LET SOLVERS REASON AND MODELS UNDERSTAND: PROOF-OBLIGATION-GUIDED BRIDGING

Anonymous authors
Paper under double-blind review

ABSTRACT

Neuro-symbolic approaches aim to combine the flexible language understanding of neural models with the rigorous reasoning of symbolic solvers. These approaches typically use a large language model (LLM) to translate the problem expressed in natural language (NL) into formal representations and a symbolic solver to reason over these representations. However, even when individual NL statements are faithfully formalized, their semantically related expressions may still be mapped to symbolically unrelated predicates, leading to failures in downstream symbolic reasoning. To overcome this challenge, we propose *Proof-Obligation-Guided Bridging* (POGB). During autoformalization, POGB maintains a grounding table that records the NL semantic grounding for each formal element. When the solver encounters an unresolved proof obligation, POGB identifies candidate semantic bridges and uses an LLM to evaluate them against the recorded grounding information, thereby recovering semantic relations missing from the formal representations. This design keeps all logical reasoning under the solver's control while restricting the LLM's role to autoformalization and the evaluation of candidate semantic bridges. By letting solvers reason and models understand, POGB leverages their complementary strengths to advance neuro-symbolic integration. Theoretical analysis proves the soundness of POGB, and extensive experiments on diverse logical reasoning benchmarks demonstrate its superior performance.

1 INTRODUCTION

Large language models (LLMs) are advancing rapidly and have demonstrated strong capabilities in understanding complex natural language (NL) (Wang et al., 2024) and exploiting broad commonsense knowledge (Bian et al., 2024). However, since their outputs are generated through probabilistic prediction, their reasoning is not guaranteed to be logically valid: they may fabricate unsupported premises (Qiu et al., 2026), confuse cause and effect (Holliday et al., 2024), or engage in circular arguments (Mouchel et al., 2025) while still producing seemingly plausible answers. These limitations restrict their applicability in settings that require high levels of accuracy and reliability.

Formal methods (FMs), powered by symbolic solvers, enable rigorous reasoning and provide a promising way to mitigate these limitations (Weyhrauch, 1980; Woodcock et al., 2009; Pan et al., 2023; Xu et al., 2024). Therefore, neuro-symbolic systems that integrate LLMs with FMs have attracted increasing research attention (Chandre et al., 2025; Yang et al., 2025; Belle & Marcus, 2026). In a typical neuro-symbolic workflow, a neural model (e.g., an LLM) automatically translates the problem expressed in NL into formal representations, a process known as *autoformalization*, after which a symbolic solver reasons over these representations. Although this workflow makes the reasoning process machine-verifiable and the resulting conclusion mathematically provable, its correctness guarantees apply only to the translated formal representations, not to the original NL problem (Kneuper, 1997; Pan et al., 2023; Olausson et al., 2023). Consequently, the main challenge shifts from validating LLM-generated reasoning to ensuring semantic consistency during the autoformalization process (Olausson et al., 2023; Ryu et al., 2025; Chen et al., 2026).

Specifically, an autoformalized result may be syntactically correct and semantically plausible, yet still fail to retain the semantic relations required for proof composition, leading to failures in downstream symbolic reasoning. A major source of such failures is *variation in linguistic form* and *implicit semantic content in NL* (Gerber & Chai, 2012; Bhagat & Hovy, 2013; Moore & Shah,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

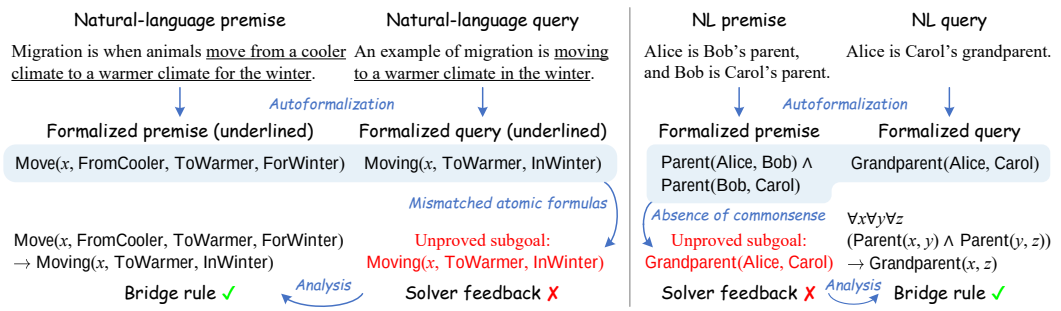

Figure 1: Examples of major challenges in faithful autoformalization and our proposed strategy.

2026). Consider the left-hand example in Figure 1. The NL premise uses the verb "move," whereas the NL query uses its gerund form "moving;" similarly, the NL premise uses the phrase "for the winter," whereas the NL query uses "in the winter." Although these expressions convey equivalent meanings in this context, their different linguistic forms lead to distinct predicate symbols, "Move" and "Moving," as well as distinct constants, "ForWinter" and "InWinter," in the formalized results. Moreover, the phrase "to a warmer climate" implies that the movement originates in a cooler climate. This information is explicitly stated in the NL premise but implicit in the NL query, leading to the additional argument term "FromCooler" in the formalized premise. Consequently, even faithful autoformalization may yield mismatched atomic formulas for the premise and the query: $\mathsf{Move}(x, \mathsf{FromCooler}, \mathsf{ToWarmer}, \mathsf{ForWinter})$ and $\mathsf{Moving}(x, \mathsf{ToWarmer}, \mathsf{InWinter})$. Symbolic solvers therefore treat them as distinct formulas, even though they encode equivalent meanings in the given context. Another important source of failures is the absence of commonsense knowledge about the physical and social world (Davis, 2017; Toroghi et al., 2024; Teo et al., 2026). Consider the right-hand example in Figure 1. Although humans readily recognize the basic commonsense fact that "a parent's parent is a grandparent," symbolic solvers treat "Parent" and "Grandparent" as unrelated predicate symbols unless this relationship is explicitly encoded as a symbolic rule or axiom. Consequently, solvers are unable to complete the intended symbolic reasoning.

To overcome these challenges, we propose *Proof-Obligation-Guided Bridging* (POGB), a method for recovering missing semantic relations in the formal representations generated by autoformalization, without requiring LLMs to perform potentially unreliable multi-step reasoning. Specifically, we construct a grounding table that records each formal element with its corresponding NL semantic grounding and makes this information accessible throughout the reasoning process. We then conduct a *proof-obligation-guided search* to identify the missing semantic relations required for the proof to proceed. The resulting candidate semantic bridges are evaluated by the LLM based on the recorded NL semantic groundings. In this way, LLM involvement is restricted to initial autoformalization and the subsequent evaluation of candidate semantic bridges, while symbolic solvers perform the reasoning (including quantifier instantiation, logical decomposition, recursive discharge, and proof construction) and final verification. For the examples shown in Figure 1, the proposed method therefore establishes the following semantic bridges: $\mathsf{Move}(x, \mathsf{FromCooler}, \mathsf{ToWarmer}, \mathsf{ForWinter}) \rightarrow \mathsf{Moving}(x, \mathsf{ToWarmer}, \mathsf{InWinter})$ and $\forall x \forall y \forall z.\, (\mathsf{Parent}(x, y) \wedge \mathsf{Parent}(y, z)) \rightarrow \mathsf{Grandparent}(x, z)$.

Overall, the key innovations and main contributions of our research can be summarized as follows:

- We introduce a grounding table that records the NL semantic grounding for each formal element during autoformalization. This table provides coarse-grained type information to constrain which instantiations are admissible and fine-grained NL information to guide the recovery of semantic relations missing from the formal representations when exact symbolic matching fails.
- We propose proof-obligation-guided bridging to identify semantic bridges when the symbolic solver encounters an unresolved proof obligation. The method decouples semantic understanding from logical reasoning by assigning the former to the neural model and the latter to the symbolic solver. Furthermore, we prove the method's soundness through theoretical analysis.
- Extensive experiments across diverse logical reasoning benchmarks demonstrate that our proposed method significantly improves end-to-end reasoning accuracy while substantially enhancing the logical validity and machine verifiability of the reasoning process.

2 Related Work

Neuro-symbolic approaches aim to combine the flexible language understanding of neural models with the rigorous reasoning of symbolic solvers, leveraging the complementary strengths of both paradigms (Li et al., 2026a). To improve the reasoning capabilities of LLMs, a series of studies integrates symbolic expressions and logical rules into chain-of-thought (CoT) prompting (Xu et al., 2024) and, more broadly, throughout the LLM reasoning process (Toroghi et al., 2024; Xu et al., 2025). Recent studies also leverage feedback from formal verification to guide model training (Xu et al., 2026; Cao et al., 2026). Complementary to these LLM-centric reasoning methods, solver-centric reasoning methods (Lyu et al., 2023; Pan et al., 2023; Olausson et al., 2023) typically adopt a *translate-then-solve* architecture: an LLM first translates the problem expressed in NL into formal representations, and then a symbolic solver reasons over these representations. This architecture makes the reasoning process logically valid and machine-verifiable, which is particularly valuable for complex problems. However, it shifts much of the burden to the translation fidelity. Consequently, overall performance depends largely on the quality of the generated formal representations.

Automatic translation of NL into formal representations is commonly referred to as *autoformalization*. Research in this area has traditionally focused on formal mathematics (Wang et al., 2018; Szegedy, 2020; Wu et al., 2022) and has recently expanded to a broader range of tasks (Mensfelt et al., 2026). Among these tasks, logical reasoning tasks are particularly challenging to autoformalize because they involve a wide variety of linguistic expressions and often require contextual understanding and real-world knowledge (Pan et al., 2023; Lalwani et al., 2025). Consequently, their resulting formal representations are particularly likely to contain lexical, syntactic, and semantic errors. Existing research has addressed these issues by incorporating feedback from solver error messages (Pan et al., 2023), guiding generation with abstract syntax trees (ASTs) (Putra et al., 2026), and performing semantic verification (Ryu et al., 2025; Raza & Milic-Frayling, 2025). Although these methods substantially improve executability, syntactic validity, and semantic faithfulness, they may still fail to ensure the global semantic consistency of the generated formal representations.

Specifically, two NL statements may each be translated into a locally plausible formal representation, while their semantically related expressions may still be mapped to symbolically unrelated predicates, as shown in Figure 1. Because the symbolic solver cannot access the NL semantics underlying these predicates, relevant context, or commonsense knowledge, it cannot infer the intended semantic relations from the formal representations alone and encode them explicitly as additional symbolic rules or axioms. As a result, proof search cannot link these predicates and fails when the target derivation requires interactions between them, even if every statement has been faithfully formalized in isolation. Recent advances attempt to mitigate this problem through two types of methods. Premise-enhanced methods (Hu et al., 2025; Callewaert et al., 2025; Li et al., 2026b) use LLM-based reasoning or retrieval augmentation to complete missing semantic relations and align formal representations *before proof search begins*. However, because these modifications are made without feedback from the proof process, such methods cannot reliably anticipate which missing relations will obstruct a particular derivation in the proof. By contrast, proof- or verification-guided methods (Lee & Hwang, 2025; Quan et al., 2025; 2026; Singh et al., 2026) use solver feedback *after failed proof attempts*. However, they typically rely on an LLM to repair these failures, potentially introducing counterfactual assumptions or modifications not warranted by the original premises. In summary, both pre-proof completion and post-proof repair have inherent limitations; therefore, we propose a real-time bridging approach that operates during proof, which we term **proof-time bridging**.

3 Methodology

In this section, we present our proposed approach, Proof-Obligation-Guided Bridging (POGB), as illustrated in Figure 2. We begin with a grounding autoformalization step, in which an LLM translates the problem expressed in NL into formal representations while maintaining a grounding table that records the type and associated NL information for each formal element. This table then serves as a *proof-time interface* for *proof-obligation-guided search*. During this search, type information constrains which instantiations are admissible, while NL information guides the identification of semantic bridges via an LLM-based `Gate` when symbolic reasoning encounters an unresolved proof obligation. This design keeps all logical reasoning under the solver's control while restricting the LLM's role to autoformalization and the evaluation of candidate semantic bridges.

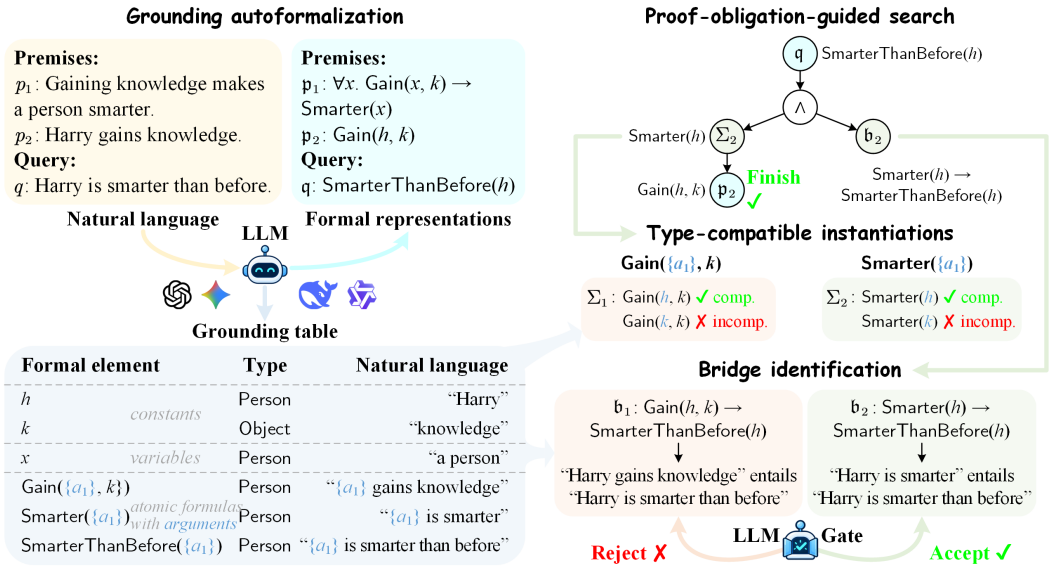

Figure 2: Illustration of the proposed approach. Variables in the atomic formulas of the grounding table are replaced with argument placeholders $\{a_i\}$, which are highlighted in blue and subsequently instantiated with type-compatible constants.

3.1 Problem Formulation

Let $X = (\mathbb{P}, q)$ denote a problem comprising a set of NL premises $\mathbb{P} = \{p_i\}_{i=1}^n$ and an NL query q. We use a grounding autoformalization F to map X to $F(X) = (\mathcal{P}, \mathfrak{q}, \mathcal{T})$, thereby yielding formal representations that are explicitly associated with their NL semantics in X. Here, $\mathcal{P} = \{\mathfrak{p}_i\}_{i=1}^n$ is the set of formalized premises, where $\mathfrak{p}_i$ denotes the ith formalized premise; $\mathfrak{q}$ is the formalized query; and $\mathcal{T}$ is a grounding table that records the NL semantic grounding of every formal element (see Section 3.2 for details). We adopt first-order logic (FOL) as the target formal language.

Consider two proof goals corresponding to the query and its negation: $\mathfrak{q}$ and $\neg\mathfrak{q}$. For each goal $\mathfrak{g} \in \{\mathfrak{q}, \neg\mathfrak{q}\}$, the symbolic solver performs a proof search and identifies unresolved proof obligations resulting from missing semantic relations. Let $\mathcal{S}$ denote the symbolic solver. Our objective is to find a *reasonable* bridge set $\mathcal{B}$ that enables the proof to be completed:

$$\mathcal{S}(\mathcal{P} \cup \mathcal{B}, \mathfrak{g}) = \textsc{Proved} \quad \wedge \quad \mathrm{Gate}(\mathcal{B}, \mathcal{T}) = \textsc{Accepted}. \tag{1}$$

Here, $\mathcal{B} = \{\mathfrak{b}_j\}_{j=1}^m$, where each $\mathfrak{b}_j$ is a semantic bridge that explicitly encodes an entailment relation supported by the NL semantics but not represented in the symbolic ontology. The LLM-based Gate evaluates only whether these candidate semantic bridges are reasonable given the grounding table $\mathcal{T}$; it does not perform proof search, conduct intermediate reasoning, or repair failures in response to solver feedback. The final decision is determined by the proof outcomes for the two goals:

$$\mathrm{Decide}(X) = \begin{cases} \textsc{Entailed}, & \big(\mathcal{S}(\mathcal{P} \cup \mathcal{B}, \mathfrak{q}),\ \mathcal{S}(\mathcal{P} \cup \mathcal{B}, \neg\mathfrak{q})\big) = \big(\textsc{Proved},\ \textsc{Failed}\big), \\ \textsc{Contradicted}, & \big(\mathcal{S}(\mathcal{P} \cup \mathcal{B}, \mathfrak{q}),\ \mathcal{S}(\mathcal{P} \cup \mathcal{B}, \neg\mathfrak{q})\big) = \big(\textsc{Failed},\ \textsc{Proved}\big), \\ \textsc{Unknown}, & \big(\mathcal{S}(\mathcal{P} \cup \mathcal{B}, \mathfrak{q}),\ \mathcal{S}(\mathcal{P} \cup \mathcal{B}, \neg\mathfrak{q})\big) = \big(\textsc{Failed},\ \textsc{Failed}\big), \\ \textsc{Conflicted}, & \big(\mathcal{S}(\mathcal{P} \cup \mathcal{B}, \mathfrak{q}),\ \mathcal{S}(\mathcal{P} \cup \mathcal{B}, \neg\mathfrak{q})\big) = \big(\textsc{Proved},\ \textsc{Proved}\big). \end{cases}$$

3.2 The Grounding Table as a Proof-Time Interface

As discussed in Sections 1 and 2, even when individual NL statements are faithfully formalized, their semantically related expressions may still be mapped to symbolically unrelated predicates. To recover the semantic relations between such predicates during proof search, we maintain two complementary kinds of information for every formal element. Let $\mathfrak{e}$ denote a formal element occurring in $\mathcal{P}$ or $\mathfrak{q}$, including a constant, a variable, or an atomic formula. For each $\mathfrak{e}$, the grounding table $\mathcal{T}$ records both type information, $\mathrm{type}_{\mathcal{T}}(\mathfrak{e})$, and NL information, $\mathrm{NL}_{\mathcal{T}}(\mathfrak{e})$, thereby *providing an interface for proof-time inference*.

Algorithm 1 Proof-obligation-guided search

Input: A set of formalized premises $\mathcal{P}$, a symbolic solver $\mathcal{S}$, a proof obligation ϕ, a set of ancestor obligations $\mathcal{A}$, and a grounding table $\mathcal{T}$.
Output: A set of accepted bridges sufficient to prove ϕ from $\mathcal{P}$, or FAILED if the search fails.

function PROVE($\phi, \mathcal{A}, \mathcal{T}$)
  if $\mathcal{S}(\mathcal{P}, \phi) =$ PROVED **then** ▷ Discharge ϕ directly with the symbolic solver
    return $\varnothing$
  $\phi \leftarrow$ NORMALIZE(ϕ) ▷ Rewrite ϕ in canonical form
  if CYCLIC($\phi, \mathcal{A}$) **then** ▷ Reject circular reasoning
    return FAILED
  $\mathcal{A}' \leftarrow \mathcal{A} \cup \{\phi\}$ ▷ Record the current proof obligation to check the circular reasoning
  if $\phi = \exists x.\, \psi$ **then** ▷ Branch over type-compatible instantiations of x
    return any successful result of PROVE($\psi', \mathcal{A}', \mathcal{T}$) with $\psi' \in$ INST($\psi, \mathcal{T}$)
  else if $\phi = \forall x.\, \psi$ **then** ▷ Introduce a fresh constant α and prove ψ after mapping variable x to α
    return PROVE($\psi[x \mapsto \alpha], \mathcal{A}', \mathcal{T} \cup \{(\alpha, \mathrm{type}_{\mathcal{T}}(x), \mathrm{NL}_{\mathcal{T}}(x))\}$)
  else if $\phi = \phi_1 \wedge \phi_2$ **then** ▷ Discharge both sub-obligations
    return PROVE($\phi_1, \mathcal{A}', \mathcal{T}$) $\cup$ PROVE($\phi_2, \mathcal{A}', \mathcal{T}$)
  else if $\phi = \phi_1 \vee \phi_2$ **then** ▷ Discharge at least one sub-obligation
    return the successful result of PROVE($\phi_1, \mathcal{A}', \mathcal{T}$) or PROVE($\phi_2, \mathcal{A}', \mathcal{T}$)
  for each $(\mathcal{B}, \mathfrak{o})$ yielded by IDENTIFY($\phi, \mathcal{P}, \mathcal{T}$) **do** ▷ IDENTIFY($\phi, \mathcal{P}, \mathcal{T}$) is given in Algorithm 2
    $\mathcal{B}_{\mathfrak{o}} \leftarrow$ PROVE($\mathfrak{o}, \mathcal{A}', \mathcal{T}$)
    if $\mathcal{B}_{\mathfrak{o}} \neq$ FAILED **then** ▷ The residual proof obligation has been discharged
      return $\mathcal{B} \cup \mathcal{B}_{\mathfrak{o}}$ ▷ Combine the accepted bridges
  return FAILED ▷ The search fails

Inspired by type systems (Cardelli, 1996), $\mathrm{type}_{\mathcal{T}}(\mathfrak{e})$ retains a coarse-grained semantic category assigned to $\mathfrak{e}$ by the LLM. This category constrains which instantiations are admissible, thereby narrowing the search space for candidate semantic bridges. For instance, in the running example in Appendix A, a variable of type Person can be instantiated with h (Harry) but not with w (*Walden*). By contrast, $\mathrm{NL}_{\mathcal{T}}(\mathfrak{e})$ retains the finer-grained NL semantics of $\mathfrak{e}$ by storing referring expressions (e.g., "Harry," "*Walden*," "a person," and "a book") and predicate meanings (e.g., "$\{a_1\}$ reads $\{a_2\}$"). When exact symbolic matching fails, this information guides the search for plausible semantic bridges between *type-compatible* candidates, such as the bridge between $\mathsf{Smarter}(h)$ and $\mathsf{SmarterThanBefore}(h)$. We next show how this interface supports *proof-obligation-guided search*.

3.3 PROOF-OBLIGATION-GUIDED SEARCH

We propose a proof-search method controlled by a symbolic solver and guided by proof obligations, as described in Algorithms 1 and 2. Algorithm 1 recursively decomposes each obligation through symbolic reasoning. When symbolic reasoning alone is insufficient to discharge an obligation, Algorithm 1 invokes Algorithm 2 to identify local semantic bridges. Algorithm 2 then returns the identified bridges together with the residual obligations required to establish each bridge, after which Algorithm 1 recursively discharges these residual obligations. A running example and a range of case studies are presented in Appendices A and B, respectively.

3.3.1 SOLVER-CONTROLLED RECURSION

Algorithm 1 defines a recursive function, PROVE($\phi, \mathcal{A}, \mathcal{T}$), that performs proof-obligation-guided search. Here, ϕ denotes the current proof obligation and is initially set to $\mathfrak{g}$; $\mathcal{A}$ denotes the set of ancestor obligations to detect circular reasoning and is initially empty; and $\mathcal{T}$ is the grounding table. The solver first attempts to discharge ϕ directly through symbolic reasoning. If this attempt fails, it normalizes ϕ and distinguishes cases based on the top-level logical structure of ϕ.

For an existential obligation $\phi = \exists x.\, \psi$, the solver branches over the type-compatible instantiations of x generated by the operator INST($\psi, \mathcal{T}$), where x may denote a set of variables. More precisely, for a formula ψ containing variables $x_1, \ldots, x_n$,

$$\textsc{Inst}(\psi, \mathcal{T}) = \{\psi[\theta] \mid \theta = \{x_i \mapsto c_i\}_{i=1}^{n} \text{ and } \mathrm{type}_{\mathcal{T}}(c_i) \preceq \mathrm{type}_{\mathcal{T}}(x_i) \text{ for all } i\}\,.$$

Here, θ is a substitution that maps each variable x_i to a constant c_i; $\psi[\theta]$ denotes the result of applying θ to ψ; and $\tau_1 \preceq \tau_2$ denotes that τ_1 is a subtype of τ_2, such as $\mathsf{Student} \preceq \mathsf{Person}$. Type informa-

Algorithm 2 Semantic bridge identification

Input: An unresolved obligation ϕ, a set of formalized premises $\mathcal{P}$, a grounding table $\mathcal{T}$, and a threshold t.
Output: A set of accepted semantic bridge (possibly FAILED) and its corresponding residual obligation.

function IDENTIFY($\phi, \mathcal{P}, \mathcal{T}$)
  $\mathcal{Q} \leftarrow \varnothing$ ▷ Initialize the candidate pool
  for all $\mathcal{I} \subseteq \bigcup_{\mathfrak{p}\in\mathcal{P}} \textsc{Inst}(\mathfrak{p}, \mathcal{T})$ **do** ▷ Enumerate subsets of all type-compatible instantiations
    for all $S \subseteq \bigcup_{\mathfrak{i}\in\mathcal{I}} \textsc{Lit}(\mathfrak{i})$ with $|S| \leq 2$ **do** ▷ Enumerate support sets comprising at most two literals
      $\Sigma \leftarrow \bigwedge_{s\in S} s$ ▷ Construct the candidate semantic support
      $\mathfrak{o} \leftarrow \textsc{Abduce}(\mathcal{I}, \Sigma)$ ▷ Derive the residual proof obligation
      $\rho \leftarrow \mathrm{sim}(\mathrm{NL}_{\mathcal{T}}(\Sigma), \mathrm{NL}_{\mathcal{T}}(\phi))$ ▷ Compute semantic similarity
      $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\Sigma, \mathfrak{o}, \rho)\}$
  for $(\Sigma, \mathfrak{o}, \rho) \in \mathcal{Q}$ in nonincreasing order of ρ **do** ▷ Evaluate candidates in ranked order
    $\mathcal{B} \leftarrow \varnothing$ if $\Sigma = \phi$; otherwise $\mathcal{B} \leftarrow \{\Sigma \rightarrow \phi\}$
    if $\mathcal{B} = \varnothing$ **or** $\mathrm{NLI}(\mathrm{NL}_{\mathcal{T}}(\Sigma), \mathrm{NL}_{\mathcal{T}}(\phi)) \geq t$ **then** ▷ Perform NLI filtering
      if $\mathrm{Gate}(\mathcal{B}, \mathcal{T}) = \textsc{Accepted}$ **then** ▷ Evaluate the bridge using the LLM-based Gate
        yield $(\mathcal{B}, \mathfrak{o})$

tion thus rules out type-incompatible instantiations before bridge identification begins. For a universal obligation $\phi = \forall x.\, \psi$, the solver introduces a fresh constant α representing an arbitrary value of x and then proves $\psi[x \mapsto \alpha]$, following the standard eigenvariable technique (Lincoln & Shankar, 1994). For a conjunctive obligation $\phi = \phi_1 \wedge \phi_2$, both sub-obligations must be discharged. In contrast, for a disjunctive obligation $\phi = \phi_1 \vee \phi_2$, it suffices to discharge at least one sub-obligation.

3.3.2 Semantic Bridge Identification

If ϕ remains unresolved after the above process, Algorithm 2 is invoked to identify bridges that can provide semantic support for ϕ. The key idea is to address two questions separately: *which semantic support can bridge the gap to the current proof obligation*, and *what residual obligation must still be discharged to establish that support*. The former is addressed by LLM-based evaluation, whereas the latter is addressed by symbolic reasoning.

Specifically, for each formalized premise $\mathfrak{p} \in \mathcal{P}$, the operator $\textsc{Inst}(\mathfrak{p}, \mathcal{T})$ first generates type-compatible instantiations. For each instantiation $\mathfrak{i}$, let $\textsc{Lit}(\mathfrak{i})$ denote the set of literals it contains, where a literal is an atomic formula or its negation. For example, the instantiation $\mathsf{Gain}(h, k) \rightarrow \mathsf{Smarter}(h)$ is equivalent to $\neg\mathsf{Gain}(h, k) \vee \mathsf{Smarter}(h)$, which contains the literals $\neg\mathsf{Gain}(h, k)$ and $\mathsf{Smarter}(h)$. Algorithm 2 then considers a subset $\mathcal{I}$ of all type-compatible instantiations and selects a support set $S \subseteq \bigcup_{\mathfrak{i}\in\mathcal{I}} \textsc{Lit}(\mathfrak{i})$, whose literals may come from multiple premises and may jointly support the current obligation ϕ. For computational tractability, S is restricted to at most two literals. The literals in S are conjoined to form a candidate *semantic support* $\Sigma = \bigwedge_{s\in S} s$, yielding a candidate *semantic bridge* $\Sigma \rightarrow \phi$. For example, when $\phi = \mathsf{SmarterThanBefore}(h)$, selecting $S = \{\mathsf{Smarter}(h)\}$ yields $\Sigma = \mathsf{Smarter}(h)$ and the corresponding bridge $\mathsf{Smarter}(h) \rightarrow \mathsf{SmarterThanBefore}(h)$. Next, candidate semantic bridges are filtered using a lightweight semantic similarity model and a natural language inference (NLI) model, both of which take $\mathrm{NL}_{\mathcal{T}}(\Sigma)$ and $\mathrm{NL}_{\mathcal{T}}(\phi)$ as inputs. Finally, the LLM-based Gate evaluates whether each remaining bridge holds.

Even after a bridge is accepted by Gate, its support Σ may remain unresolved. We therefore use logical abduction (Josephson & Josephson, 1996) to derive a residual proof obligation $\mathfrak{o}$ that, together with the instantiations in $\mathcal{I}$, is sufficient to establish Σ:

$$\mathfrak{o} = \textsc{Abduce}(\mathcal{I}, \Sigma), \qquad \left(\bigwedge\nolimits_{\mathfrak{i}\in\mathcal{I}} \mathfrak{i}\right) \wedge \mathfrak{o} \models \Sigma.$$

The residual obligation $\mathfrak{o}$ is then passed to Algorithm 1 for recursive discharge. Thus, Algorithm 2 decouples semantic understanding from logical reasoning: neural models evaluate only the candidate semantic bridge $\Sigma \rightarrow \phi$, whereas the residual obligation required to establish Σ is derived symbolically through logical abduction and must still be discharged by the symbolic solver.

3.3.3 Termination and Soundness

When the search succeeds, Algorithm 1 returns a bridge set that satisfies both criteria in Equation (1): the set enables the solver to complete the proof of goal $\mathfrak{g}$, and every bridge in the set is accepted by

Gate. Theoretical proof of this soundness guarantee is provided in Appendix C. A search branch fails if the current obligation already appears on its ancestor path, if all candidate bridges are rejected, or if a search limit is reached. The ancestor check aims to prevent circular reasoning (Bol et al., 1991): if an attempt to prove ϕ generates ψ and a subsequent attempt to prove ψ regenerates ϕ, the repeated obligation is rejected because it cannot serve as its own justification.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Datasets. We evaluate the proposed method, POGB, on five logical reasoning benchmarks: FOLIO (Han et al., 2024), ProofWriter (Tafjord et al., 2021), PrOntoQA (Saparov & He, 2023), ProverQA (Qi et al., 2025), and EntailmentBank (Dalvi et al., 2021). For ProverQA, we use only the *hard* subset. These benchmarks cover both template-constrained synthetic language and linguistically diverse, human-written NL. They also vary in their reliance on implicit commonsense knowledge and in proof complexity. Additional dataset details are provided in Appendix D. We report end-to-end reasoning accuracy on the first four benchmarks in Table 1 and the verified proof rate on all benchmarks in Table 2, with corresponding analyses in Sections 4.2 and 4.3, respectively.

Models and baselines. We conduct experiments using four LLMs: Gemini3-Flash (Google DeepMind, 2025), DeepSeek-V3.2 (Liu et al., 2025), Qwen3.6-Flash (Qwen Team, 2026), and GPT-5.5 (OpenAI, 2026). For each LLM, we compare POGB with two standard prompting methods—Direct Prediction (Direct) and Chain-of-Thought (CoT) (Wei et al., 2022)—and three representative neuro-symbolic methods: Logical Inference via Neurosymbolic Computation (LINC) (Olausson et al., 2023), Logic-LM (Pan et al., 2023), and Symbolic CoT (SymbCoT) (Xu et al., 2024).

Settings. For each baseline, we follow its standard prompting and inference protocols, including repeated sampling, iterative refinement, and fallback mechanisms. Specifically, LINC samples three formalizations and aggregates the corresponding solver outputs by voting; Logic-LM performs up to three rounds of self-refinement and uses a CoT fallback mechanism; and SymbCoT executes the complete translator–planner–solver–verifier pipeline. For POGB, we sample up to three formalizations, terminating early once a valid formal proof is found and falling back to CoT otherwise. For semantic bridge identification, we use `all-MiniLM-L6-v2` (Reimers & Gurevych, 2019) to compute semantic similarity and `roberta-large-mnli` (Liu et al., 2019) for NLI. The NLI entailment threshold t is set to 0.80 across all datasets except EntailmentBank, for which we use 0.75.

4.2 MAIN RESULTS

As shown in Table 1, POGB achieves average accuracies of 95.68%, 94.57%, 93.63%, and 95.90% using Gemini3-Flash, DeepSeek-V3.2, Qwen3.6-Flash, and GPT-5.5 as the underlying LLMs, respectively. These results correspond to improvements of 5.38, 5.88, 3.94, and 5.77 percentage points over the strongest baseline for each LLM. Moreover, POGB attains the highest accuracy across *all 16 LLM–dataset combinations*, demonstrating its effectiveness across diverse models and datasets.

The performance gains are particularly significant on benchmarks characterized by great linguistic diversity or complex proof composition. On FOLIO, which comprises linguistically diverse, human-written instances, POGB outperforms the strongest baseline using the same LLM by up to 5.89 percentage points. On ProverQA, which requires deeper and more structurally complex deductive reasoning, the corresponding improvement reaches up to 18.40 percentage points. On template-constrained synthetic benchmarks such as ProofWriter and PrOntoQA, where competitive baselines already approach the accuracy ceiling, POGB achieves near-perfect or perfect accuracy. Appendix E provides a detailed failure analysis covering a diverse range of representative error modes.

4.3 EVALUATION OF MACHINE-VERIFIABLE FORMAL REASONING

A correct final prediction does not necessarily imply a valid reasoning process, even for methods that achieve high end-to-end reasoning accuracy (Chen et al., 2025; Arcuschin et al., 2026). Therefore, we further evaluate whether each method can generate a machine-verifiable formal proof that supports the ground-truth label.

Table 1: End-to-end reasoning accuracy (%). The evaluation set size (N) for each dataset is shown below its column header.

LLM	Method	FOLIO $N = 204$	ProofWriter $N = 600$	PrOntoQA $N = 500$	ProverQA $N = 500$	Average
Gemini3-Flash	Direct	69.61	77.50	95.80	65.00	78.22
	CoT	83.82	95.33	**100.00**	74.80	89.63
	LINC	73.04	98.33	**100.00**	64.60	86.59
	Logic-LM	82.35	99.67	99.60	73.00	90.30
	SymbCoT	79.90	93.83	**100.00**	77.40	89.41
	POGB (Ours)	**88.73**	**100.00**	**100.00**	**89.00**	**95.68**
DeepSeek-V3.2	Direct	65.69	57.67	68.20	58.80	61.81
	CoT	84.80	91.67	**100.00**	69.40	87.03
	LINC	69.12	98.50	**100.00**	67.00	86.86
	Logic-LM	78.43	93.00	93.20	66.80	84.15
	SymbCoT	75.49	90.83	98.40	81.80	88.69
	POGB (Ours)	**90.69**	**99.83**	**100.00**	**84.40**	**94.57**
Qwen3.6-Flash	Direct	85.29	93.33	**100.00**	76.80	89.69
	CoT	82.84	94.33	**100.00**	74.60	89.14
	LINC	73.04	98.33	99.40	64.40	86.36
	Logic-LM	84.80	96.17	95.80	74.00	88.64
	SymbCoT	81.37	93.00	89.00	80.00	86.97
	POGB (Ours)	**87.75**	**99.17**	**100.00**	**83.00**	**93.63**
GPT-5.5	Direct	77.45	98.67	**100.00**	67.80	88.08
	CoT	79.41	98.50	**100.00**	66.40	87.86
	LINC	77.45	98.50	**100.00**	70.40	88.75
	Logic-LM	85.78	**100.00**	**100.00**	70.20	90.13
	SymbCoT	81.86	94.67	**100.00**	71.60	88.30
	POGB (Ours)	**88.24**	**100.00**	**100.00**	**90.00**	**95.90**

Table 2: VPR (%) with Gemini3-Flash as the underlying LLM. The evaluation set size (N) for each dataset is shown below its column header.

LLM	Method	FOLIO $N = 136$	ProofWriter $N = 400$	PrOntoQA $N = 500$	ProverQA $N = 326$	EntailmentBank $N = 340$	Average
Gemini3-Flash	LINC	63.24	**100.00**	**100.00**	51.53	39.41	75.68
	Logic-LM	62.50	99.50	99.60	57.06	30.59	74.68
	POGB (Ours)	**76.47**	**100.00**	**100.00**	**75.15**	**57.65**	**84.90**

Specifically, we re-evaluate the methods on non-UNKNOWN instances from the four benchmarks in Table 1 and include EntailmentBank (Dalvi et al., 2021) as an additional stress test. EntailmentBank is particularly challenging because many semantic relations required to construct its entailment trees are implicit. Moreover, all of its queries are entailed and thus labeled TRUE, so answer accuracy alone cannot guarantee the validity of the underlying reasoning. We compare LINC, Logic-LM, and POGB with all CoT fallbacks disabled. Direct, CoT, and SymbCoT are excluded because their LLM-generated reasoning traces cannot be formally verified. Our evaluation metric is the verified proof rate (VPR), defined as the percentage of instances for which the generated formal proof both passes machine verification and supports the ground-truth label.

As shown in Table 2, POGB achieves an average VPR of 84.90% using Gemini3-Flash as the underlying LLM, exceeding the strongest baseline by 9.22 percentage points. The performance gains are particularly significant on FOLIO, ProverQA, and EntailmentBank, where POGB outperforms the strongest baseline by 13.23, 18.09, and 18.24 percentage points, respectively. In addition, POGB achieves a perfect VPR of 100% on both ProofWriter and PrOntoQA. These results further demonstrate that POGB substantially enhances the logical validity and machine verifiability of the reasoning process across diverse benchmarks. Results for the remaining LLMs are reported in Appendix F.

Table 3: Ablation results (accuracy, %) for POGB with Gemini3-Flash as the underlying LLM.

Method variant	FOLIO	ProverQA
FOL (w/o GRD, BDG, or CoT-FB)	69.61	71.80
GRD + LLM-CMP (BDG replaced; w/o CoT-FB)	70.59	71.40
FOL + BDG (w/o GRD or CoT-FB)	73.53	72.40
GRD + BDG (w/o CoT-FB)	**81.86**	**82.60**
CoT	83.82	74.80
POGB (Ours; GRD + BDG + CoT-FB)	**88.73**	**89.00**

4.4 Ablation Studies

POGB contains three main components: the grounding table (GRD), the semantic bridge (BDG), and the CoT fallback (CoT-FB). To assess the contribution of each component, we conduct ablation studies with Gemini3-Flash on FOLIO and ProverQA, which emphasize semantic alignment and complex deductive reasoning, respectively. We exclude ProofWriter and PrOntoQA from this analysis because all ablation variants achieve near-perfect accuracy on these benchmarks, making it difficult to distinguish the contribution of each component. We first examine the contributions of GRD and BDG with CoT-FB disabled, as shown in the first four rows of Table 3. The *FOL* variant applies the solver directly to the initial FOL representation. The *GRD + LLM-CMP* variant introduces GRD but replaces BDG with LLM-based completion of missing semantic relations (LLM-CMP), followed by Gate evaluation. The *FOL + BDG* variant removes GRD and CoT-FB from *POGB*, and *GRD + BDG* removes only CoT-FB. Finally, we restore CoT-FB to obtain the full *POGB* pipeline and compare it with the standalone *CoT*, as shown in the last two rows of Table 3.

Applying the semantic bridge directly to *FOL* (i.e., *FOL + BDG*) yields modest accuracy gains of 3.92 and 0.60 percentage points on FOLIO and ProverQA, respectively. Incorporating the grounding table (i.e., *GRD + BDG*) further improves accuracy to 81.86% and 82.60%, outperforming *FOL + BDG* by 8.33 and 10.20 percentage points. Conversely, removing the grounding table not only substantially reduces accuracy but also increases candidate bridges by a factor of 75.2 and Gate calls by 124.8%. Moreover, replacing the semantic bridge in *GRD + BDG* with LLM-based completion (i.e., *GRD + LLM-CMP*) drops performance by 11.27 and 11.20 percentage points. These results highlight the complementary roles of both components: the grounding table preserves NL semantics and narrows the search space for candidate bridges, while the semantic bridge uses solver-controlled logical reasoning to recover missing semantic relations more reliably than LLM-based completion.

Incorporating CoT fallback into *GRD + BDG* improves accuracy by 6.87 and 6.40 percentage points on FOLIO and ProverQA, respectively. The resulting full *POGB* pipeline outperforms *CoT* by 4.91 and 14.20 percentage points. Further analysis shows that the formal component achieves substantially higher accuracy than *CoT* on instances for which it returns a non-UNKNOWN decision, particularly on ProverQA instances requiring longer reasoning chains. These findings reveal the complementary roles of the formal component and CoT fallback: the former provides more reliable results when it returns a non-UNKNOWN decision, while the latter handles the remaining unresolved cases.

5 Conclusion

In this paper, we propose Proof-Obligation-Guided Bridging (POGB), a method for recovering missing semantic relations in the formal representations generated by autoformalization. POGB preserves the NL semantic groundings of formal elements and uses proof obligations to localize missing semantic relations, thereby guiding their recovery. Moreover, POGB decouples semantic understanding from logical reasoning by assigning the former to the neural model and the latter to the symbolic solver. Specifically, the neural model's role is restricted to autoformalization and the evaluation of candidate semantic bridges, while the symbolic solver retains control over all logical reasoning. By letting solvers reason and models understand, POGB leverages their complementary strengths and offers a promising approach to neuro-symbolic integration. It is worth noting that the effectiveness of POGB depends on the quality of autoformalization and the reliability of Gate. Future advances in LLM-based autoformalization and Gate evaluation could therefore further enhance POGB's capabilities.

AI USE STATEMENT

In this work, we used generative AI tools for providing feedback on research methodology or experiments, implementing methods, and assisting with (autoformalization) translation. We have not used generative AI tools for helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, cleaning and reformatting dataset, supporting qualitative and thematic data analysis, or interpreting results; and generating synthetic data sets, proposing or refining hypotheses are not applicable to this work. Additionally, we used generative AI tools for editing software code, sourcing/searching for information, editing the research paper to improve readability, and identifying relevant literature. We have reviewed all AI-assisted work. AI-assisted literature search and methodological suggestions were critically reviewed by the authors, and AI-assisted code was checked and tested for correctness. All final methodological decisions, experimental analyses, claims, and conclusions were made and verified by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide detailed descriptions of the POGB framework, implementation details, experimental settings, and ablation studies. We use publicly available benchmarks, with the corresponding data processing and evaluation procedures described in the paper. We will publicly release the complete source code and experimental configurations upon acceptance.

REFERENCES

Iván Arcuschin, Jett Janiak, Robert Krzyzanowski, Senthooran Rajamanoharan, Neel Nanda, and Arthur Conmy. Chain-of-thought reasoning in the wild is not always faithful. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Vaishak Belle and Gary Marcus. The future is neuro-symbolic: Where has it been, and where is it going? In *Proceedings of the 40th AAAI Conference on Artificial Intelligence*, pp. 40954–40961, 2026.

Rahul Bhagat and Eduard Hovy. What is a paraphrase? *Computational Linguistics*, 39(3):463–472, 2013.

Ning Bian, Xianpei Han, Le Sun, Hongyu Lin, Yaojie Lu, Ben He, Shanshan Jiang, and Bin Dong. ChatGPT is a knowledgeable but inexperienced solver: An investigation of commonsense problem in large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation*, pp. 3098–3110, 2024.

Roland N Bol, Krzysztof R Apt, and Jan Willem Klop. An analysis of loop checking mechanisms for logic programs. *Theoretical Computer Science*, 86(1):35–79, 1991.

Andrea Brunello, Cristian Curaba, Luca Geatti, Michele Mignani, Angelo Montanari, and Nicola Saccomanno. Fixing FOLIO and MALLS: Verified annotations and an LLM-assisted framework to focus human relabeling. *arXiv preprint arXiv:2606.02837*, 2026.

Benjamin Callewaert, Simon Vandevelde, and Joost Vennekens. VERUS-LM: A versatile framework for combining LLMs with symbolic reasoning. In *Proceedings of the 41st International Conference on Logic Programming*, pp. 47–62, 2025.

Chuxue Cao, Jinluan Yang, Haoran Li, Kunhao Pan, Zijian Zhao, Zhengyu Chen, Yuchen Tian, Lijun Wu, Conghui He, Sirui Han, et al. Pushing the boundaries of natural reasoning: Interleaved bonus from formal-logic verification. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Luca Cardelli. Type systems. *ACM Computing Surveys*, 28(1):263–264, 1996.

Pankaj Chandre, Parikshit Mahalle, Gitanjali Shinde, Bhagyashree Shendkar, and Shraddha Kashid. Neuro-symbolic AI: A future of tomorrow. *ASEAN Journal on Science and Technology for Development*, 42(2):187–200, 2025.

Guoxin Chen, Wu Jing, Xinjie Chen, Xin Zhao, Ruihua Song, Chengxi Li, Kai Fan, Dayiheng Liu, and Minpeng Liao. ReForm: Reflective autoformalization with prospective bounded sequence optimization. In *Proceedings of the 14th International Conference on Learning Representations*, pp. 38792–38817, 2026.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner, Fabien Roger, et al. Reasoning models don't always say what they think. *arXiv preprint arXiv:2505.05410*, 2025.

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*, pp. 7358–7370, 2021.

Ernest Davis. Logical formalizations of commonsense reasoning: A survey. *Journal of Artificial Intelligence Research*, 59:651–723, 2017.

Matthew Gerber and Joyce Y Chai. Semantic role labeling of implicit arguments for nominal predicates. *Computational Linguistics*, 38(4):755–798, 2012.

Google DeepMind. Gemini 3 Flash - Model card, 2025. URL `https://deepmind.google/models/model-cards/gemini-3-flash/`.

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*, pp. 22017–22031, 2024.

Wesley H Holliday, Matthew Mandelkern, and Cedegao E Zhang. Conditional and modal reasoning in large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 3800–3821, 2024.

Ruikang Hu, Shaoyu Lin, Yeliang Xiu, and Yongmei Liu. LTRAG: Enhancing autoformalization and self-refinement for logical reasoning with thought-guided RAG. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 2483–2493, 2025.

John R Josephson and Susan G Josephson. *Abductive inference: Computation, philosophy, technology*. Cambridge University Press, 1996.

Ralf Kneuper. Limits of formal methods. *Formal Aspects of Computing*, 9(4):379–394, 1997.

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*, pp. 132–147, 2025.

Jinu Lee and Wonseok Hwang. SymBa: Symbolic backward chaining for structured natural language reasoning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics*, pp. 2468–2484, 2025.

Jindong Li, Yali Fu, Yang Yang, Jiahong Liu, Hongce Zhang, Haoxuan Li, Yutao Yue, and Menglin Yang. A survey on LLM symbolic reasoning. In *AAAI 2026 Bridge Program on Logical and Symbolic Reasoning in Language Models*, 2026a.

Qingchuan Li, Jiatong Li, Zirui Liu, Mingyue Cheng, Yitong Zhou, Yuting Zeng, Qi Liu, and Tongxuan Liu. Are LLMs stable formal logic translators in logical reasoning across linguistically diversified texts? In *Proceedings of the ACM Web Conference 2026*, pp. 3633–3644, 2026b.

PD Lincoln and Natarajan Shankar. Proof search in first-order linear logic and other cut-free sequent calculi. In *Proceedings of the 9th Annual Symposium on Logic in Computer Science*, pp. 282–291, 1994.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. DeepSeek-V3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. Faithful chain-of-thought reasoning. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pp. 305–329, 2023.

Agnieszka Mensfelt, David Tena Cucala, Santiago Franco, Angeliki Koutsoukou-Argyraki, Vince Trencsenyi, and Kostas Stathis. Towards a common framework for autoformalization. In *Proceedings of the 40th AAAI Conference on Artificial Intelligence*, pp. 40971–40980, 2026.

Hayden Moore and Asfahan Shah. Evaluating autoformalization robustness via semantically similar paraphrasing. In *AAAI 2026 Bridge Program on Logical and Symbolic Reasoning in Language Models*, 2026.

Luca Mouchel, Debjit Paul, Shaobo Cui, Robert West, Antoine Bosselut, and Boi Faltings. A logical fallacy-informed framework for argument generation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics*, pp. 7296–7314, 2025.

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*, pp. 5153–5176, 2023.

OpenAI. GPT-5.5 system card, 2026. URL `https://deploymentsafety.openai.com/gpt-5-5`.

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*, pp. 3806–3824, 2023.

Rizky Ramadhana Putra, Raihan Sultan Pasha Basuki, Yutong Cheng, and Peng Gao. NL2LOGIC: AST-guided translation of natural language into first-order logic with large language models. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 6035–6051, 2026.

Chengwen Qi, Ren Ma, Bowen Li, Binyuan Hui, Jinwang Wu, Yuanjun Laili, Conghui He, et al. Large language models meet symbolic provers for logical reasoning evaluation. In *Proceedings of the 13th International Conference on Learning Representations*, pp. 54186–54205, 2025.

Yiwen Qiu, Linjuan Wu, Yizhou Liu, Yuchen Yan, Jin Ma, Xu Tan, Yao Hu, Daoxin Zhang, Wenqi Zhang, Weiming Lu, et al. Pause or fabricate? training language models for grounded reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 23056–23077, 2026.

Xin Quan, Marco Valentino, Louise A Dennis, and Andre Freitas. Faithful and robust LLM-driven theorem proving for NLI explanations. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pp. 17734–17755, 2025.

Xin Quan, Marco Valentino, Louise A Dennis, and André Freitas. Decompose-and-formalise: Recursively verifiable natural language inference. *arXiv preprint arXiv:2601.19605*, 2026.

Qwen Team. Qwen3.6-35B-A3B: Agentic coding power, now open to all, 2026. URL `https://qwen.ai/blog?id=qwen3.6-35b-a3b`.

Mohammad Raza and Natasa Milic-Frayling. Instantiation-based formalization of logical reasoning tasks using language models and logical solvers. In *Proceedings of the 34th International Joint Conference on Artificial Intelligence*, pp. 4633–4641, 2025.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 3982–3992, 2019.

Hyun Ryu, Gyeongman Kim, Hyemin S Lee, and Eunho Yang. Divide and translate: Compositional first-order logic translation and verification for complex logical reasoning. In *Proceedings of the 13th International Conference on Learning Representations*, pp. 24935–24964, 2025.

Abulhair Saparov and He He. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *Proceedings of the 11th International Conference on Learning Representations*, 2023.

Vikash Singh, Darion Cassel, Nathaniel Weir, Nick Feng, and Sam Bayless. VERGE: Formal refinement and guidance engine for verifiable LLM reasoning. *arXiv preprint arXiv:2601.20055*, 2026.

Christian Szegedy. A promising path towards autoformalization and general artificial intelligence. In *Proceedings of the 13th International Conference on Intelligent Computer Mathematics*, pp. 3–20, 2020.

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*, pp. 3621–3634, 2021.

Nicole Teo, Donghao Huang, and Zhaoxia Wang. A survey of commonsense reasoning in LLMs. *ACM Computing Surveys*, 58(14):371:1–371:34, 2026.

Armin Toroghi, Willis Guo, Ali Pesaranghader, and Scott Sanner. Verifiable, debuggable, and repairable commonsense logical reasoning via LLM-based theory resolution. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 6634–6652, 2024.

Qingxiang Wang, Cezary Kaliszyk, and Josef Urban. First experiments with neural translation of informal to formal mathematics. In *Proceedings of the 11th International Conference on Intelligent Computer Mathematics*, pp. 255–270, 2018.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Proceedings of the 38th Annual Conference on Neural Information Processing Systems*, pp. 95266–95290, 2024.

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. In *Proceedings of the 36th Annual Conference on Neural Information Processing Systems*, pp. 24824–24837, 2022.

Richard W Weyhrauch. Prolegomena to a theory of mechanized formal reasoning. *Artificial Intelligence*, 13(1-2):133–170, 1980.

Jim Woodcock, Peter Gorm Larsen, Juan Bicarregui, and John Fitzgerald. Formal methods: Practice and experience. *ACM Computing Surveys*, 41(4):19:1–19:36, 2009.

Yuhuai Wu, Albert Qiaochu Jiang, Wenda Li, Markus Rabe, Charles Staats, Mateja Jamnik, and Christian Szegedy. Autoformalization with large language models. In *Proceedings of the 36th Annual Conference on Neural Information Processing Systems*, pp. 32353–32368, 2022.

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*, pp. 13326–13365, 2024.

Jundong Xu, Hao Fei, Meng Luo, Qian Liu, Liangming Pan, William Yang Wang, Preslav Nakov, Mong-Li Lee, and Wynne Hsu. Aristotle: Mastering logical reasoning with a logic-complete decompose-search-resolve framework. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pp. 3052–3075, 2025.

Jundong Xu, Hao Scofield Fei, Huichi Zhou, Xin Quan, Qijun Huang, Shengqiong Wu, William Wang, Mong-Li Lee, and Wynne Hsu. LogicReward: Incentivizing LLM reasoning via step-wise logical supervision. In *Proceedings of the 14th International Conference on Learning Representations*, pp. 43490–43511, 2026.

Xiao-Wen Yang, Jie-Jing Shao, Lan-Zhe Guo, Bo-Wen Zhang, Zhi Zhou, Lin-Han Jia, Wang-Zhou Dai, and Yu-Feng Li. Neuro-symbolic artificial intelligence: Towards improving the reasoning abilities of large language models. In *Proceedings of the 34th International Joint Conference on Artificial Intelligence*, pp. 10770–10778, 2025.

A RUNNING EXAMPLE

In this section, we illustrate our proposed method through a simplified example from the FOLIO dataset (Han et al., 2024). The NL premise set $\mathbb{P} = \{p_1; p_2; p_3\}$ states: "Reading a book gives a person knowledge; gaining knowledge makes that person smarter; and Harry read *Walden*." The NL query q asks whether the statement "Harry is smarter than before" can be concluded from these premises. Let the constants h, w, and k denote Harry, *Walden*, and knowledge, respectively. We first perform a grounding autoformalization F to map $X = (\mathbb{P}, q)$ to

$$\begin{aligned}\mathcal{P} = \{\ \mathfrak{p}_1 :\ & \forall x_1 \forall y.\ \big(\mathsf{Book}(y) \wedge \mathsf{Read}(x_1, y)\big) \rightarrow \mathsf{Gain}(x_1, k),\\ \mathfrak{p}_2 :\ & \forall x_2.\ \mathsf{Gain}(x_2, k) \rightarrow \mathsf{Smarter}(x_2),\\ \mathfrak{p}_3 :\ & \mathsf{Book}(w) \wedge \mathsf{Read}(h, w)\ \},\qquad \mathfrak{q} = \mathsf{SmarterThanBefore}(h),\end{aligned}$$

and the grounding table $\mathcal{T}$ shown in Table 4. During proof-time inference, the solver can therefore access all entries in $\mathcal{T}$. In particular, type information $\mathrm{type}_{\mathcal{T}}(\mathfrak{e})$ constrains which instantiations are admissible, whereas NL information $\mathrm{NL}_{\mathcal{T}}(\mathfrak{e})$ guides the search for semantic bridges when exact symbolic matching fails.

Table 4: The grounding table $\mathcal{T}$ of the running example.

Formal element $\mathfrak{e}$	**Type information** $\mathrm{type}_{\mathcal{T}}(\mathfrak{e})$	**Natural-language information** $\mathrm{NL}_{\mathcal{T}}(\mathfrak{e})$
h	Person	"Harry"
w	Book	"*Walden*"
k	Object	"knowledge"
x_1	Person	"a person"
x_2	Person	"a person"
y	Book	"a book"
$\mathsf{Book}(\{a_1\})$	Book	"$\{a_1\}$ is a book"
$\mathsf{Read}(\{a_1\}, \{a_2\})$	Person, Book	"$\{a_1\}$ reads $\{a_2\}$"
$\mathsf{Gain}(\{a_1\}, k)$	Person	"$\{a_1\}$ gains knowledge"
$\mathsf{Smarter}(\{a_1\})$	Person	"$\{a_1\}$ is smarter"
$\mathsf{SmarterThanBefore}(\{a_1\})$	Person	"$\{a_1\}$ is smarter than before"

After completing the grounding autoformalization process described above, we perform the proof-obligation-guided search. Initialize the proof obligation as $\phi = \mathfrak{q} = \mathsf{SmarterThanBefore}(h)$. Algorithm 2 instantiates $\mathfrak{p}_1$ and $\mathfrak{p}_2$ using the substitutions $x_1 \mapsto h$, $x_2 \mapsto h$, and $y \mapsto w$, yielding $\mathfrak{i}_1$ and $\mathfrak{i}_2$. Type compatibility permits these substitutions while ruling out alternatives such as $x_1 \mapsto w$, $x_1 \mapsto k$, $x_2 \mapsto w$, $x_2 \mapsto k$, $y \mapsto h$, and $y \mapsto k$. Algorithm 2 considers $I = \{\mathfrak{i}_2\}$ and obtains

$$\textsc{Lit}(\mathfrak{i}_2) = \{\neg\mathsf{Gain}(h, k), \mathsf{Smarter}(h)\},$$

It then selects $S = \{\mathsf{Smarter}(h)\} \subseteq \textsc{Lit}(\mathfrak{i}_2)$, and constructs the candidate semantic support $\Sigma = \mathsf{Smarter}(h)$ and the candidate semantic bridge

$$\begin{aligned}\mathfrak{b} &= \Sigma \rightarrow \phi\\ &= \mathsf{Smarter}(h) \rightarrow \mathsf{SmarterThanBefore}(h).\end{aligned}$$

Logical abduction then derives the residual obligation $\mathfrak{o} = \mathsf{Gain}(h, k)$. Once the LLM-based Gate accepts $\mathfrak{b}$ based on its NL interpretation that "Harry is smarter" entails "Harry is smarter than before," Algorithm 1 proceeds to discharge the residual obligations:

$$\begin{aligned}\mathsf{SmarterThanBefore}(h) &\xrightarrow[\mathfrak{b}]{\mathfrak{p}_2[x_2 \mapsto h]} \{\mathsf{Gain}(h, k)\}\\ &\xrightarrow{\mathfrak{p}_1[x_1 \mapsto h, y \mapsto w]} \{\mathsf{Book}(w), \mathsf{Read}(h, w)\}\\ &\xrightarrow{\mathfrak{p}_3} \checkmark.\end{aligned}$$

Here, each arrow represents the residual obligations produced in a single round of derivation, with bridging applied as needed. Finally, premise $\mathfrak{p}_3$ discharges the two remaining obligations. In this example, only the displayed bridge $\mathfrak{b}$ requires LLM evaluation throughout the derivation.

B Case Studies

In each entailment-tree comparison below, step_i labels an inference step. int_i and φ_i denotes source intermediate natural-language conclusions and formalized intermediate representation, respectively.

B.1 Instantiated Alignment: From Freezing to a Change of State

Natural-language problem

Dataset: EntailmentBank **Gold label:** TRUE

p_1 "The water reaches its freezing point."
p_2 "Liquid is a kind of state of matter."
p_3 "Freezing is when a substance reaches its freezing point."
p_4 "Water is a kind of substance."
p_5 "Freezing means a substance changes from a liquid into a solid by decreasing heat energy."
p_6 "Solid is a kind of state of matter."
q "Water will change its state of matter when it reaches its freezing point."

Grounding autoformalization

$$
\begin{aligned}
&\mathfrak{p}_1 : \mathsf{ReachesFreezingPoint}(water),\\
&\mathfrak{p}_2 : \mathsf{StateOfMatter}(liquid),\\
&\mathfrak{p}_3 : \forall x\big(\mathsf{Substance}(x) \to (\mathsf{Freezing}(x) \leftrightarrow \mathsf{ReachesFreezingPoint}(x))\big),\\
&\mathfrak{p}_4 : \mathsf{Substance}(water),\\
&\mathfrak{p}_5 : \forall x\big(\mathsf{Substance}(x) \to \big(\mathsf{Freezing}(x) \leftrightarrow\\
&\qquad (\mathsf{ChangesFromTo}(x, liquid, solid) \wedge \mathsf{DecreasingHeatEnergy}(x))\big)\big),\\
&\mathfrak{p}_6 : \mathsf{StateOfMatter}(solid).\\
&\mathfrak{q} : \mathsf{ReachesFreezingPoint}(water) \to \mathsf{ChangesStateOfMatter}(water).
\end{aligned}
$$

Accepted alignment bridge

Local support. "Water changes from liquid to solid."
Local target. "Water changes its state of matter."

$$\mathfrak{b}_{\mathrm{align}} : \quad \mathsf{ChangesFromTo}(water, liquid, solid) \to \mathsf{ChangesStateOfMatter}(water).$$

Manual audit. This is a directionally valid semantic weakening from a named change between two states to a change of state.

Proof-obligation-guided bridging

Under implication introduction, the antecedent becomes a local assumption:

$$
\begin{aligned}
\mathfrak{q} &\xrightarrow[\text{assume } \mathsf{ReachesFreezingPoint}(water)]{\to\text{-intro}} \{\mathsf{ChangesStateOfMatter}(water)\}\\
&\xrightarrow{\mathfrak{b}_{\mathrm{align}}} \{\mathsf{ChangesFromTo}(water, liquid, solid)\}\\
&\xrightarrow{\mathfrak{p}_5} \{\mathsf{Substance}(water), \mathsf{Freezing}(water)\}\\
&\xrightarrow{\mathfrak{p}_3} \{\mathsf{Substance}(water), \mathsf{ReachesFreezingPoint}(water)\}\\
&\xrightarrow{\mathfrak{p}_4,\ \text{assumption}} \checkmark.
\end{aligned}
$$

Entailment-tree comparison

Benchmark gold tree.

step_1 $\{p_3, p_5\} \Rightarrow int_1$: reaching a freezing point changes a substance from liquid to solid.

step_2 $\{int_1, p_1, p_4\} \Rightarrow int_2$: the preceding statement instantiated for water.

step_3 $\{int_2, p_2, p_6\} \Rightarrow q$.

Our derivation.

step_1 Under the antecedent of $\mathfrak{q}$, $\mathfrak{p}_3, \mathfrak{p}_4, \mathfrak{p}_5 \Rightarrow \varphi_1$, where $\varphi_1 = \mathsf{ChangesFromTo}(water, liquid, solid)$.

step_2 $\{\varphi_1, \mathfrak{b}_{\text{align}}\} \Rightarrow \varphi_2$, where $\varphi_2 = \mathsf{ChangesStateOfMatter}(water)$.

step_3 $\{\varphi_2\} \Rightarrow \mathfrak{q}$ by implication introduction.

The two derivations share the same liquid-to-solid core. The gold edge uses p_2 and p_6 to type its natural-language endpoints. Our formalization names those endpoints as constants and makes the final lexical weakening explicit; p_1 is replaced by the local assumption introduced from $\mathfrak{q}$.

B.2 Pair Bridge: Alias Resolution through Two-Support Composition

Natural-language problem

Dataset: FOLIO **Gold label:** TRUE

p_1 "Ailton Silva, born in 1995, is commonly known as Ailton."

p_2 "Ailton is a football player who was loaned out to Braga."

p_3 "Ailton Silva is a Brazillian footballer who plays for Nautico. "

p_4 "Nautico is a football club along with Braga."

p_5 "Fluminense is a football club."

q "Ailton Silva was loaned out to a football club. "

Grounding autoformalization

$$\mathfrak{p}_1 : \mathsf{BornIn}(ailtonSilva, year1995) \wedge \mathsf{KnownAs}(ailtonSilva, ailton),$$
$$\mathfrak{p}_2 : \mathsf{FootballPlayer}(ailton) \wedge \mathsf{LoanedTo}(ailton, braga),$$
$$\mathfrak{p}_3 : \mathsf{BrazilianFootBaller}(ailtonSilva) \wedge \mathsf{PlaysFor}(ailtonSilva, nautico).$$
$$\mathfrak{p}_4 : \mathsf{FootballClub}(nautico) \wedge \mathsf{FootballClub}(braga).$$
$$\mathfrak{p}_5 : \mathsf{FootballClub}(flumninense).$$
$$\mathfrak{q} : \exists x\big(\mathsf{FootballClub}(x) \wedge \mathsf{LoanedTo}(ailtonSilva, x)\big).$$

Accepted two-support bridge

Support 1. "Ailton Silva is commonly known as Ailton."
Support 2. "Ailton was loaned out to Braga."
Target. "Ailton Silva was loaned out to Braga."

$$\mathfrak{b}_{\text{pair}} : \quad \big(\mathsf{KnownAs}(ailtonSilva, ailton) \wedge \mathsf{LoanedTo}(ailton, braga)\big) \rightarrow \mathsf{LoanedTo}(ailtonSilva, braga).$$

Manual audit and Gate scope. The first support explicitly fixes the two names to the same person in this discourse; the second transfers an unchanged loan relation.

Proof-obligation-guided bridging

Using $braga$ as the existential witness,

$$\mathfrak{q} \xrightarrow[x \mapsto braga]{\exists, \wedge} \{\mathsf{FootballClub}(braga), \mathsf{LoanedTo}(ailtonSilva, braga)\}$$
$$\xrightarrow{\mathfrak{b}_{\text{pair}}} \left\{ \begin{array}{l} \mathsf{FootballClub}(braga), \\ \mathsf{KnownAs}(ailtonSilva, ailton), \\ \mathsf{LoanedTo}(ailton, braga) \end{array} \right\} \xrightarrow{\mathfrak{p}_4, \mathfrak{p}_1, \mathfrak{p}_2} \checkmark.$$

The replay dependency core is exactly $\mathfrak{p}_1$, $\mathfrak{p}_2$, $\mathfrak{p}_4$, and the pair bridge.

B.3 Universal Alignment: Relational Weakening under a Universal Goal

Natural-language problem

Dataset: EntailmentBank **Gold label:** TRUE

p_1 "Vibrations are a source of sound."
p_2 "Plucking a string can cause that string to vibrate."
q "The vibrations of a string are a source of sound."

Grounding autoformalization

$$\mathfrak{p}_1 : \forall z\big(\mathsf{Vibration}(z) \to \mathsf{SourceOfSound}(z)\big),$$
$$\mathfrak{p}_2 : \forall x \forall y\big(\mathsf{String}(x) \wedge \mathsf{Pluck}(x, y) \to \mathsf{CauseToVibrate}(y, x)\big),$$
$$\mathfrak{q} : \forall x \forall y\big(\mathsf{String}(x) \wedge \mathsf{VibrationOf}(y, x) \to \mathsf{SourceOfSound}(y)\big).$$

Accepted alignment bridge

Local support. "y is a vibration of x."
Local target. "y is a vibration."

$$\mathfrak{b}_{\text{align}} : \quad \forall x \forall y\big(\mathsf{VibrationOf}(y, x) \to \mathsf{Vibration}(y)\big).$$

Manual audit. This is strict relational weakening: being a vibration *of* an object entails being a vibration.

Proof-obligation-guided bridging

Introducing eignvariables α, β and assume $\mathsf{String}(\alpha) \wedge \mathsf{VibrationOf}(\beta, \alpha)$:

$$\mathfrak{q} \xrightarrow{\forall,\forall,\to\text{-intro}} \{\mathsf{SourceOfSound}(\beta)\} \xrightarrow{\mathfrak{p}_1} \{\mathsf{Vibration}(\beta)\} \xrightarrow{\mathfrak{b}_{\text{align}}} \{\mathsf{VibrationOf}(\beta, \alpha)\} \xrightarrow{\text{assumption}} \checkmark.$$

The same eigenvariables α, β are preserved throughout; no fact about a fixed string or vibration is generalized. Vampire replay uses P1 and the bridge, and removal of the bridge reopens the goal.

Entailment-tree comparison

Benchmark gold tree.

step_1 $\{p_1, p_2\} \Rightarrow q$.

The benchmark treats the plucking scenario and the general source-of-sound fact as one joint natural-language explanation.

Our grounding derivation.

step_1 Introduce arbitrary α, β and assume the antecedent of $\mathfrak{q}$, yielding $\varphi_1 = \mathsf{VibrationOf}(\beta, \alpha)$.
step_2 $\{\varphi_1, \mathfrak{b}_{\text{align}}\} \Rightarrow \varphi_2$, where $\varphi_2 = \mathsf{Vibration}(\beta)$.
step_3 $\{\varphi_2, \mathfrak{p}_1\} \Rightarrow \mathfrak{q}$ by universal and implication introduction.

The formal antecedent already supplies the vibration-of relation, so $\mathfrak{p}_2$'s $\mathsf{CauseToVibrate}$ predicate is not required by the replay. Thus the gold edge uses p_1 and p_2, whereas the certified formal core uses $\mathfrak{p}_1$ and the explicit lexical bridge. Gold and formal dependencies are compared here rather than presented as identical.

B.4 Universal Alignment: A Solver-Derived Existential Residual

Natural-language problem

Dataset: EntailmentBank **Gold label:** TRUE

p_1 "Trees require a certain thickness of soil to grow."
p_2 "A tree is a kind of plant."
p_3 "A plant is a kind of organism."
p_4 "The decrease of something required by an organism has a negative impact on that organism's survival."
p_5 "Drought decreases the thickness of soil."
q "Drought can cause harm to trees by decreasing the thickness of soil."

Grounding autoformalization

Let $d = \textit{drought}$ and $s = \textit{soilThickness}$.

$$\begin{aligned}
&\mathfrak{p}_1 : \forall x\big(\mathsf{Tree}(x) \to \mathsf{Require}(x, s)\big),\\
&\mathfrak{p}_2 : \forall x\big(\mathsf{Tree}(x) \to \mathsf{Plant}(x)\big),\\
&\mathfrak{p}_3 : \forall x\big(\mathsf{Plant}(x) \to \mathsf{Organism}(x)\big),\\
&\mathfrak{p}_4 : \forall x \forall y \forall z \Big(\mathsf{Organism}(x) \wedge \mathsf{Require}(x, y) \wedge \mathsf{Decrease}(z, y)\\
&\qquad\qquad \to \mathsf{NegativeImpactOnSurvival}(z, x)\Big),\\
&\mathfrak{p}_5 : \mathsf{Decrease}(d, s),\\
&\mathfrak{q} : \forall x\Big(\mathsf{Tree}(x) \to \big(\mathsf{Decrease}(d, s) \wedge \mathsf{CauseHarm}(d, x)\big)\Big).
\end{aligned}$$

Accepted universal alignment bridge

Local support. "Drought has a negative impact on the survival of an arbitrary tree α."
Local target. "Drought causes harm to that same tree α."

$$\mathfrak{b}_\forall : \quad \forall x\big(\mathsf{NegativeImpactOnSurvival}(d, x) \to \mathsf{CauseHarm}(d, x)\big).$$

Manual audit. The bridge weakens a negative impact on an organism's survival to harm.

Proof-obligation-guided bridging

After introducing eignvariable α and assuming $\mathsf{Tree}(\alpha)$, P4 exposes the solver-derived residual

$$\rho(\alpha) = \exists y\Big(\mathsf{Decrease}(d, y) \wedge \mathsf{Organism}(\alpha) \wedge \mathsf{Require}(\alpha, y)\Big).$$

The exact prover chooses $y := s$, and the two algorithms compose as follows:

$$\begin{aligned}
\mathfrak{q} &\xrightarrow{\forall, \to, \wedge} \{\mathsf{Decrease}(d, s), \mathsf{CauseHarm}(d, \alpha)\}\\
&\xrightarrow{\mathfrak{b}_\forall,\, \mathfrak{p}_4[x \mapsto \alpha, y \mapsto s, z \mapsto d]} \{\mathsf{Decrease}(d, s), \mathsf{Organism}(\alpha), \mathsf{Require}(\alpha, s)\},\\
\mathsf{Organism}(\alpha) &\xrightarrow{\mathfrak{p}_3} \{\mathsf{Plant}(\alpha)\} \xrightarrow{\mathfrak{p}_2} \{\mathsf{Tree}(\alpha)\} \xrightarrow{\text{assumption}} \checkmark,\\
\mathsf{Require}(\alpha, s) &\xrightarrow{\mathfrak{p}_1} \{\mathsf{Tree}(\alpha)\} \xrightarrow{\text{assumption}} \checkmark, \qquad \mathsf{Decrease}(d, s) \xrightarrow{\mathfrak{p}_5} \checkmark.
\end{aligned}$$

All five source facts and $\mathfrak{b}_\forall$ occur in the final Vampire replay; the bridge is subset-minimal and the augmented theory is consistent.

Entailment-tree comparison

Benchmark gold tree.

step_1	$\{p_2, p_3\} \Rightarrow \textit{int}_1$: a tree is an organism.
step_2	$\{\textit{int}_1, p_1\} \Rightarrow \textit{int}_2$: trees are organisms requiring soil thickness.
step_3	$\{\textit{int}_2, p_4\} \Rightarrow \textit{int}_3$: decreasing soil thickness can harm trees.
step_4	$\{\textit{int}_3, p_5\} \Rightarrow q$.

Our grounding derivation.

step_1	$\{\mathfrak{p}_2, \mathfrak{p}_3\} \Rightarrow \varphi_1$, where $\varphi_1 = \mathsf{Organism}(\alpha)$.
step_2	$\{\mathfrak{p}_1\} \Rightarrow \varphi_2$, where $\varphi_2 = \mathsf{Require}(\alpha, s)$.
step_3	$\{\varphi_1, \varphi_2, \mathfrak{p}_4, \mathfrak{p}_5\} \Rightarrow \varphi_3$, where $\varphi_3 = \mathsf{NegativeImpactOnSurvival}(d, \alpha)$.
step_4	$\{\varphi_3, \mathfrak{b}_\forall\} \Rightarrow \mathsf{CauseHarm}(d, \alpha)$; conjunction and universal introduction derive $\mathfrak{q}$.

The dependency DAGs are nearly isomorphic. Our derivation only makes explicit the lexical step from "negative impact on survival" to "harm," which the third gold edge leaves implicit.

B.5 Instantiated Alignment: A Long Refutation through Lexical Paraphrase

Natural-language problem

Dataset: ProverQA **Gold label:** FALSE

p_1	Hunter uses bait or knows ictiobus.	p_7	Hunter travels rivers or knows ictiobus.
p_2	Hunter is not skilled.	p_8	Hunter knows the river.
p_3	If Hunter can read the water, then he is either skilled or experienced, but not both.	p_9	Every fisherman is either successful or unsuccessful, but not both.
p_4	Hunter does not understand currents.	p_{10}	If Hunter is patient, then he is either adventurous or can read the water, but not both.
p_5	If Hunter knows ictiobus, then he either reads the water or understands currents, but not both.	p_{11}	Hunter does not use bait.
p_6	Hunter either catches fish or is successful.	p_{12}	If Hunter knows the river, then he either has experience or catches fish, but not both.
q	Hunter is an unsuccessful fisherman.		

Grounding autoformalization (constant h = Hunter)

$$
\begin{aligned}
\mathfrak{p}_1 &: \mathsf{UsesBait}(h) \vee \mathsf{KnowsIctiobus}(h),\\
\mathfrak{p}_2 &: \neg\mathsf{Skilled}(h),\\
\mathfrak{p}_3 &: \mathsf{CanReadWater}(h) \to (\mathsf{Skilled}(h) \oplus \mathsf{Experienced}(h)),\\
\mathfrak{p}_4 &: \neg\mathsf{UnderstandsCurrents}(h),\\
\mathfrak{p}_5 &: \mathsf{KnowsIctiobus}(h) \to (\mathsf{CanReadWater}(h) \oplus \mathsf{UnderstandsCurrents}(h)),\\
\mathfrak{p}_6 &: \mathsf{CatchesFish}(h) \oplus \mathsf{Successful}(h),\\
\mathfrak{p}_7 &: \mathsf{TravelsRivers}(h) \vee \mathsf{KnowsIctiobus}(h),\\
\mathfrak{p}_8 &: \mathsf{KnowsRiver}(h),\\
\mathfrak{p}_9 &: \forall x\big(\mathsf{Fisherman}(x) \to (\mathsf{Successful}(x) \oplus \mathsf{Unsuccessful}(x))\big),\\
\mathfrak{p}_{10} &: \mathsf{Patient}(h) \to (\mathsf{Adventurous}(h) \oplus \mathsf{CanReadWater}(h)),\\
\mathfrak{p}_{11} &: \neg\mathsf{UsesBait}(h),\\
\mathfrak{p}_{12} &: \mathsf{KnowsRiver}(h) \to (\mathsf{HasExperience}(h) \oplus \mathsf{CatchesFish}(h)),\\
\mathfrak{q} &: \mathsf{Unsuccessful}(h) \wedge \mathsf{Fisherman}(h).
\end{aligned}
$$

Accepted instantiated alignment bridge

Local support. "Hunter is experienced."
Local target. "Hunter has experience."

$$\mathfrak{b}_{\text{align}}: \quad \mathsf{Experienced}(h) \to \mathsf{HasExperience}(h).$$

Manual audit and evidence scope. The support directionally entails the target under an ordinary reading.

Proof-obligation-guided bridging

For the false label, the refutation target is $\neg\mathfrak{q} \equiv \neg\mathsf{Unsuccessful}(h) \vee \neg\mathsf{Fisherman}(h)$. The arrows summarize a solver-derived backward dependency trace; braces list sufficient residual obligations that remain for recursive search:

$$
\begin{aligned}
\neg\mathfrak{q} &\xrightarrow{\mathfrak{p}_9[x\mapsto h]} \{\mathsf{Successful}(h)\},\\
\mathsf{Successful}(h) &\xrightarrow{\mathfrak{p}_6} \{\neg\mathsf{CatchesFish}(h)\},\\
\neg\mathsf{CatchesFish}(h) &\xrightarrow{\mathfrak{p}_{12}} \{\mathsf{HasExperience}(h), \mathsf{KnowsRiver}(h)\},\\
\mathsf{HasExperience}(h) &\xrightarrow[\mathfrak{b}_{\text{align}}]{\mathfrak{p}_3} \{\mathsf{CanReadWater}(h), \neg\mathsf{Skilled}(h)\},\\
\mathsf{CanReadWater}(h) &\xrightarrow{\mathfrak{p}_5} \{\mathsf{KnowsIctiobus}(h), \neg\mathsf{UnderstandsCurrents}(h)\},\\
\mathsf{KnowsIctiobus}(h) &\xrightarrow{\mathfrak{p}_1} \{\neg\mathsf{UsesBait}(h)\} \xrightarrow{\mathfrak{p}_{11}} \checkmark.
\end{aligned}
$$

Proof-obligation-guided bridging (continued)

The remaining side obligations discharged directly by $\mathfrak{p}_8$, $\mathfrak{p}_2$, and $\mathfrak{p}_4$. Thus the alignment is proposed only when backward search reaches $\mathsf{HasExperience}(h)$: p_3 supplies the local support $\mathsf{Experienced}(h)$, while its two solver-derived residuals are proved recursively.

B.6 Two-Bridge Composition: Grounding Typing plus Instantiated Alignment

Natural-language problem

Dataset: ProverQA **Gold label:** FALSE

p_1	Kehlani thrives in change.
p_2	If Kehlani has complex social bonds, then she is either resourceful or intelligent, but not both.
p_3	A leech either prioritizes its own survival or respects the needs of others, but not necessarily both.
p_4	Kehlani does not prioritize her own survival.
p_5	If Kehlani is intelligent, then she either cares for her host or protects the ecosystem, but not both.
p_6	Kehlani does not care for her host.
p_7	If Kehlani maintains balance in her ecosystem, then she is either adaptable or resourceful, but not both.
p_8	Kehlani does not adjust her behavior.
p_9	If Kehlani lives with crayfish and respects their needs, then she has complex social bonds.
q	Kehlani does not protect the ecosystem.
p_{10}	All leeches that are sensitive to their environment are adaptable and have strong instincts.
p_{11}	If Kehlani either adjusts her behavior or thrives in change (but not both), then she is adaptable.
p_{12}	Kehlani is cooperative.
p_{13}	Kehlani communicates effectively.
p_{14}	Kehlani maintains balance in her ecosystem.
p_{15}	If Kehlani forms long-term attachments or shows empathy towards others, then she can have complex social bonds.
p_{16}	If Kehlani is cooperative, then she lives with crayfish and communicates effectively.
p_{17}	Jamir does not adjust her behavior.
p_{18}	If Princeton is intelligent, then she either cares for her host or protects the ecosystem, but not both.

Grounding autoformalization (constants k, j, p)

Let $k = kehlani$, $j = jamir$, and $p = princeton$. The grounding constant table assigns all three the entity type *leech*; in particular, k : *leech*.

$$
\begin{aligned}
&\mathfrak{p}_1 : \mathsf{ThriveInChange}(k),\\
&\mathfrak{p}_2 : \mathsf{HasComplexSocialBonds}(k) \rightarrow \big(\mathsf{Resourceful}(k) \oplus \mathsf{Intelligent}(k)\big),\\
&\mathfrak{p}_3 : \forall x\big(\mathsf{Leech}(x) \rightarrow \big(\mathsf{PrioritizeOwnSurvival}(x) \oplus \mathsf{RespectNeedsOfOthers}(x)\big)\big),\\
&\mathfrak{p}_4 : \neg\mathsf{PrioritizeOwnSurvival}(k),\\
&\mathfrak{p}_5 : \mathsf{Intelligent}(k) \rightarrow \big(\mathsf{CareForHost}(k) \oplus \mathsf{ProtectEcosystem}(k)\big),\\
&\mathfrak{p}_6 : \neg\mathsf{CareForHost}(k),\\
&\mathfrak{p}_7 : \mathsf{MaintainBalanceInEcosystem}(k) \rightarrow \big(\mathsf{Adaptable}(k) \oplus \mathsf{Resourceful}(k)\big),\\
&\mathfrak{p}_8 : \neg\mathsf{AdjustBehavior}(k),\\
&\mathfrak{p}_9 : \big(\mathsf{LiveWithCrayfish}(k) \wedge \mathsf{RespectsTheirNeeds}(k)\big) \rightarrow \mathsf{HasComplexSocialBonds}(k),\\
&\mathfrak{p}_{10} : \forall x\big(\mathsf{Leech}(x) \wedge \mathsf{SensitiveToEnvironment}(x) \rightarrow\\
&\qquad \mathsf{Adaptable}(x) \wedge \mathsf{StrongInstincts}(x)\big),\\
&\mathfrak{p}_{11} : \big(\mathsf{AdjustBehavior}(k) \oplus \mathsf{ThriveInChange}(k)\big) \rightarrow \mathsf{Adaptable}(k),\\
&\mathfrak{p}_{12} : \mathsf{Cooperative}(k),\\
&\mathfrak{p}_{13} : \mathsf{CommunicateEffectively}(k),\\
&\mathfrak{p}_{14} : \mathsf{MaintainBalanceInEcosystem}(k),\\
&\mathfrak{p}_{15} : \big(\mathsf{FormLongTermAttachments}(k) \vee \mathsf{ShowEmpathyTowardsOthers}(k)\big) \rightarrow\\
&\qquad \mathsf{CanHaveComplexSocialBonds}(k),\\
&\mathfrak{p}_{16} : \mathsf{Cooperative}(k) \rightarrow \big(\mathsf{LiveWithCrayfish}(k) \wedge \mathsf{CommunicateEffectively}(k)\big),\\
&\mathfrak{p}_{17} : \neg\mathsf{AdjustBehavior}(j),
\end{aligned}
$$

Grounding autoformalization (constants k, j, p) (continued)

$$\mathfrak{p}_{18} : \mathsf{Intelligent}(p) \to \big(\mathsf{CareForHost}(p) \oplus \mathsf{ProtectEcosystem}(p)\big),$$
$$\mathfrak{q} : \neg\mathsf{ProtectEcosystem}(k).$$

Accepted two-bridge set

Bridge 1: contextual type-to-predicate materialization. During autoformalization, the grounding table assigns k the entity type *leech*. A separate contextual-background check accepts the contextual bridge $\mathfrak{b}_{\mathrm{type}} : \mathsf{Leech}(k)$.

Bridge 2: instantiated lexical alignment. Local support: "Kehlani respects the needs of others."
Local target: "Kehlani respects their needs."

$$\mathfrak{b}_{\mathrm{align}} : \quad \mathsf{RespectNeedsOfOthers}(k) \to \mathsf{RespectsTheirNeeds}(k).$$

The two clauses are a directional paraphrase with the same subject.

Proof-obligation-guided bridging

For the false label, backward search starts from $\neg\mathfrak{q} \equiv \mathsf{ProtectEcosystem}(k)$. Braces again denote residual obligations, rather than forward-derived facts:

$$\neg\mathfrak{q} \xrightarrow{\mathfrak{p}_5} \{\mathsf{Intelligent}(k), \neg\mathsf{CareForHost}(k)\},$$
$$\mathsf{Intelligent}(k) \xrightarrow{\mathfrak{p}_2} \{\mathsf{HasComplexSocialBonds}(k), \neg\mathsf{Resourceful}(k)\},$$
$$\neg\mathsf{Resourceful}(k) \xrightarrow{\mathfrak{p}_7} \{\mathsf{Adaptable}(k), \mathsf{MaintainBalanceInEcosystem}(k)\},$$
$$\mathsf{Adaptable}(k) \xrightarrow{\mathfrak{p}_{11}} \{\mathsf{ThriveInChange}(k), \neg\mathsf{AdjustBehavior}(k)\},$$
$$\mathsf{HasComplexSocialBonds}(k) \xrightarrow{\mathfrak{p}_9} \{\mathsf{LiveWithCrayfish}(k), \mathsf{RespectsTheirNeeds}(k)\},$$
$$\mathsf{LiveWithCrayfish}(k) \xrightarrow{\mathfrak{p}_{16}} \{\mathsf{Cooperative}(k)\},$$
$$\mathsf{RespectsTheirNeeds}(k) \xrightarrow[\mathfrak{b}_{\mathrm{align}}]{\mathfrak{p}_3[x\mapsto k]} \{\mathsf{Leech}(k), \neg\mathsf{PrioritizeOwnSurvival}(k)\},$$
$$\mathsf{Leech}(k) \xrightarrow{\mathfrak{b}_{\mathrm{type}}} \checkmark.$$

$\mathfrak{p}_6$, $\mathfrak{p}_{14}$, $\mathfrak{p}_1$, $\mathfrak{p}_8$, $\mathfrak{p}_{12}$, and $\mathfrak{p}_4$ discharge the side obligations. Retrieval proposes $\mathfrak{b}_{\mathrm{align}}$ only at the unresolved $\mathsf{RespectsTheirNeeds}(k)$; its $\mathfrak{p}_3$ residual then exposes the second unresolved obligation $\mathsf{Leech}(k)$, discharged by the contextual materialization $\mathfrak{b}_{\mathrm{type}}$.

C PROOF OF SOUNDNESS

Lemma 1 (Existential decomposition)**.** *For $\phi = \exists x.\, \psi$, let $\mathcal{B}$ be the bridge set returned by a recursive call for an admissible instance $\psi[\theta] \in \textsc{Inst}(\psi, \mathcal{T})$. If $\mathcal{S}(\mathcal{P} \cup \mathcal{B}, \psi[\theta]) = \textsc{Proved}$, then $\mathcal{S}(\mathcal{P} \cup \mathcal{B}, \phi) = \textsc{Proved}$.*

Proof. Apply existential introduction with the witness θ's bridge set. □

Lemma 2 (Universal decomposition)**.** *For $\phi = \forall x.\, \psi$, introduce a fresh eigenvariable α and let $\mathcal{B}$ be the bridge set returned for $\psi[x \mapsto \alpha]$. Each bridge involving α is universally quantified over the corresponding variable and accepted by Gate with that universal scope. If $\mathcal{S}(\mathcal{P} \cup \mathcal{B}, \psi[x \mapsto \alpha]) = \textsc{Proved}$, then $\mathcal{S}(\mathcal{P} \cup \mathcal{B}, \phi) = \textsc{Proved}$.*

Proof. The relevant bridges are asserted for every admissible assignment to the variable represented by α. Freshness and universal closure ensure that the assumptions do not depend on a particular value of α. Universal introduction therefore extends the subproof of $\psi[x \mapsto \alpha]$ to a proof of $\forall x.\, \psi$, using the same bridge set. □

Lemma 3 (Conjunctive decomposition)**.** *For $\phi = \phi_1 \wedge \phi_2$, let $\mathcal{B}_1$ and $\mathcal{B}_2$ be the bridge sets returned by the two recursive calls. If $\mathcal{S}(\mathcal{P} \cup \mathcal{B}_i, \phi_i) = \textsc{Proved}$ for $i = 1, 2$, then their merged set $\mathcal{B} = \mathcal{B}_1 \cup \mathcal{B}_2$ satisfies $\mathcal{S}(\mathcal{P} \cup \mathcal{B}, \phi) = \textsc{Proved}$.*

Proof. Combine the two subproofs by conjunction introduction. □

Lemma 4 (Disjunctive decomposition)**.** *For $\phi = \phi_1 \vee \phi_2$, select any successful branch $i \in \{1, 2\}$ with returned bridge set $\mathcal{B}_i$. If $\mathcal{S}(\mathcal{P} \cup \mathcal{B}_i, \phi_i) = \textsc{Proved}$, then the selected set $\mathcal{B} = \mathcal{B}_i$ satisfies $\mathcal{S}(\mathcal{P} \cup \mathcal{B}, \phi) = \textsc{Proved}$.*

Proof. Apply disjunction introduction to the selected subproof, retaining only its bridge set. □

Lemma 5 (Bridge identification)**.** *If Algorithm 2 yields $(\mathcal{B}, \mathfrak{o})$ for ϕ, $\mathcal{S}(\mathcal{P} \cup \mathcal{B}_\mathfrak{o}, \mathfrak{o}) = \textsc{Proved}$ with a Gate-accepted bridge set $\mathcal{B}_\mathfrak{o}$, then*

$$\mathcal{S}(\mathcal{P} \cup \mathcal{B} \cup \mathcal{B}_\mathfrak{o}, \phi) = \textsc{Proved} \quad \wedge \quad \mathrm{Gate}(\mathcal{B} \cup \mathcal{B}_\mathfrak{o}, \mathcal{T}) = \textsc{Accepted}.$$

Proof. The yield statement requires Gate acceptance. The instantiated premise set $\mathcal{I}$ follows from $\mathcal{P}$. By the abduction condition in Section 3.3, the instantiated premises and the residual subproof jointly establish the selected support Σ. □

Theorem 1 (Soundness of successful search)**.** *If Algorithm 1 returns a bridge set $\mathcal{B}$ for a goal $\mathfrak{g}$, then $\mathcal{B}$ satisfies both criteria in Equation (1):*

$$\mathcal{S}(\mathcal{P} \cup \mathcal{B}, \mathfrak{g}) = \textsc{Proved} \quad \wedge \quad \mathrm{Gate}(\mathcal{B}, \mathcal{T}) = \textsc{Accepted}.$$

Proof. We prove by induction on the successful recursive derivation that each successful call $\textsc{Prove}(\phi, \mathcal{A}, \mathcal{T})$ returns a Gate-accepted bridge set under which the solver can discharge ϕ.

Base case. The solver directly proves ϕ from $\mathcal{P}$. The returned bridge set is empty, so Gate acceptance is vacuous.

Inductive step. Assume the claim for the successful recursive subcalls. Normalization preserves equivalence, so it suffices to discharge the normalized obligation. Its four logical decomposition cases are as follows.

(i) $\phi = \exists x.\, \psi$. A recursive call succeeds for an admissible instance $\psi[\theta]$. The induction hypothesis establishes that instance with accepted bridges, and Lemma 1 discharges ϕ.

(ii) $\phi = \forall x.\, \psi$. The recursive call succeeds for $\psi[x \mapsto \alpha]$, with α recorded as arbitrary in the extended grounding table. The induction hypothesis establishes the body with accepted bridges; those involving α retain their universal scope. Lemma 2 discharges ϕ.

(iii) $\phi = \phi_1 \wedge \phi_2$. Both recursive calls must succeed. By the induction hypothesis, their returned sets $\mathcal{B}_1$ and $\mathcal{B}_2$ contain accepted bridges and establish ϕ_1 and ϕ_2, respectively. Lemma 3 discharges ϕ under the merged set.

(iv) $\phi = \phi_1 \vee \phi_2$. At least one recursive call succeeds. The induction hypothesis establishes the disjunct of any selected successful branch with its accepted bridge set $\mathcal{B}_i$. Lemma 4 discharges ϕ with $\mathcal{B} = \mathcal{B}_i$.

If no logical decomposition applies, Algorithm 2 yields $(\mathcal{B}, \mathfrak{o})$, and Algorithm 1 recursively proves $\mathfrak{o}$. The induction hypothesis supplies its proof and accepted bridge set $\mathcal{B}_\mathfrak{o}$. Lemma 5 then discharges ϕ under $\mathcal{B} \cup \mathcal{B}_\mathfrak{o}$, whose bridges are accepted by that lemma and the induction hypothesis. Thus every successful return supplies a symbolic proof and accepted bridges. Taking $\phi = \mathfrak{g}$ gives Equation 1. □

D Dataset Details

FOLIO. FOLIO (Han et al., 2024) is a human-curated benchmark for natural-language reasoning with first-order logic. Each instance consists of a set of premises and a hypothesis whose logical status must be determined from the corresponding theory. We use the complete official validation split containing 204 instances. The split uses three labels: TRUE, FALSE, and UNKNOWN.

ProofWriter. ProofWriter (Tafjord et al., 2021) is a synthetically generated deductive-reasoning benchmark where each instance contains a natural-language theory composed of facts and rules, together with a query. The dataset supports controllable proof depths and both closed- and open-world reasoning. We use the complete 600-instance Depth-5 test split under the open-world assumption. The selected set uses three labels: TRUE, FALSE, and UNKNOWN.

PrOntoQA. PrOntoQA (Saparov & He, 2023) is a synthetic benchmark for controlled multi-step deductive reasoning over natural-language facts and rules. We use the 500 released test instances from the fictional-entity, five-hop configuration. This configuration uses two labels, TRUE and FALSE, and contains no UNKNOWN instances.

ProverQA. ProverQA (Qi et al., 2025) is a first-order logical-reasoning benchmark whose instances are generated and validated using symbolic theorem provers. It provides subsets of different difficulty levels according to their reasoning depth and structural complexity. We use the complete 500-instance hard subset of the development set. The split uses three labels: TRUE, FALSE, and UNKNOWN.

EntailmentBank. EntailmentBank (Dalvi et al., 2021) is a naturalistic multi-hop reasoning benchmark that provides supporting facts and structured entailment trees for deriving target hypotheses. We use the complete Task 1 test split containing 340 instances, where the relevant supporting facts are provided as premises. Since all hypotheses in this split are entailed, we evaluate EntailmentBank in a *prove-only* setting.

E Failure Analysis

We first manually examined failure cases and consolidated recurring causes into five categories; Table 5 presents representative examples across datasets. We then applied this taxonomy to all Gemini-3-Flash errors on FOLIO, covering both VPR failures and gold-UNKNOWN cases receiving decisive predictions. Each case was diagnosed by jointly inspecting the original natural language, selected autoformalization, prover trace, candidate bridges, and Gate decision. We classify failures caused by natural-language ambiguity or incorrect gold labels as dataset errors, rather than errors in the downstream reasoning pipeline. The distribution of failure causes is reported in Table 6.

Natural-language ambiguity. Ambiguity primarily arises from coordination and scope. For example, "Marvin cannot be from Earth and from Mars" admits both a *not-both* reading and the stronger *neither* reading. We also observe inconsistent inclusive versus exclusive interpretations of *either . . . or*, a known issue in FOLIO (Olausson et al., 2023).

Table 5: Representative errors used to construct the cross-dataset taxonomy.

Cause	Representative diagnosis	Dataset
Natural-language ambiguity	"Marvin cannot be from Earth and from Mars" is formalized as $\neg(From(m, e) \land From(m, r))$, meaning that Marvin cannot be from both places. The reference reasoning instead assumes $\neg From(m, e) \land \neg From(m, r)$, meaning that Marvin is from neither place.	*FOLIO*
Label error	The premise states that "KO is a mature company's stock", whereas the hypothesis is "KO is a company stock", The lexical inclusion is direct: a mature company's stock is a company stock. Accordingly, the gate correctly accepts $MatureCompanyStock(ko) \rightarrow CompanyStock(ko)$. However, the reference label is Unknown, this is therefore a dataset annotation error rather than gate overacceptance.	*FOLIO*
Autoformalization error	From "solid is a state of matter" and "an ice cube is a solid," the hypothesis says that an ice cube is solid in its physical state. The generated conclusion adds $PhysicalState(x, x)$, incorrectly making the ice cube itself its physical state; the otherwise sufficient $IceCube(x) \rightarrow Solid(x)$ cannot prove this extra conjunct.	*EntailmentBank*
Bridge/search coverage	Peter is taller than Michael, and Michael is taller than every other member of his class. Together with the stated transitivity rule, this implies that Peter is taller than those class members. Refuting "Peter is shorter than a man in Michael's class" additionally requires the asymmetry rule $\forall x \forall y, (Taller(x, y) \rightarrow \neg Taller(y, x))$. The current search operators do not propose such reversed-polarity universal bridges.	*FOLIO*
Gate overacceptance	The candidate bridge "Zayn loves nature" and "Zayn supports conservation." implies the obligation "Zayn cares about the environment," the gate accepts $(LovesNature(z) \land SupportsConservation(z)) \rightarrow CaresAboutEnvironment(z)$. This is plausible, but not an exception-free entailment; accepting it changes a gold-UNKNOWN case to FALSE.	*ProverQA*

Table 6: Percentage distribution of primary causes in the Gemini-3-Flash FOLIO audit. Percentages are rounded independently to two decimal places.

Primary cause	**Share** (%)
Natural-language ambiguity	27.03
Label error	24.32
Autoformalization error	24.32
Bridge/search coverage	16.22
Gate overacceptance	8.11

Label error. Some cases receive decisive reference labels even though neither the hypothesis nor its negation is derivable under open-world semantics. Similar labeling issues in FOLIO have been documented in prior work (Olausson et al., 2023; Brunello et al., 2026).

Autoformalization error. Recurring errors include malformed or solver-incompatible syntax, incorrect implication direction or connective scope, universal–existential quantifier confusion, and ontology or argument-role fragmentation. The last category is particularly difficult to repair downstream because the shared semantic structure has already been lost.

Bridge/search coverage. Some required repairs—notably common relational axioms—fall outside the current search operators, while valid pair bridges may be buried in a large candidate frontier. A targeted larger-budget diagnostic recovered 66.67% of these failures, all from the pair-bridge

subset. The remaining cases require bridge schemas or implicit assumptions that the current search procedure does not reliably propose.

Gate overacceptance. These errors occur when a plausible real-world tendency is promoted to an exception-free logical rule. Such bridges may sound natural in isolation but are not valid under the strict entailment required for formal reasoning, reflecting a semantic calibration error in the Gate.

F Extended Experimental Results

Table 7 extends the VPR evaluation in Table 2 with the available results for the remaining models.

Table 7: VPR (%) with additional underlying LLMs. The evaluation set size (N) for each dataset is shown below its column header.

LLM	Method	FOLIO $N = 136$	ProofWriter $N = 400$	PrOntoQA $N = 500$	ProverQA $N = 326$	EntailmentBank $N = 340$	Average
DeepSeek-V3.2	LINC	56.62	**100.00**	**100.00**	54.29	17.06	71.21
	Logic-LM	43.38	28.50	93.40	24.23	27.06	47.65
	POGB (Ours)	**76.47**	99.75	**100.00**	**75.77**	**42.35**	**81.90**
Qwen3.6-Flash	LINC	61.03	**99.75**	99.40	49.39	19.71	70.92
	Logic-LM	69.12	83.75	95.80	56.75	34.71	71.15
	POGB (Ours)	**77.21**	99.00	**100.00**	**66.87**	**58.24**	**83.25**
GPT-5.5	LINC	66.91	**100.00**	**100.00**	58.28	54.41	80.26
	Logic-LM	66.18	**100.00**	**100.00**	32.52	32.65	70.92
	POGB (Ours)	**77.94**	**100.00**	**100.00**	**82.21**	**60.58**	**86.96**

G Prompts

Listing 1 translates natural-language premises and conclusions into grounding FOL. Listing 2 evaluate candidate semantic bridges.

G.1 Prompt for Grounding Autoformalization

Listing 1: Grounding-autoformalization prompt.

```
Translate every premise and conclusion into first-order logic (FOL).
Ground every formal symbol in the natural-language input.
Read the complete input once before choosing symbols. Return JSON only.

CONTRACT
- Produce exactly one formula for each source sentence.
- Preserve scope, polarity, quantifiers, roles, and anaphora.
- Bind every variable. Use ∀ for generic statements and ∃ for
  local indefinites. Named entities are constants.
- Restricted forms include:
    every R is S:  ∀x (R(x) → S(x))
    some R is S:   ∃x (R(x) ∧ S(x))
- Reuse a predicate only when its meaning, arity, and roles agree.
- Keep argument order fixed. Use ⊕ only for explicit exclusion.
- Do not add background knowledge.

Allowed operators: ∧, ∨, ⊕, ¬, →, ↔, ∀, ∃.

GROUNDING OUTPUT
Return exactly these five top-level keys, in this order:
{
  "constant_table": [
    {"symbol":"...", "type":"...", "nl":"..."}
```

```
  ],
  "predicate_table": [
    {"symbol":"...", "arity":1, "arg_types":["..."],
     "template":"{1} is ..."}
  ],
  "premises": [
    {"source_sentence_id":"P1", "sentence":"...", "fol":"..."}
  ],
  "conclusions": [
    {"source_sentence_id":"C1", "sentence":"...", "fol":"..."}
  ],
  "variable_table": [
    {"source_sentence_id":"P1", "symbol":"x",
     "type":"...", "nl":"..."}
  ]
}

The three symbol tables jointly form the grounding table. In
variable_table, source_sentence_id and symbol identify a row;
type and nl are its only semantic fields. Do not add other keys.

ONE-SHOT EXAMPLE
Input:
Premises:
(1) Miroslav Venhoda was a Czech choral conductor who
    specialized in Renaissance and Baroque music.
(2) Any choral conductor is a musician.
(3) Some musicians love music.
(4) Miroslav Venhoda published a book in 1946 called
    Method of Studying Gregorian Chant.
Conclusions:
(1) Miroslav Venhoda loved music.
(2) A Czech person wrote a book in 1946.
(3) No choral conductor specialized in Renaissance.
Source sentence ids:
{"premises":["P1","P2","P3","P4"],
 "conclusions":["C1","C2","C3"]}

Output:
{
  "constant_table": [
    {"symbol":"miroslav", "type":"person",
     "nl":"Miroslav Venhoda"},
    {"symbol":"year1946","type":"year","nl":"1946"},
    ...
  ],
  "predicate_table": [
    {"symbol":"ChoralConductor", "arity":1,
     "arg_types":["person"],
     "template":"{1} is a choral conductor"},
    {"symbol":"Author", "arity":2,
     "arg_types":["person","book"],
     "template":"{1} wrote {2}"},
    ...
  ],
  "premises": [
    {"source_sentence_id":"P2",
     "sentence":"Any choral conductor is a musician.",
     "fol":"∀x (ChoralConductor(x) → Musician(x))"},
    {"source_sentence_id":"P3",
     "sentence":"Some musicians love music.",
     "fol":"∃x (Musician(x) ∧ Love(x,music))"},
    ...
  ],
  "conclusions": [
```

```
    {"source_sentence_id":"C2",
     "sentence":"A Czech person wrote a book in 1946.",
     "fol":"∃x (Czech(x) ∧ ...)"},
    {"source_sentence_id":"C3",
     "sentence":"No choral conductor specialized in Renaissance.",
     "fol":"¬∃x (ChoralConductor(x) ∧ ...)"},
    ...
  ],
  "variable_table": [
    {"source_sentence_id":"P2", "symbol":"x",
     "type":"person", "nl":"any choral conductor"},
    {"source_sentence_id":"P3", "symbol":"x",
     "type":"person", "nl":"some musicians"},
    {"source_sentence_id":"C2", "symbol":"x",
     "type":"person", "nl":"a Czech person"},
    {"source_sentence_id":"C2", "symbol":"y",
     "type":"book", "nl":"a book"},
    ...
  ]
}

NOW DO THE TASK
Input:
{INPUT_PREMISES_AND_CONCLUSIONS}

Source sentence ids:
{SOURCE_SENTENCE_IDS_JSON}

Return one JSON object with the five keys above, in that order.
Do not output markdown or explanatory text.
```

G.2 Prompt for Gate

Listing 2: Candidate bridge Gate prompt.

```
You are the semantic Gate for one localized candidate.

TASK
Assume only the displayed support statement(s). Decide whether they
jointly justify the target under its ordinary-language reading and
widely shared common sense.

This is a directional judgment, not a similarity judgment. Do not
accept merely because the target is plausible, topically related, or
useful for completing a proof.

Natural-language renderings are the semantic authority. Use formal
fields only to audit direction, negation, modality, argument roles,
entity identity, connective scope, and quantifier force.

ACCEPT
Accept when an ordinary reader would judge the displayed relation
true. This includes transparent paraphrase, synonymy, a stable subtype
relation, an unambiguous alias, a conventional semantic relation, or
well-established commonsense causal or functional knowledge.

REJECT
Reject when the relation reverses direction, changes polarity or
modality, swaps entities or roles, introduces unsupported identity,
strengthens the target, or requires an omitted fact. Also reject a
claim that contradicts established common knowledge or has a normal
counterexample under the displayed reading.
```

```
UNIVERSAL CLAIMS
If materialized_repair contains ∀, or arbitrary_variables is
present, the relation must hold for every admissible assignment.
A tendency, correlation, typical example, or fact about one named
entity is not enough. Preserve every arbitrary variable throughout.

SCOPE
Judge only this candidate. Do not reconstruct the complete problem,
infer its label, invent missing premises, inspect a proof state, or
reason from residual proof obligations. None of those are provided.

OUTPUT
Use "accept" for a supported relation and "reject" otherwise.
Use "defer" only for genuine linguistic ambiguity.

Candidate:
{"support":"local formula(s)", "target":"target formula",
 "materialized_repair":"proposed implication",
 "support_nl":"local support statement(s)",
 "target_nl":"target statement", "arbitrary_variables":[...]}

Return JSON only:
{"verdict":"accept|reject|defer", "confidence":0.0,
 "reason":"one short sentence"}
```