negated Construction

For each uncertainty-marked candidate step, PROBVERI constructs a natural interpretation and a negated interpretation. The natural interpretation encodes the claim as stated. The negated interpretation encodes the competing hypothesis that the target relation is unsupported or false under the same context.

Table 4: **Examples of natural and negated interpretations.**

Uncertain step	Natural interpretation	Negated interpretation
X may indicate Y	$\texttt{Indicates}(X, Y)$	$\neg\texttt{Indicates}(X, Y)$
A is associated with B	$\texttt{Associated}(A, B)$	$\neg\texttt{Associated}(A, B)$
C likely satisfies condition D	$\texttt{Satisfies}(C, D)$	$\neg\texttt{Satisfies}(C, D)$
X suggests risk of Y	$\texttt{SuggestsRisk}(X, Y)$	$\neg\texttt{SuggestsRisk}(X, Y)$
R could be an exception to S	$\texttt{ExceptionApplies}(R, S)$	$\neg\texttt{ExceptionApplies}(R, S)$
X may be a biomarker of Y	$\texttt{IsBiomarker}(X, Y)$	$\neg\texttt{IsBiomarker}(X, Y)$

Negation for modal claims. For modal or association claims, the negated interpretation should not be read as a claim of absolute impossibility. Rather, it encodes the competing hypothesis that the available context does not support the target relation. This distinction is important for biomedical and legal reasoning, where absence of strong proof is not equivalent to proof of absence.

Shared context constraint. Both interpretations are evaluated under the same evidence atoms, weighted rules, repair candidates, and penalty settings. This prevents the verifier from comparing two hypotheses under different evidence sets.

Contrastive confidence. After obtaining E_{nat} and E_{neg}, the verifier computes

$$\Delta E = E_{\text{neg}} - E_{\text{nat}}, \qquad \kappa = \sigma(\gamma \Delta E).$$

A step is accepted only when $\kappa > \tau_{\text{verify}}$. Thus, the verifier accepts a step only when the natural interpretation is sufficiently more compatible than the negated interpretation.

E Support-Grounded Repair and Assumption Penalty

E.1 Bridge Candidate Types

We consider three classes of bridge candidates. **Lexical bridges** map paraphrases or near-synonymous predicates. **Relational bridges** connect two domain relations, such as association and indication. **Contextual bridges** are assumptions grounded in retrieved context or earlier verified steps. Candidate assumptions that are speculative or unsupported are retained only with high cost, which allows the energy objective to penalize their use.

E.2 Bridge Generation Protocol

Bridge candidates are generated as candidate structures, not as verified facts. For each soft-routed step, the generator receives the local support context P_t, the predicate skeleton ϕ_t, and the extracted evidence atoms $\mathcal{O}_t$. It may propose only three kinds of bridges: lexical bridges between near-synonymous predicates, relational bridges between compatible domain relations, and contextual bridges grounded in retrieved context or earlier verified steps. The scoring rule is fixed across datasets and does not use the language model's self-reported confidence.

Each candidate is filtered by four checks. First, entity alignment requires the bridge arguments to match entities in P_t or ϕ_t. Second, relation compatibility requires the bridge relation to connect an observed relation to the target relation. Third, context grounding requires lexical, semantic, or previously verified support from the local context. Fourth, contradiction filtering rejects candidates that conflict with observed evidence or verified steps. Candidates that introduce new entities, broad domain knowledge, or unsupported causal claims are assigned high cost or rejected. The final verifier then searches only over the top $K = 8$ candidates and subsets of size at most two, preventing arbitrary chains of assumptions.

E.3 Repair Cost

We decompose the cost of a bridge assumption a_i as

$$\mathrm{Cost}(a_i) = \beta_1 \mathrm{Cost}_{\mathrm{source}}(a_i) + \beta_2 \mathrm{Cost}_{\mathrm{specificity}}(a_i) + \beta_3 \mathrm{Cost}_{\mathrm{complexity}}(a_i) + \beta_4 \mathrm{Cost}_{\mathrm{contradiction}}(a_i).$$

The source term measures whether the assumption is directly grounded in context. The specificity term penalizes overly broad assumptions. The complexity term penalizes assumptions that introduce multiple new entities or relations. The contradiction term sharply penalizes candidates that conflict with retrieved evidence.

Table 5: **Bridge assumption cost categories.**

Category	Description	Typical Cost
Direct contextual restatement	Explicitly stated or directly paraphrased in the context	0.05–0.15
Specific domain bridge	Not stated verbatim but grounded by a domain relation	0.20–0.45
Multi-hop bridge	Requires two or more local support links	0.45–0.70
Broad unsupported bridge	Plausible but not clearly grounded in the context	0.70–1.10
Speculative assumption	Introduces unsupported domain knowledge or new entities	> 1.10
Contradictory bridge	Conflicts with retrieved evidence or earlier verified steps	rejected

E.4 Repair Search

Exact minimal repair search is combinatorial. We approximate it by first ranking bridge candidates and then searching over a bounded subset. In our implementation, we keep the top $K = 8$ candidates after contradiction filtering and evaluate subsets up to size two. This captures the most common repair patterns while preventing arbitrary assumption chains.

$$\mathcal{C}^*(P_t \Rightarrow s_t) \approx \min_{\substack{A' \subseteq A_t \\ |A'| \leq 2 \\ P_t \cup A' \rightsquigarrow \phi_t}} \sum_{a_i \in A'} \mathrm{Cost}(a_i).$$

E.5 Relationship between w_t and λ

The support-grounded weight $w_t = \exp(-\eta \mathcal{C}_t^*)$ controls how strongly the candidate rule is trusted before PSL inference. The assumption penalty coefficient λ, by contrast, controls the inference-time cost of activating latent bridge assumptions. These two terms play different roles: w_t weakens rules that require costly repair, while $\lambda E_{\mathrm{assume}}$ discourages assignments that rely heavily on latent assumptions.

F Router and Dispatch Analysis

This appendix provides additional details about the router used by PROBVERI. The router determines whether a reasoning step should be checked by the rigid path or by the uncertainty-aware soft path. The goal is not to maximize the number of soft-routed steps, but to reserve soft verification for cases where rigid formalization is likely to be unfaithful or underdetermined.

F.1 Step-Level Routing Distribution

Table 6: **Step-level routing distribution across datasets.** "Rerouted" denotes steps first attempted by the rigid path but then sent to the soft path after an untranslatable or underdetermined rigid output.

Dataset	Rigid Path (%)	Soft Path (%)	Rerouted (%)
ProofWriter	72.4	21.6	6.0
BioASQ	38.5	52.8	8.7
LegalBench-SARA	44.1	46.3	9.6

ProofWriter contains more directly formalizable logical steps, so most steps are routed to the rigid path. BioASQ and LegalBench-SARA contain more modal, association-based, or conditionally supported reasoning, so a larger fraction of steps are routed to the soft path. The rerouted column is step-level and should not be confused with trace-level Pass Rate in the main experiments.

F.2 Router Ablation

Table 7: **Router ablation on GPT-5.1-generated CoT traces.** Results are averaged over ProofWriter, BioASQ, and LegalBench-SARA.

Router Variant	Pass Rate	Precision	VCAR
Full router	48.8	80.1	**39.4**
Lexical markers only	44.3	75.6	33.5
Support-gap only	46.8	74.4	34.8
No rerouting from rigid failures	42.0	**81.0**	34.0
Route all non-contradictions to soft	**60.6**	59.9	36.3

Lexical markers alone miss implicit uncertainty, while support-gap detection alone over-routes some deterministic steps to the soft path. Rerouting is useful because semantically plausible steps may fail rigid translation without being contradicted. Routing all non-contradictions to the soft path increases Pass Rate but lowers Precision, confirming that the router is needed for selectivity.

G Rigid Z3 Verification Details

The rigid path verifies deterministic steps through satisfiability checks. Let P_t be the local premise set and q_t the candidate claim. The symbolic translation of the premises is denoted by $\Phi(P_t)$.

$$\texttt{Verified} \iff \mathrm{UNSAT}\big(\Phi(P_t) \cup \{\neg q_t\}\big).$$

$$\texttt{Contradicted} \iff \mathrm{UNSAT}\big(\Phi(P_t) \cup \{q_t\}\big).$$

If both checks are satisfiable or the translation fails, the rigid path returns $\perp$. A $\perp$ result is not equivalent to rejection; it indicates that the rigid path cannot certify the step.

Rigid Checking Example

Premise: All drugs that reduce blood pressure reduce vascular strain. Drug X reduces blood pressure.
Candidate: Drug X reduces vascular strain.
Check: Assert premises and the negated candidate. If the solver returns UNSAT, the candidate is verified.

Rigid Failure Example

Candidate: Hypertension may be a biomarker of therapeutic efficacy.
Failure mode: The modal force *may* is not faithfully represented as a hard predicate. The rigid path either hardens the claim or returns $\perp$, so the step is routed to the soft path.

G.1 Rigid Output Semantics

The rigid path has three outputs. `Verified` indicates that the hard symbolic claim follows from the translated premises under refutation. `Contradicted` indicates that the candidate is incompatible with the translated premises. The unverifiable output $\perp$ indicates either translation failure or underdetermination. In PROBVERI, $\perp$ is not automatically treated as rejection because uncertainty-marked natural language may be semantically meaningful even when hard formalization fails.

G.2 Relation to the Router

When a deterministic step is verified or contradicted by the rigid path, the decision is final. When the rigid path returns $\perp$, the router distinguishes two cases. If the step is deterministic and lacks sufficient support, the step remains unverifiable. If the step is uncertainty-marked or contains a support gap, it is rerouted to the soft path for contrastive energy verification.

H PSL Program Construction Details

This appendix describes how predicate skeletons, evidence atoms, and bridge candidates are converted into grounded PSL-style rules. The actual predicates depend on the domain, but all programs share the same structure: evidence atoms provide observed support, bridge atoms represent possible latent support, and the target atom is evaluated under natural and negated interpretations.

H.1 Rule Templates

We use a small family of PSL rule templates. Grounded rules are instantiated from predicate skeletons, context atoms, bridge assumptions, and domain-specific relation types.

$$w_1: \quad \texttt{Supports}(c, s) \rightarrow \texttt{Verified}(s), \tag{14}$$
$$w_2: \quad \texttt{Contradicts}(c, s) \rightarrow \neg\texttt{Verified}(s), \tag{15}$$
$$w_3: \quad \texttt{Associated}(x, y) \wedge \texttt{Relevant}(y, z) \rightarrow \texttt{MayIndicate}(x, z), \tag{16}$$
$$w_4: \quad \texttt{Condition}(x, c) \wedge \texttt{Satisfies}(x, c) \rightarrow \texttt{Eligible}(x), \tag{17}$$
$$w_5: \quad \texttt{Bridge}(a, s) \wedge \texttt{Supported}(a) \rightarrow \texttt{Supports}(a, s). \tag{18}$$

These templates are not intended to be a universal ontology. They provide a compact predicate-level structure for converting uncertain natural-language steps into weighted soft constraints. Domain-specific predicates are introduced only when they are grounded in the local context or in the extracted predicate skeleton.

H.2 Domain-Specific Instantiations

Table 8: **Example PSL rule instantiations.** Weights are support-grounded and may vary by instance according to repair cost.

Domain	Rule	Typical Weight
ProofWriter	$\texttt{Rule}(A \to B) \wedge \texttt{True}(A) \to \texttt{True}(B)$	0.90–1.00
BioASQ	$\texttt{Associated}(X, M) \wedge \texttt{MechanismRelevant}(M, Y) \to \texttt{MayIndicate}(X, Y)$	0.45–0.80
LegalBench-SARA	$\texttt{ConditionSatisfied}(T, C) \wedge \texttt{Applies}(C, E) \to \texttt{Eligible}(T, E)$	0.50–0.85
Generic	$\texttt{Contradicts}(C, S) \to \neg\texttt{Verified}(S)$	0.90–1.00

H.3 Evidence Atoms and Latent Atoms

Observed evidence atoms $\mathcal{O}_t$ are extracted from the local support context P_t. High-confidence observations may be clamped or given large evidence weights, while weakly matched paraphrases are softly penalized through E_{evi}. Latent atoms include target predicates, bridge assumptions, and intermediate support predicates. The same evidence and latent structure are used for both $\mathcal{Z}_{\text{nat}}(s_t)$ and $\mathcal{Z}_{\text{neg}}(s_t)$; only the target interpretation changes.

H.4 Handling Negation

For negated-state inference, the target relation is constrained toward its complement while evidence and rule weights are unchanged. This design ensures that E_{nat} and E_{neg} differ because of the interpretation being tested, not because of changes in evidence, repair candidates, or penalty settings.

I ADMM Inference and Solver Settings

PSL inference can be solved by decomposing the global hinge-loss objective into local rule subproblems. Introduce local copy x_r, global consensus variable y, and scaled dual variable u_r. The augmented Lagrangian is

$$L_\rho(x, y, u) = \sum_r w_r \phi_r(x_r) + \frac{\rho}{2} \sum_r \|x_r - y + u_r\|_2^2.$$

Each iteration alternates between local updates, consensus updates, and dual updates:

$$x_r^{k+1} \leftarrow \arg\min_{x_r} \; w_r \phi_r(x_r) + \frac{\rho}{2} \|x_r - y^k + u_r^k\|_2^2, \tag{19}$$

$$y^{k+1} \leftarrow \Pi_{[0,1]^m} \left(\frac{1}{|\mathcal{R}|} \sum_r (x_r^{k+1} + u_r^k) \right), \tag{20}$$

$$u_r^{k+1} \leftarrow u_r^k + x_r^{k+1} - y^{k+1}. \tag{21}$$

Table 9: **Default solver hyperparameters.**

Parameter	Value
ADMM penalty ρ	1.0
Maximum ADMM iterations	250
Primal tolerance	10^{-4}
Dual tolerance	10^{-4}
Energy exponent p	2
Evidence weight α	1.0
Support-cost temperature η	0.8
Assumption penalty λ	0.8
Confidence temperature γ	1.5
Verification threshold τ_{verify}	0.50
Bridge candidate budget K	8
Maximum bridge subset size	2

I.1 Complexity

Let m be the number of grounded atoms and $|\mathcal{R}|$ the number of grounded rules. Assuming bounded rule arity, each ADMM iteration scales approximately linearly in the number of grounded rule potentials. Natural and negated inference require two optimization solves per soft-routed step. In practice, the soft path is used only for uncertainty-marked or support-gap steps, which keeps total verification cost manageable.

I.2 Stopping Criteria

The solver stops when both primal and dual residuals fall below the specified tolerances or when the maximum number of iterations is reached. If convergence is not reached within the iteration budget, the current feasible assignment is retained and marked for diagnostic logging.

J Metrics and Counting Rules

We use the following metrics:

$$\text{PassRate} = \frac{\#\text{Verified}}{\#\text{Total}}, \qquad \text{Precision} = \frac{\#\text{VerifiedCorrect}}{\#\text{Verified}},$$
$$\text{VCAR} = \frac{\#\text{VerifiedCorrect}}{\#\text{Total}}, \qquad \text{FAR} = \frac{\#\text{FalseAccepted}}{\#\text{Unsupported}}.$$

Verified. A step is counted as verified when its verification label is accepting. A trace is counted as verified when all evaluated steps are verified and no step is contradicted. This trace-level rule matches the conservative aggregation rule in Appendix A.

Verified correct. A verified trace is counted as verified correct when the final answer is correct and the verified reasoning trace is consistent with the task label or gold rationale. This quantity is used to compute VCAR.

False accept. A false accept occurs when an unsupported or contradicted step receives an accepting verification label. In uncertainty-stratified analysis, FAR is computed over steps or traces marked unsupported under the corresponding annotation protocol.

Task accuracy. Task accuracy is the final-answer accuracy regardless of whether the reasoning trace is verified:

$$\text{TA} = \frac{\#\text{CorrectAnswers}}{\#\text{TotalExamples}}.$$

Because verifiers do not revise final answers, TA is reported once per dataset–backbone block in the main text.

Baselines without step-level labels. Raw CoT and Self-Consistency do not produce step-level verification decisions. We therefore report final-answer accuracy or answer-level recovery for these methods, but leave Pass Rate and Precision undefined when no explicit verification label is available.

Rounding. All percentages in the main text are reported to one decimal place.

K Dataset and Annotation Protocol

This appendix describes the evaluation datasets, trace construction procedure, uncertainty-heavy subset construction, and annotation targets used in the experiments. The main paper reports trace-level verification metrics.

K.1 Dataset Statistics

Table 10: **Dataset statistics and evaluation protocol.** The example counts correspond to the evaluation split used for the main reported results.

Dataset	Examples	Evaluation Target
ProofWriter	800	logical entailment
BioASQ	650	biomedical QA
LegalBench-SARA	272	statutory reasoning

K.2 CoT Generation and Step Segmentation

For each dataset, CoT traces are generated using the same question and context format across reasoning backbones. We use temperature 0.0 for deterministic decoding where supported. Each trace is segmented into steps by sentence boundaries and discourse markers. Very short connective fragments are merged into neighboring steps. The verifier is then applied to the segmented trace without revising the final answer.

K.3 Uncertainty-Heavy Subset Construction

We construct uncertainty-heavy subsets using three criteria: explicit epistemic markers, association predicates, and support-gap detection. A step is marked uncertainty-heavy if it contains modal or evidential language, expresses association rather than entailment, or requires bridge assumptions to connect context to claim.

K.4 Annotation and Verification Targets

For ProofWriter, correctness is derived from logical labels. For BioASQ, correctness is determined by the answer label and whether the step is consistent with the biomedical context. For LegalBench-SARA, correctness follows the statutory reasoning label. Verification is evaluated at both step and trace level; the main paper reports trace-level VCAR. A trace is counted as verified only when all evaluated steps are accepted and no step is contradicted.

L Assets, Licenses, and Terms of Use

We use existing benchmarks, solvers, and model interfaces. The paper cites the original sources for all benchmark datasets and software components. For assets distributed under provider-specific or challenge-specific terms, we follow the corresponding release terms and do not redistribute restricted raw data or closed-source model outputs beyond anonymized aggregate results.

Table 11: **Assets, datasets, models, and tools used in this work.**

Asset	Use in this paper	Access / Terms	Citation
ProofWriter	Logical reasoning benchmark	Original dataset release terms	[22]
BioASQ	Biomedical QA benchmark	BioASQ challenge data terms	[23]
LegalBench-SARA	Statutory reasoning benchmark	LegalBench and SARA release terms	[24, 25]
Z3	SMT-style rigid checking	Open-source solver release	[19]
Probabilistic Soft Logic	Soft logic formalization and inference	Original PSL release and documentation	[15, 16]
GPT-5.1	Closed-source reasoning backbone	API access under provider terms	
Claude 3.5	Closed-source reasoning backbone	API access under provider terms	
Qwen3-8B	Open-weight reasoning backbone	Model release terms	

The closed-source backbones are used only to generate CoT traces under their provider access terms. The verification method itself does not depend on proprietary model internals. The open-weight backbone is used as an auditable generator, while all reported verification decisions are computed by the verifier and not by a model's self-reported confidence.

M Experimental Protocol and Reproducibility

This appendix summarizes the experimental settings needed to reproduce the main reported trends. Full prompts are provided in Appendix Q, solver settings are provided in Appendix I, and metric definitions are provided in Appendix J.

Table 12: **Reproducibility summary.**

Item	Setting
Reasoning backbones	GPT-5.1, Claude 3.5, Qwen3-8B
Decoding	Temperature 0.0; deterministic decoding where supported
Evaluation datasets	ProofWriter, BioASQ, LegalBench-SARA
Trace segmentation	Sentence boundaries and discourse markers; short fragments merged
Rigid solver	SMT-style satisfiability and entailment checks
Soft solver	PSL-style hinge-loss inference solved by ADMM
Main threshold	$\tau_{\text{verify}} = 0.50$, selected on validation VCAR under a precision constraint
Bridge candidate budget	$K = 8$, with bridge subsets up to size two
Reported metrics	Pass Rate, Precision, VCAR, Task Accuracy, False Accept Rate
Randomness control	Fixed prompts, fixed evaluation order, fixed candidate budgets, and deterministic decoding where supported

M.1 Model and Prompting Details

The same dataset prompts are used across reasoning backbones. The backbone is used to generate the initial CoT trace, while the verifier operates on the generated trace and the original context. The final verification score is computed from the energy gap ΔE, not from a model's self-reported confidence.

M.2 Threshold and Hyperparameter Selection

The verification threshold τ_{verify} is selected on a validation split by maximizing VCAR subject to avoiding a large precision drop. After selection, the same threshold is used across datasets and reasoning backbones in the main comparison.

M.3 Reproduction Path

A complete reproduction runs the following stages: generate CoT traces from each backbone, segment traces into steps, route each step to the rigid or soft path, compute rigid or contrastive soft verification labels, aggregate step decisions into trace-level labels, and compute PR, Precision, VCAR, TA, and FAR.

N Additional Ablation Studies

This appendix extends the main-text ablations to additional reasoning backbones. The results use the same evaluation protocol as Table 1 and report trace-level verification metrics.

N.1 Ablations for Additional Backbones

Table 13: **Ablation study across reasoning backbones.** Each backbone block reports Pass Rate / Precision / VCAR averaged over ProofWriter, BioASQ, and LegalBench-SARA.

Variant	GPT-5.1	Claude 3.5	Qwen3-8B	Avg. VCAR
PROBVERI	48.8/80.1/**39.4**	46.4/76.0/**35.8**	41.7/71.4/**30.3**	**35.2**
w/o Soft Verification Path	30.9/89.3/27.8	28.8/88.0/25.7	23.6/83.8/20.3	24.6
w/o Negated-Energy Comparison	46.2/70.7/33.1	43.8/67.4/29.8	39.1/64.3/25.3	29.4
w/o Assumption Penalty	54.4/62.6/34.6	51.8/59.7/31.2	46.3/56.1/26.5	30.8
w/o Support-Grounded Weighting	45.1/70.5/32.4	42.5/67.2/28.8	38.4/61.6/23.8	28.3
Direct LLM Confidence as Weight	44.3/69.4/31.4	41.7/65.3/27.7	37.5/60.2/22.6	27.2
Soft-only PSL Verification	49.5/68.0/34.3	46.7/65.3/31.2	43.0/58.3/26.0	30.5

The same qualitative pattern holds across backbones. Removing the soft path collapses the method toward rigid verification and reduces VCAR. Removing the negated-energy comparison or support-grounded weighting also reduces recovery. The direct-LLM-confidence variant performs below the full model, supporting the design choice that rule strength should be derived from support and repair cost rather than from model self-confidence.

O Uncertainty-Stratified Evaluation

The main experiments evaluate all traces. This appendix focuses on uncertainty-heavy subsets, where rigid verification is expected to under-cover correct but non-deductive reasoning. Each row corresponds to a subset of steps or traces selected by the annotation criteria described below. We report VCAR and False Accept Rate (FAR), where lower FAR is better.

Table 14: **Uncertainty-stratified verification.** Each cell reports VCAR / FAR. Lower FAR is better.

Uncertainty Type	Count	Rigid	Soft-only PSL	PROBVERI
Explicit modal claims	147	31.7 / **3.1**	34.8 / 15.2	**39.6** / 8.7
Biomedical associations	183	13.5 / **3.4**	22.9 / 14.8	**29.4** / 8.9
Legal bridge assumptions	161	8.6 / **2.1**	17.5 / 14.5	**23.1** / 8.3
Weighted average	491	17.3 / **2.9**	24.7 / 14.8	**30.4** / 8.6

Rigid verification has the lowest false accept rate but misses many supported uncertain steps. Soft-only PSL recovers more steps but accepts too many unsupported claims. PROBVERI improves VCAR while keeping FAR substantially below Soft-only PSL. This supports the main claim that soft formalization alone is insufficient; contrastive comparison and assumption regularization are needed for reliable uncertainty-aware verification.

O.1 Annotation Guidelines

Explicit modal claims are identified by epistemic markers such as *may*, *likely*, *suggests*, *could indicate*, and *is associated with*. Biomedical association claims include biomarker, mechanism, risk-factor,

treatment-response, and correlation statements. Legal bridge assumptions are steps that require an applicability condition or statutory bridge between a factual premise and a legal conclusion.

O.2 Interpretation

The largest relative gains appear in biomedical associations and legal bridge assumptions. These categories are difficult for rigid verification because the relevant support is often evidential, conditional, or mediated by a bridge assumption rather than a strict entailment. The FAR reduction relative to Soft-only PSL shows that PROBVERI does not merely accept more uncertain steps; it filters them using contrastive energy comparison and assumption penalties.

P Failure Taxonomy

This appendix lists representative failure modes of uncertainty-aware verification and their possible causes.

Table 15: **Failure taxonomy.**

Failure Type	Description	Typical Cause
Wrong skeleton	Predicate extraction misses the intended relation	Ambiguous or domain-specific language
Missing bridge	Repair search fails to recover a needed assumption	Sparse context or long-range dependency
False soft accept	Soft path accepts a weak claim	Penalty too small or bridge too broad
False rigid reject	Rigid path rejects a plausible uncertain claim	Uncertainty not routed to soft path
Low-confidence tie	$E_{\mathrm{nat}} \approx E_{\mathrm{neg}}$	Evidence insufficient or balanced
Over-specific rule	Skeleton is too narrow to match evidence	Entity normalization or predicate mismatch
Contradiction missed	Evidence contradicts an assumption but is not retrieved	Incomplete context extraction

P.1 Error Interpretation

The most important residual failure is not that the soft path fails to assign a low natural energy, but that it sometimes cannot distinguish a genuinely supported weak inference from an underconstrained one. The natural-versus-negated comparison reduces this failure mode, but it does not eliminate it when evidence is sparse or contradictory evidence is not retrieved. This motivates the use of assumption penalties and contradiction filtering in the main method.

Q Prompt Templates and LLM Usage

This appendix documents the LLM usage in the experimental pipeline. LLMs are used to generate CoT traces, instantiate candidate symbolic structures, and implement neural baselines such as LLM-as-a-Judge. The final soft verification score is not a self-reported LLM confidence; it is computed from the PSL energy gap $\Delta E = E_{\mathrm{neg}} - E_{\mathrm{nat}}$.

Q.1 LLM Usage Summary

Table 16: **LLM usage in the pipeline.**

Component	LLM Role	Final Decision Source
CoT generation	Produce the initial reasoning trace from question and context	Backbone answer and trace
Uncertainty detection	Identify modal or weak-support markers	Router plus support-gap checks
Skeleton extraction	Propose predicate-level skeletons	Verified by rule construction and evidence matching
Bridge generation	Propose minimal candidate bridge assumptions	Filtered by support, contradiction, and cost
LLM-as-a-Judge baseline	Directly judge support status	Baseline label
Soft verification	No direct confidence assignment	PSL energy gap ΔE

Q.2 CoT Generation Prompt

CoT Generation Prompt

Given the question and context, produce a concise step-by-step reasoning trace before giving the final answer. Each reasoning step should be written as a separate sentence. If the evidence is uncertain, use appropriate epistemic language such as *may*, *suggests*, or *is associated with*; do not convert weak evidence into deterministic claims.

Q.3 Uncertainty Marker Detection Prompt

Uncertainty Detection Prompt

Given a reasoning step, identify whether it contains epistemic uncertainty, weak support, association, or conditional applicability. Return: (1) uncertainty markers, (2) whether the step should be treated as deterministic or uncertainty-marked, and (3) a short explanation.

Q.4 Predicate Skeleton Extraction Prompt

Skeleton Extraction Prompt

Given a reasoning step and its local context, extract a predicate-level skeleton. Preserve the relational structure of the claim but do not assign a confidence score. Return: predicate name, arguments, uncertainty markers, and candidate evidence atoms.

Q.5 Bridge Assumption Generation Prompt

Bridge Assumption Prompt

Given a local support context P_t and a target step s_t, list minimal bridge assumptions that would make the step supported. Each bridge must be grounded in the context or explicitly marked as unsupported. Do not invent domain facts that are absent from the context.

Q.6 Contradiction Filtering Prompt

Contradiction Filtering Prompt

Given a candidate bridge assumption and the local context, determine whether the bridge contradicts any stated evidence or previously verified step. Return `KEEP`, `WEAK`, or `REJECT`. Reject assumptions that introduce facts directly contradicted by the context.

Q.7 Rigid Formalization Prompt

Rigid Formalization Prompt

Translate deterministic reasoning steps into symbolic predicates and constraints. If the step contains epistemic uncertainty or cannot be faithfully represented as strict entailment, return `UNTRANSLATABLE` rather than forcing a hard predicate.

Q.8 PSL Rule Construction Prompt

PSL Rule Construction Prompt

Given a predicate skeleton, local support context, and bridge assumptions, construct weighted soft rules in PSL-style form. Return: (1) head predicate, (2) body predicates, (3) support source, and (4) whether the rule encodes direct support, partial support, or weak support. Do not assign a high weight unless the support is explicit or minimally repaired.

Q.9 LLM-as-a-Judge Baseline Prompt

LLM-as-a-Judge Prompt

Given the context, question, and reasoning step, decide whether the step is supported, contradicted, or unsupported. Return one of: `SUPPORTED`, `CONTRADICTED`, or `UNSUPPORTED`. Provide a one-sentence explanation. Do not use hidden assumptions that are not present in the context.

Q.10 Use of LLM Outputs

LLM outputs are treated as candidate structures rather than final verification decisions, except for the explicit LLM-as-a-Judge baseline. Candidate skeletons and bridges are checked through evidence matching, contradiction filtering, repair cost, and PSL inference. This separation is important because PROBVERI does not rely on opaque LLM confidence as a rule weight.

R Compute Resources and Runtime

This appendix reports approximate runtime and compute characteristics. Runtime varies with trace length, number of soft-routed steps, and the number of bridge candidates retained after filtering. The values below are intended to document the scale of the experiments rather than to claim system-level optimization.

R.1 Runtime per Reasoning Step

Table 17: **Approximate verification runtime per reasoning step.**

Path	Median Time	90th Percentile	Dominant Cost
Rigid path	0.05s	0.18s	symbolic translation and solver call
Soft path	0.43s	1.12s	bridge search and PSL inference
Full step average	0.20s	0.70s	routing mix dependent

R.2 Compute Summary

Table 18: **Compute and execution summary.**

Item	Description
Closed-source backbones	CoT traces are generated through API access under provider terms.
Open-weight backbone	Qwen3-8B traces are generated with local inference; checkpoint and hardware metadata are tracked as part of the experiment metadata.
Rigid verification	SMT-style checks are executed on CPU.
Soft verification	PSL-style optimization and bridge filtering are executed on CPU in the reported experiments.
Dominant cost	Soft-routed steps dominate verification cost because each requires natural and negated optimization.
Parallelization	Steps and traces can be parallelized because each trace-level verification instance is independent after CoT generation.

R.3 Scaling Considerations

Let T be the number of reasoning steps in a trace and T_s be the number of soft-routed steps. Rigid checks are inexpensive relative to soft inference. The dominant term is therefore the number of soft-routed steps times two PSL solves, one for E_{nat} and one for E_{neg}. Since the router sends directly formalizable steps to the rigid path, the practical cost depends on the proportion of uncertain or support-gap steps rather than on trace length alone.

S Broader Impact, Safeguards, and Responsible Use

PROBVERI is intended as a verification aid rather than a replacement for domain experts. In medical and legal settings, a confidence-scored verification result should not be interpreted as factual certainty or professional advice. The method can reduce unsupported reasoning by distinguishing hard entailment, weak support, contradiction, and uncertainty, but it remains dependent on the quality of context, predicate extraction, bridge assumptions, and calibration diagnostics.

S.1 Potential Benefits

The main potential benefit is improved process-level reliability. By separating unsupported steps from weakly but consistently supported inferences, the verifier can help identify reasoning traces that should not be trusted even when the final answer appears plausible. This is especially relevant in domains where intermediate reasoning contains modal, associative, or conditional language.

S.2 Risks and Limitations

The most important risk is over-trust. A high verification score indicates that the natural interpretation is more compatible with the available evidence than the negated interpretation under the constructed rules; it does not guarantee that the underlying evidence is complete or that the domain conclusion is correct. Errors may also arise from wrong predicate skeletons, missing contradiction evidence, overly broad bridge assumptions, or domain-specific ambiguity.

S.3 Safeguards

We recommend using the method as part of a human-in-the-loop verification workflow, especially in high-stakes settings. Verification outputs should be accompanied by the supporting evidence atoms, bridge assumptions, energy gap, and confidence score. Claims that require professional medical, legal, or safety judgment should remain subject to expert review.

S.4 Human Subjects and Sensitive Data

This work evaluates existing benchmark datasets and generated reasoning traces. It does not involve human-subject experiments, crowdsourcing, or collection of new personal data. The method is not intended to infer sensitive attributes of individuals. Any deployment in real medical or legal settings would require additional governance, privacy review, and domain-specific validation.

S.5 Responsible Release

If code or processed traces are released, they should include prompt templates, solver settings, dataset references, and instructions for reproducing aggregate metrics. Restricted benchmark data and closed-source model outputs should not be redistributed in violation of their original terms. The release should also document known limitations, including dependence on context quality, extraction quality, and bridge candidate coverage.

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: The abstract and introduction accurately outline our research questions and faithfully reflect the paper's contributions and scope.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: The paper discusses the main limitations of the proposed method, including its dependence on extraction quality, bridge assumptions, and domain-specific verification conditions.

   Guidelines:

   - The answer [N/A] 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 paper states the assumptions behind the formal objective, and the appendix provides derivations and proofs for the theoretical properties used in the method.

Guidelines:

- The answer [N/A] 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: The paper provides the experimental protocol, datasets, metrics, solver settings, and prompts needed to understand and reproduce the main reported results.

   Guidelines:

   - The answer [N/A] 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: [No]

Justification: A full public code release is not included at submission time. However, the paper provides detailed algorithmic descriptions, prompts, solver settings, and dataset protocols to support reproducibility.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

Answer: [Yes]

Justification: The experimental settings, including datasets, reasoning backbones, baselines, metrics, thresholds, and solver hyperparameters, are described in the main paper and appendix.

Guidelines:

- The answer [N/A] 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: The appendix provides additional details and discussions related to the main verification results.

Guidelines:

- The answer [N/A] 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: The appendix provides a summary of the compute setting and approximate runtime required for rigid and soft verification.

   Guidelines:

   - The answer [N/A] 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 uses existing benchmarks and generated reasoning traces, and we are not aware of any aspect that conflicts with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] 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: The paper discusses potential benefits and risks of uncertainty-aware verification, including reliability improvements and the risk of over-trust in high-stakes settings.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: The paper describes responsible-use considerations, including the need for human oversight and caution when applying verification outputs in medical, legal, or other high-stakes domains.

    Guidelines:

    - The answer [N/A] 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: The paper credits the datasets, models, and tools used in the experiments, and follows the corresponding release terms where applicable.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The paper does not introduce a new dataset, benchmark, or model asset.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The work does not involve crowdsourcing or human-subject experiments.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The work does not involve human-subject experiments, so IRB approval is not applicable.

    Guidelines:

- The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The paper describes the use of LLMs for generating reasoning traces and candidate structures, while the final verification score is computed by the proposed verification objective rather than by model-reported confidence.

    Guidelines:

    - The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.