Logical-SAGE: A Logical Socratic Architecture for Guided Evolution in Neuro-Symbolic Reasoning

Jinlong Tian[1*], Jiang Yu[1*], Kewei Cheng[2], Yue He[3], Yunfei Wang[4]
Huibin Tan[1], Haotian Wang[1], Wenjing Yang[1], Shixuan Liu[1†]
[1]College of Computer Science and Technology, National University of Defense Technology
[2]Amazon
[3]Renmin University of China
[4]National Key Laboratory of Information Systems Engineering, National University of Defense Technology
szftandy@hotmail.com

Abstract

While Large Language Models (LLMs) excel at semantic understanding, they fundamentally lack the rigorous deductive reliability required for complex reasoning, often succumbing to hallucinations. Neuro-Symbolic (NeSy) approaches attempt to bridge this gap by offloading reasoning to formal solvers, yet they suffer from a critical *"Translation Fragility"* bottleneck: a single syntactic error in the generated logic program causes catastrophic execution failure. Existing paradigms typically adopt a *Generate-and-Hope* strategy, treating the solver as a binary judge and discarding imperfect code. In this paper, we introduce **Logical-SAGE** (**L**ogic-informed **S**ocratic **A**gent for **G**uided **E**volution), a novel Dual-Process architecture that shifts the paradigm to *Guided Evolution.* Synergizing a *System 1* neural reasoner with a *System 2* symbolic validator, our framework features a *Socratic Error Correction* mechanism. Instead of rejecting failed programs, Logical-SAGE treats solver error messages as pedagogical feedback, engaging in a dialectic loop to iteratively repair and evolve logic programs towards executability. We further introduce an Adaptive Fusion mechanism to balance rigorous proofs with robust neural intuition. Extensive experiments on five benchmarks (FOLIO, ProofWriter, ProntoQA, LogicalDeduction, AR-LSAT) demonstrate a breakthrough result: Logical-SAGE, powered by a parameter-efficient 8B model (Qwen-3-8B), achieves a new state-of-the-art average accuracy of 90.42%, significantly outperforming baselines relying on the massive GPT-4. This establishes that architectural innovation can supersede model scale in achieving faithful and grounded reasoning.

Introduction

The pursuit of rigorous logical reasoning in Artificial Intelligence faces a fundamental dichotomy. On one hand, Large Language Models (LLMs) have achieved mastery over linguistic fluency and semantic intuition, exhibiting emergent capabilities akin to Kahneman's *System 1* fast thinking (Wei et al. 2023; Kahneman 2011). However, they fundamentally rely on probabilistic correlation rather than axiomatic derivation, rendering them prone to hallucinations and logical inconsistencies, especially in multi-step deduction (Ji et al. 2023; Wei et al. 2022). On the other hand, symbolic systems (e.g., Theorem Provers, SAT Solvers) offer the precision and explainability of *System 2* slow thinking but lack the flexibility to parse the ambiguity of natural language.

To bridge this gap, Neuro-Symbolic (NeSy) AI has emerged as a promising paradigm, typically employing LLMs as semantic parsers to translate natural language into formal logic programs (e.g., FOL, Python) for deterministic execution (Olausson et al. 2023; Cheng et al. 2023; Liu et al. 2024). While theoretically sound, existing NeSy pipelines suffer from a critical bottleneck we term "Translation Fragility". Symbolic solvers are notoriously intolerant; a single syntactic mismatch or undefined predicate triggers a catastrophic execution failure. Current approaches predominantly adopt a "Generate-and-Hope" strategy: they generate logic programs and discard those that fail to compile, or rely on computationally expensive over-generation (Pan et al. 2023; Wang et al. 2022). Crucially, these methods treat the solver as a binary "judge" (Success/Fail) rather than an informative "guide", wasting the rich diagnostic signals hidden in execution error logs.

In this paper, we propose a paradigm shift from "Generate-and-Hope" to "Guided Evolution", as illustrated in Figure 1. Current frameworks (Figure 1a) fundamentally treat the symbolic solver as a binary judge—accepting perfect code while rejecting anything with minor syntax errors, leading to the "Translation Fragility" bottleneck. In contrast, Logical-SAGE (Figure 1b) redefines this interaction by treating the solver as a teacher. We introduce a novel architecture that mimics the human cognitive process of debugging. Just as a programmer learns from compiler errors, Logical-SAGE views solver failures not as terminal deadends but as pedagogical feedback.

Our framework introduces a Socratic Error Correction mechanism that establishes a dialectic loop between the LLM (the student) and the Symbolic Solver (the teacher). By recursively feeding error messages (e.g., `SyntaxError`, `NameError`) back into the context, the model iteratively refines its logic program until it aligns with the strict syntax

[*]These authors contributed equally.
[†]Corresponding author

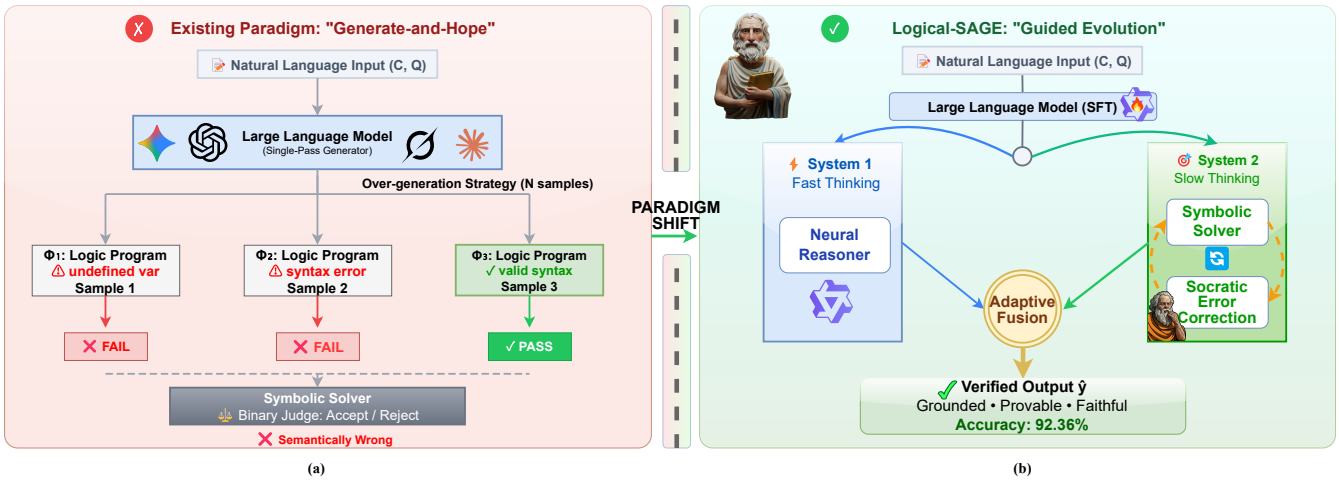

Figure 1: **The Paradigm Shift in Neuro-Symbolic Reasoning.** (a) Existing approaches suffer from *Translation Fragility*, employing a "Generate-and-Hope" strategy where solver errors lead to terminal failure. (b) **Logical-SAGE** introduces "Guided Evolution" via a Dual-Process architecture. It features a *Socratic Error Correction* loop that iteratively repairs logic programs based on solver feedback, synergizing with a neural reasoner for robust execution.

of the solver. To ensure this alignment is also semantically faithful to the original premises, we further incorporate a Back-Translation Verification protocol. Furthermore, recognizing that formalization is not always tractable (e.g., due to ambiguity), we design an Adaptive Fusion mechanism that synergizes this rigorous symbolic loop with a robust Neural Chain-of-Thought (CoT) fallback, ensuring the system is both logically sound and practically resilient.

The contributions of this work are three-fold:

1. **A Reflexive Neuro-Symbolic Architecture:** We propose Logical-SAGE, which integrates a dynamic Socratic debugging loop into the NeSy pipeline. This mechanism effectively resolves the inherent "Translation Fragility" bottleneck, thereby transforming execution errors into constructive supervision signals.
2. **Small Model, Giant Leap:** We demonstrate that architectural innovation can effectively supersede raw model scale. Logical-SAGE, powered by a parameter-efficient 8B model (Qwen-3-8B), achieves state-of-the-art performance, significantly outperforming baselines relying on the massive, closed-source GPT-4.
3. **Comprehensive Evaluation:** Extensive experiments on four diverse benchmarks (FOLIO, ProofWriter, PrOntoQA, LogicalDeduction, AR-LSAT) show that Logical-SAGE achieves an average accuracy of 90.42%, setting a new benchmark for faithfulness and executability in logical reasoning tasks.

Related Work

In this section, we review recent advancements in logical reasoning with Large Language Models (LLMs), categorizing them into prompt-based strategies, neuro-symbolic architectures, and instruction-tuning approaches. We then position our work, Logical-SAGE, within this landscape, highlighting how it addresses the critical limitations of existing frameworks regarding robustness and error correction.

Prompting Strategies for Logical Reasoning

The emergence of Chain-of-Thought (CoT) prompting (Wei et al. 2022) marked a significant milestone, enabling LLMs to perform multi-step deduction by generating intermediate reasoning traces. Several extensions have since been proposed to enhance logical consistency. Logic-of-Thought (LoT) (Liu et al. 2025) injects propositional logic derived from context into prompts to ground the reasoning process. Similarly, Tree of Thoughts (ToT) (Yao et al. 2023) and Graph of Thoughts (Besta et al. 2024) explore non-linear reasoning paths. However, despite these structural enhancements, purely prompt-based methods remain fundamentally bound by the probabilistic nature of LLMs (*System 1*). They lack a rigorous verification mechanism, leaving them prone to "hallucinations" and unfaithful reasoning, particularly in complex proofs where a single false step invalidates the entire chain (Maynez et al. 2020).

Neuro-Symbolic Reasoning Frameworks

To overcome the unreliability of pure LLMs, Neuro-Symbolic (NeSy) AI delegates the reasoning burden to deterministic solvers. Pipeline Approaches. A dominant paradigm involves using LLMs as semantic parsers to translate Natural Language (NL) into formal specifications (e.g., FOL, Python), which are then executed by external solvers. LINC Olausson et al. (2023) and NL2FOL (Lalwani et al. 2024) demonstrate that offloading deduction to theorem provers (e.g., Prover9) significantly boosts performance on benchmarks like FOLIO and ProofWriter. Logic-LM (Pan et al. 2023) further refines this by integrating a self-correction mechanism based on solver error messages. More recently, Logic-LM++ (Kirtania, Gupta, and Radhakrishna

2024) extends this with multi-step refinement and paired comparisons, while LTRAG (Hu et al. 2025) enhances the process via retrieval-augmented generation of similar logical templates. VERUS-LM (Callewaert, Vandevelde, and Vennekens 2025) and Aristotle (Xu et al. 2025) propose modular architectures that decompose complex problems into sub-tasks solved by specialized symbolic engines. Nevertheless, while these systems achieve high precision when successful, they frequently encounter the critical "Translation Fragility" bottleneck. Most existing refinement mechanisms (e.g., in Logic-LM) focus primarily on syntactic validity. If the semantic parsing is syntactically correct but logically misaligned with the premises, the solver yields a wrong answer without warning. Furthermore, systems like LINC lack a robust fallback: if the symbolic branch fails after refinement, the system returns no answer, leading to poor recall in real-world scenarios.

Logic-Enhanced Fine-Tuning and Optimization

Instead of relying solely on in-context learning, recent works explore fine-tuning LLMs to better understand logic. LogicLlama (Yang et al. 2024) and Symbol-LLM (Xu et al. 2024a) employ Supervised Fine-Tuning (SFT) on large-scale NL-FOL pairs to improve translation accuracy. LOGICPO (Viswanadha, Ghosal, and Aditya 2025) utilizes Direct Preference Optimization (DPO) to align LLMs with valid logical forms, reducing syntax errors. ProverGen (Qi et al. 2025) and Logic-Thinker (Wen et al. 2025) construct synthetic datasets with solver-verified reasoning chains to teach LLMs to "think" like a prover. Although fine-tuning significantly bolsters the base model's capabilities, it does not fully eliminate the inherent stochasticity of generation. Even with alignment techniques, fine-tuned models can still hallucinate under distribution shift if they lack a runtime verification loop (*System 2*) and a robust fusion strategy to validate their outputs.

Our proposed framework, Logical-SAGE, distinguishes itself from the aforementioned works in three key aspects: 1. Guided Evolution via Socratic Agent: Unlike Logic-LM's simple error feedback, our Socratic module engages in a deeper, pedagogical dialogue with the solver, treating error correction as an iterative learning process rather than just bug fixing. 2. Dual-Process Robustness: Unlike LINC or Logic-LM++ which rely heavily on the success of the symbolic path, Logical-SAGE maintains a parallel *System 1* (Neural CoT) branch. This ensures the system remains functional even when formalization is intractable (e.g., due to ambiguity), offering a "best of both worlds" solution. 3. Semantic Alignment: We introduce a Back-Translation Verification step (absent in most pipeline approaches), actively detecting semantic divergence before execution, thus addressing the "silent failure" mode of neuro-symbolic systems.

Method

In this section, we present Logical-SAGE, a unified neuro-symbolic framework designed to effectively bridge the gap between the linguistic flexibility of Large Language Models (LLMs) and the logical rigor of formal solvers. Drawing inspiration from the Dual-Process Theory in cognitive science (Evans, 2003), our architecture synergizes a **System 1** intuitive reasoner (neural branch) with a **System 2** analytical engine (symbolic branch), dynamically mediated by a novel Socratic Error Correction mechanism.

Problem Formulation

We consider a logical reasoning task defined by a dataset $\mathcal{D} = \{(C_i, Q_i, y_i)\}_{i=1}^{N}$, where C denotes the natural language context (premises), Q denotes the query, and y is the ground truth label. Our goal is to learn a mapping function $\mathcal{F} : (C, Q) \rightarrow \hat{y}$ that maximizes the accuracy while maintaining interpretability. We introduce an intermediate symbolic representation Z (e.g., First-Order Logic formulas), decomposing the problem into semantic parsing $P(Z|C, Q)$ and symbolic execution $P(y|Z)$.

The Logical-SAGE Architecture

The overall framework, as illustrated in Figure 2, follows a comprehensive five-stage pipeline (Input, Parse, Reason, Fuse, Output). It consists of three core components: (1) A Neural Semantic Parser that translates ambiguous natural language into executable domain-specific logic programs; (2) A Dual-Branch Inference Engine comprising a rigorous Symbolic Validator (incorporating a multi-step Socratic Refinement loop) and a robust Neural Reasoner; (3) An Adaptive Fusion Mechanism that dynamically integrates results based on execution states and semantic consistency.

Neural Semantic Parsing with Back-Translation Verification. The first module serves as the interface between natural language and formal logic.

Logic Program Generation. We employ a fine-tuned LLM (specifically, a parameter-efficient variant of Qwen-3-8B) as the generator G_θ. Given input (C, Q), the model generates a logic program Φ:

$$\Phi = G_\theta(C, Q, \mathcal{T}_{schema}) \tag{1}$$

where $\mathcal{T}_{schema}$ represents the domain-specific schema. As depicted in the *PARSE* module of Figure 1, our system adaptively maps different reasoning types to optimal formalisms: Deductive Reasoning is mapped to Logic Programming (LP), First-Order Logic tasks to FOL, Constraint Satisfaction problems to CSP, and Analytical Reasoning tasks to SAT solvers.

Semantic Alignment via Back-Translation. To mitigate the "Translation Gap" where generated formulas are syntactically correct but semantically divergent, we introduce a verification step. We employ a separate LLM instance to back-translate Φ into natural language C'_{rec} and compute a semantic consistency score S_{sem}:

$$S_{sem} = \text{Sim}(C, C'_{rec}) \tag{2}$$

If $S_{sem} < \tau$ (a predefined threshold), the generation is flagged for regeneration, ensuring the logic program faithfully represents the original premises.

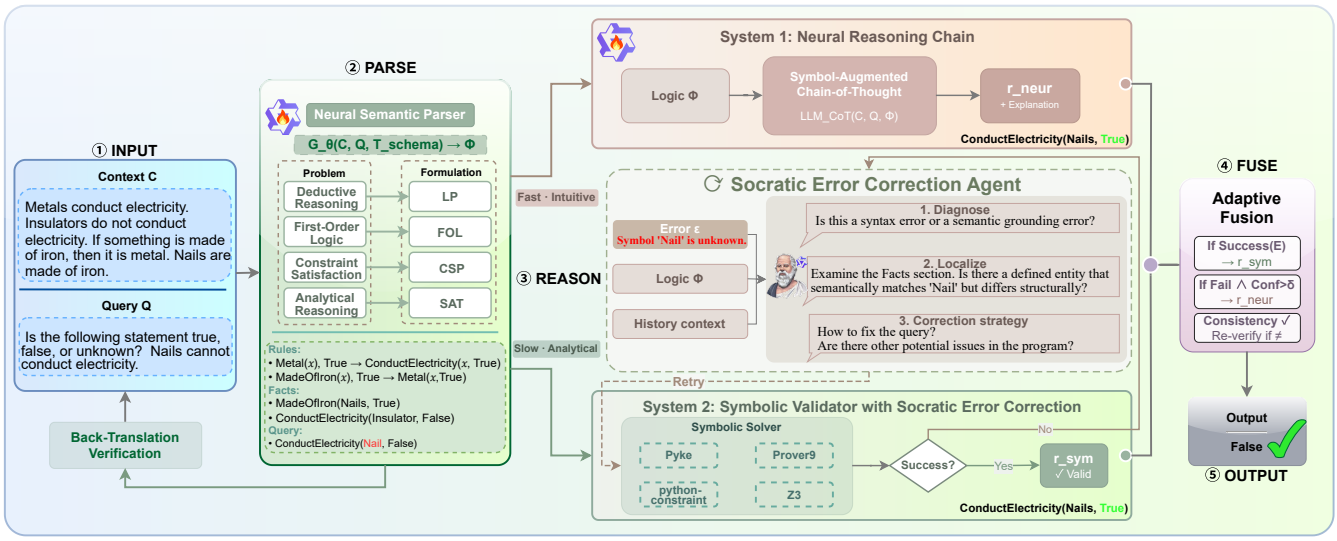

Figure 2: **Overview of the Logical-SAGE Framework.** The pipeline consists of five stages: ① Input encoding; ② **Neural Semantic Parsing**, which translates natural language (C, Q) into logic programs (Φ); ③ **Dual-Process Reasoning**, featuring a *System 1* Neural Chain-of-Thought branch and a *System 2* Symbolic Validator. The symbolic branch incorporates a **Socratic Error Correction** loop that utilizes solver feedback (e.g., error ϵ) to iteratively diagnose and fix semantic misalignments (e.g., resolving the entity mismatch between *Nail* and *Nails*); ④ **Adaptive Fusion**, which selects the final prediction $\hat{y}$ by prioritizing valid symbolic execution (r_{sym}) or falling back to high-confidence neural reasoning (r_{neur}); and ⑤ Final Output generation.

System 1: Neural Reasoning Chain (The Robust Intuition) Representing the "fast thinking" component of our dual-process architecture, this module serves as a robust engine for scenarios where strict formalization is incomplete or intractable (e.g., questions requiring external commonsense). Here, we utilize the LLM to perform **Symbolic-Augmented Chain-of-Thought (CoT)**.

$$\hat{y}_{neur}, \text{Expl} = \text{LLM}_{CoT}(C, Q, \Phi) \tag{3}$$

Unlike standard CoT, we explicitly inject the generated logic program Φ back into the prompt as a structured context. This forces the neural reasoner to ground its intuitive deductions in the formal skeleton it previously generated, thereby reducing hallucinations while leveraging its inherent flexibility to interpret ambiguous predicates that might otherwise crash a deterministic solver.

System 2: Symbolic Validator with Socratic Error Correction Parallel to the neural pathway, this module embodies the deliberative "slow thinking" analytical process, aiming to derive undeniably correct conclusions by strictly executing Φ via a deterministic solver E (e.g., Prover9 for FOL, Z3 for SMT). However, traditional symbolic systems suffer from inherent "fragility"—a single syntactic mismatch or undefined predicate causes catastrophic execution failure $(\perp)$. To effectively address this, we implement a **Socratic Error Correction (SEC)** mechanism.

Instead of treating solver failures as terminal states, the SEC mechanism reinterprets the solver's error message ϵ as pedagogical feedback, thereby establishing a robust iterative refinement loop. The process begins with an execution attempt $E(\Phi)$; If this yields a runtime error, the system dynamically constructs a comprehensive diagnostic prompt $\mathcal{P}_{refine} = (\Phi, \epsilon, \mathcal{I}_{history})$, which encapsulates the original logic program, the specific error signal, and a history of prior failed attempts to prevent redundant mistakes. Guided by this context, the LLM adopts the persona of a "Socratic Student", engaging in reflexive self-correction to generate a refined program Φ_{t+1}. This cycle continues recursively—mimicking a human programmer's debugging workflow—until the solver returns a valid result or the maximum iteration limit T_{max} is reached. This approach ensures that the system strictly adheres to logical axioms while possessing the resilience to iteratively correct syntactic and semantic misalignments based on compiler feedback.

Adaptive Result Fusion The final decision $\hat{y}$ is derived via a hierarchical gating mechanism designed to balance the provable soundness of *System 2* with the probabilistic coverage of *System 1*. Let r_{sym} denote the deterministic result from the Symbolic Validator and r_{neur} denote the probabilistic prediction from the Neural Reasoning Chain.

To synthesize these distinct modalities, we formulate the fusion strategy as a cascade decision process. The primary gate is the execution status of the logic program Φ. While *System 2* is prioritized for its rigorous derivation, *System 1* serves as a critical fallback when formalization is intractable. The fusion function is defined as:

$$\hat{y} = \begin{cases} r_{\text{sym}}, & \text{if } \mathcal{S}(\Phi) = \text{Success} \wedge r_{\text{sym}} = r_{\text{neur}}; \\ \mathcal{V}_{\text{verify}}(r_{\text{sym}}, r_{\text{neur}}), & \text{if } \mathcal{S}(\Phi) = \text{Success} \wedge r_{\text{sym}} \neq r_{\text{neur}}; \\ r_{\text{neur}}, & \text{if } \mathcal{S}(\Phi) = \text{Fail} \wedge \text{Conf}(r_{\text{neur}}) > \delta; \\ \text{Unknown}, & \text{otherwise.} \end{cases} \tag{4}$$

where $\mathcal{S}(\cdot)$ represents the solver's execution status, and $\text{Conf}(\cdot)$ computes the confidence score of the neural generation based on average token log-probabilities.

A critical vulnerability in neuro-symbolic systems is the *Silent Failure* mode, where a logic program is syntactically valid (executable) but semantically unfaithful to the natural language premises. To mitigate this, we introduce a verification protocol for critical instances. When *System 2* executes successfully but yields a contradiction with *System 1*'s high-confidence prediction (i.e., $r_{sym} \neq r_{neur}$), the system triggers a **Re-verification** step. Crucially, we re-engage the Socratic Error Correction module to resolve this conflict. Functioning as a semantic arbiter rather than just a code debugger, the Socratic Correction Module is prompted to determine whether the discordance stems from a subtle formalization error in Φ or a hallucination in the neural chain, ensuring that the rigorous output is not only valid but also grounded.

This hybrid architecture ensures that Logical-SAGE benefits from the "best of both worlds": the provable correctness of symbolic logic and the semantic robustness of large language models.

Experiments

In this section, we provide a comprehensive empirical evaluation of Logical-SAGE. We aim to answer the following research questions:

- **RQ1 (Performance):** How does Logical-SAGE compare against state-of-the-art LLMs and existing neuro-symbolic frameworks across diverse reasoning tasks?
- **RQ2 (Socratic Mechanism):** How does the Socratic Error Correction module mitigate "Translation Fragility" and impact the executability of logic programs?
- **RQ3 (Dual-Process Necessity):** Is the Dual-Process Fusion mechanism essential? How does the synergy between *System 1* (Neural) and *System 2* (Symbolic) compare to relying on either modality alone?

Experimental Setup

Datasets We evaluate our framework on four standard logical reasoning benchmarks, comprehensively covering a broad spectrum of reasoning types:

- ProntoQA (Saparov and He 2022): A synthetic dataset focusing on multi-hop deduction with purely fictional predicates devoid of prior knowledge.
- ProofWriter (Tafjord, Mishra, and Clark 2021): Requires generating explicit natural language proofs over N-hop rules, testing strict deductive capabilities.
- FOLIO (Han et al. 2024): A challenging dataset with expert-written First-Order Logic problems containing complex linguistic nuances and real-world knowledge.
- LogicalDeduction (Srivastava et al. 2023): A task from BigBench comprising Constraint Satisfaction Problems (CSP) that require deducing the order of objects from a minimal set of conditions.
- AR-LSAT (Zhong et al. 2022): Analytical Reasoning questions from the Law School Admission Test, representing highly complex real-world constraint satisfaction problems (CSP) in out-of-distribution scenarios.

Baselines To rigorously evaluate the effectiveness of Logical-SAGE, we compare it against a comprehensive set of baselines, categorized into two paradigms based on their reasoning methodology:

1. **Pure Neural Prompting Baselines**: These methods rely primarily on the internal reasoning capabilities of Large Language Models via advanced prompting strategies.
   - **GPT-4 (Standard)** and **GPT-4-CoT**: We report the performance of `GPT-4` using both direct answering and Chain-of-Thought (CoT) prompting (Wei et al. 2022) to represent the upper bound of pure neural reasoning.
   - **SymbCOT** (Xu et al. 2024b): A strong baseline that augments the standard Chain-of-Thought with symbolic execution traces to guide the generation process, representing the state-of-the-art in prompt-based logical enhancement.
2. **Neuro-Symbolic Frameworks**: These methods, like ours, explicitly integrate LLMs with external solvers but differ in their architecture and refinement strategies.
   - **Logic-LM** (Pan et al. 2023): The foundational baseline that employs a standard generate-then-verify loop based on error feedback using GPT-4.
   - **Logic-LM++** (Kirtania, Gupta, and Radhakrishna 2024): An enhanced version of Logic-LM that incorporates multi-step refinement and paired candidate comparisons to improve translation accuracy.
   - **LTRAG** (Hu et al. 2025): A Retrieval-Augmented Generation (RAG) framework that retrieves similar logical templates to serve as demonstrations to assist the solver, particularly effective for tasks requiring external knowledge (e.g., AR-LSAT).
   - **VERUS** (Callewaert, Vandevelde, and Vennekens 2025): A recent state-of-the-art modular neuro-symbolic solver that uses a specialized ensemble approach decomposing problems into sub-tasks for high-precision deduction.

Implementation Details To demonstrate the efficiency and democratized potential of our framework, we restrict the neural backbone to Qwen-3-8B [1], a parameter-efficient open-source LLM. Specifically, the Neural Semantic Parser, the Socratic Error Correction Agent, and the Neural Reasoning Chain (*System 1*) are all instantiated using the same Qwen-3-8B model, fine-tuned via LoRA on the training splits of the respective datasets. This unified setup ensures that our performance gains stem from architectural innovation rather than raw model scale. For symbolic execution, we align specific deterministic solvers with the logical topology of each benchmark: deductive reasoning tasks

[1] https://huggingface.co/Qwen/Qwen3-8B

Table 1: Accuracy (%) comparison on five logical reasoning benchmarks. **Logical-SAGE (SFT)**, despite using a small `Qwen-3-8B` backbone, achieves new state-of-the-art results across all datasets, significantly outperforming GPT-4 based baselines. Best results are bolded; best baseline results are underlined.

Method	PrOntoQA	ProofWriter	FOLIO	LogicalDed.	AR-LSAT	Avg.
Pure Neural Prompting Baselines						
GPT-4[†] (Standard)	77.40	52.67	69.11	71.33	33.33	60.77
GPT-4-CoT[†]	<u>98.79</u>	68.11	70.58	75.25	35.06	69.56
SymbCOT[*]	99.60	82.50	<u>83.33</u>	<u>93.00</u>	43.91	80.47
Neuro-Symbolic Baselines						
Logic-LM (GPT-4)[*]	83.20	78.80	68.55	87.63	23.40	68.32
LOGIC-LM++ (GPT-4)[*]	—	79.66	84.80	—	46.32	—
LTRAG[*]	—	—	78.57	—	<u>68.40</u>	—
VERUS[*]	95.80	<u>93.83</u>	78.43	88.67	68.36	<u>85.02</u>
Ours						
Logical-SAGE (Qwen3-8B)	91.60	81.30	73.20	91.80	53.60	78.30
Logical-SAGE (SFT Qwen3-8B)	**100.00**	**95.83**	**86.12**	**99.00**	**71.15**	**90.42**

Note: "—" denotes results not reported. [†] indicates results from (Xu et al. 2024b). [*] indicates results from original paper.

Table 2: Comparison of **Execution Rate (Er, %)** and **Execution Accuracy (Ea, %)** against SOTA baselines. **Logical-SAGE (SFT)** achieves near-perfect execution rates across most tasks, virtually eliminating the "Translation Fragility" issue that plagues GPT-4 based methods like Logic-LM (e.g., 39.8% Er on AR-LSAT).

Model	PrOntoQA		ProofWriter		FOLIO		LogicalDed.		AR-LSAT	
	Er	Ea	Er	Ea	Er	Ea	Er	Ea	Er	Ea
Logic-LM (GPT-4)	100.0	83.2	99.0	79.6	85.8	79.9	100.0	87.6	39.8	58.8
LOGIC-LM++	99.0	79.6	—	—	86.7	85.8	—	—	32.0	66.2
VERUS	98.2	97.6	99.0	94.8	100.0	78.4	99.3	89.3	**98.7**	69.3
Logical-SAGE (Qwen3-8B)	95.3	96.1	97.0	83.8	93.0	78.7	97.0	94.6	75.3	71.0
Logical-SAGE (SFT Qwen3-8B)	**100.0**	**100.0**	**100.0**	**95.8**	**100.0**	**86.1**	**100.0**	**99.0**	95.6	**74.4**

Note: Er: Execution Rate; Ea: Exact Match Accuracy of executed programs. "—" indicates data not reported.

(PrOntoQA and ProofWriter) utilize the `Pyke` logic programming engine; First-Order Logic problems (FOLIO) are offloaded to the `Prover9` automated theorem prover; Constraint satisfaction tasks (LogicalDeduction) employ the `python-constraint` library; and complex analytical reasoning tasks (AR-LSAT) are solved using the `Z3` SMT solver. The Socratic Agent operates with a maximum refinement depth of $T_{max} = 3$. All experiments were conducted on a server with a single NVIDIA H100 (80GB) GPU.

Main Results

Overall Accuracy Table 1 presents the comprehensive comparison results across five logical reasoning benchmarks. The most striking finding is that **Logical-SAGE (SFT)** establishes a new state-of-the-art, achieving a remarkable average accuracy of **90.42%**. This significantly surpasses the strongest baseline, VERUS (85.02%), by a margin of 5.40%. Crucially, this performance is achieved using a parameter-efficient `Qwen-3-8B` backbone, whereas top-tier baselines like SymbCOT and Logic-LM++ rely on the massive, closed-source GPT-4. This empirically validates our hypothesis that for rigorous logical tasks, architectural soundness—specifically the integration of a specialized *System 2* parser with Socratic error correction—is more deterministic than raw model scale. We demonstrate that a compact, well-tuned neuro-symbolic system can effectively "punch above its weight," outperforming general-purpose giant models by substantial margins.

The performance dominance is particularly evident in complex, linguistically nuanced scenarios. On the FOLIO benchmark, which involves intricate First-Order Logic translation, Logical-SAGE (SFT) achieves **86.12%**, surpassing the previous best of 84.80% set by Logic-LM++. Similarly, on the notoriously difficult AR-LSAT, our model outperforms even retrieval-augmented methods like LTRAG (71.15% vs. 68.40%), suggesting effective internalization of logical constraints without external retrieval. Furthermore, the comparison between the base Logical-SAGE (78.30%) and the SFT variant (90.42%) highlights the pivotal role of supervised fine-tuning. The substantial 12% improvement indicates that SFT transforms the 8B model from a capable generalist into a precision engine, aligning its outputs strictly with the syntactic requirements of symbolic solvers and enabling near-perfect execution rates on deductive tasks like PrOntoQA and LogicalDeduction.

Executability and Reliability Analysis To deeply investigate how Logical-SAGE resolves the "Translation Fragility" bottleneck, we compare the **Execution Rate (Er)**—the percentage of generated programs that compile without syn-

Table 3: Analysis of the Socratic Error Correction (SEC) effectiveness. We report the **Execution Rate (Er)** and **Exact Match Accuracy (Ea)** before and after applying the Socratic loop.

Configuration	PrOntoQA		ProofWriter		FOLIO		LogicalDed.		AR-LSAT	
	Er	Ea	Er	Ea	Er	Ea	Er	Ea	Er	Ea
w/o SEC	78.5	91.2	72.4	79.5	58.6	70.1	84.2	89.5	31.5	55.8
w/ SEC (Ours)	**100.0**	**100.0**	**100.0**	**95.8**	**100.0**	**86.1**	**100.0**	**99.0**	**95.6**	**74.4**
Improvement (Δ)	+21.5	+8.8	+27.6	+16.3	+41.4	+16.0	+15.8	+9.5	+64.1	+18.6

Note: Er: Execution Rate (%), measures the percentage of problems that produce syntactically valid programs; Ea: Exact Match Accuracy (%), measures the percentage of problems solved correctly. Improvements are calculated as (w/ SEC) - (w/o SEC).

Table 4: Ablation study on the necessity of the Dual-Process architecture. We report **Accuracy (%)** for *System 1* alone, *System 2* alone, and the full Logical-SAGE framework. Performance drops relative to the full model are shown in red. The results confirm that while *System 2* drives the majority of performance gains, *System 1* acts as a critical safety net, particularly for tasks with high linguistic ambiguity.

Configuration	PrOntoQA	ProofWriter	FOLIO	LogicalDed.	AR-LSAT	Avg.
System 1 Only (Neural CoT)	88.50 (↓11.5%)	86.20 (↓10.0%)	79.40 (↓7.8%)	86.3 (↓12.8%)	60.50 (↓15.0%)	80.18 (↓11.3%)
System 2 Only (Symbolic)	98.64 (↓1.4%)	87.50 (↓8.7%)	81.20 (↓5.7%)	91.30 (↓7.8%)	61.80 (↓13.14%)	84.09 (↓7.0%)
Full Model (Logical-SAGE)	**100.00**	**95.83**	**86.12**	**99.00**	**71.15**	**90.42**

Note: Percentages in red with ↓ arrows indicate performance degradation compared to the full Logical-SAGE model. *System 1* refers to the neural chain-of-thought module; *System 2* refers to the symbolic reasoning engine.

tax errors—and the **Execution Accuracy (Ea)**—the correctness of those executed programs—against leading neuro-symbolic baselines. The results are summarized in Table 2.

The most critical insight derived from Table 2 is that Logical-SAGE (SFT) effectively solves the crash-proneness inherent in symbolic generation. This capability is most strikingly illustrated on the complex AR-LSAT benchmark, where GPT-4 based baselines suffer from catastrophic failure rates (Er $< 40\%$). In contrast, our SFT model achieves an impressive Er of 95.6%.

Crucially, Logical-SAGE demonstrates superior reliability across different logic types. On four out of five benchmarks (PrOntoQA, ProofWriter, LogicalDeduction, and FOLIO), the model achieves a perfect **100% Execution Rate**. This establishes that the Socratic Error Correction mechanism successfully eliminates syntax errors even in complex First-Order Logic tasks. Furthermore, on the challenging FOLIO dataset, Logical-SAGE (SFT) demonstrates superior semantic fidelity compared to the state-of-the-art modular solver VERUS. While both models achieve 100% executability, our model significantly outperforms VERUS in **Execution Accuracy** (**86.1%** vs. 78.4%). This distinction validates that Logical-SAGE generates not merely *compilable* code, but *logically correct* proofs that faithfully capture the problem semantics.

Sensitivity Analysis: The Efficiency-Performance Trade-off

To determine the optimal configuration for real-world deployment, we analyze the critical trade-off between reasoning rigor and computational cost by plotting the Average Accuracy against Inference Latency across varying refinement depths (T_{max}). As visualized in **Figure 3**, the system exhibits a steep performance ascent from 78.3% at the baseline ($T = 0$) to a peak of 92.36% at $T = 3$, reflecting the Socratic Agent's ability to effectively resolve fatal syntax errors and semantic ambiguities through iterative feedback. However, this accuracy gain eventually saturates, with $T = 4$ yielding negligible marginal improvements ($< 0.2\%$). Conversely, the computational cost follows a strict linear growth trajectory, adding approximately 15-20 seconds of latency per iteration due to the combined overhead of symbolic execution and LLM re-generation. With

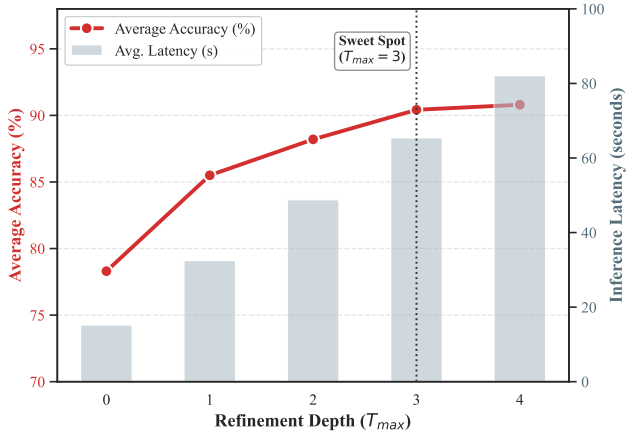

Figure 3: **Efficiency-Performance Trade-off.** The dual-axis plot compares Average Accuracy (line) against Inference Latency (bars).

the total latency exceeding 80 seconds at $T = 4$, we identify $T_{max} = 3$ as the operational "sweet spot", striking the ideal balance that maximizes logical faithfulness within a practical time budget ($\sim$65s), thus validating the efficiency of our Guided Evolution paradigm.

Ablation Study

Effectiveness of Socratic Error Correction To rigorously quantify the contribution of the Socratic Error Correction (SEC) module, we evaluate the quality of generated logic programs using two key metrics: **Execution Rate (Er)** and **Execution Accuracy (Ea)**. As shown in Table 3, the baseline configuration without SEC exposes a severe "Translation Fragility" inherent to the compact Qwen-3-8B backbone. While the model maintains decent executability ($\sim$80%) on straightforward deductive tasks like PrOntoQA, it falters significantly on complex benchmarks. Most notably, on AR-LSAT, which demands intricate Python Z3 constraint formulations, the execution rate plummets to a catastrophic 31.5%. This indicates that in a single-pass generation setting, the small model is prone to hallucinations, frequently generating undefined variables or violating rigid syntactic structures that render the symbolic solver useless.

Integrating the Socratic Error Correction loop fundamentally reverses this trend, transforming the system's reliability. We observe a dramatic surge in Execution Rate across all datasets. Notably, on PrOntoQA, ProofWriter, FOLIO, and LogicalDeduction, the SEC mechanism achieves a perfect 100% Execution Rate, completely eliminating compilation failures. The impact is most profound on the complex AR-LSAT benchmark, where executability leaps from 31.5% to 95.6% (a massive **+64.1 point** improvement). This confirms that the Socratic agent effectively interprets solver error logs (e.g., `NameError`, `SyntaxError`) to iteratively "debug" the code. More critically, the benefits extend beyond mere compilation; the Exact Match Accuracy (Ea) also sees substantial gains (e.g., +18.6% on AR-LSAT). This suggests that the SEC process transcends surface-level syntax repair. By forcing the model to reflect on execution feedback, it rectifies deeper semantic misalignments—such as correcting boundary conditions (e.g., swapping $>$ with $\geq$) or fixing entity mappings—thereby ensuring the final symbolic proofs are not only executable but also logically sound.

Necessity of the Dual-Process Architecture To validate the architectural premise of Logical-SAGE—that combining neural intuition (*System 1*) with symbolic rigor (*System 2*) is superior to relying on either modality alone—we conduct an ablation study focusing on the final accuracy across all benchmarks. As detailed in Table 4, we compare the full framework against two decoupled variants: "System 1 Only", which relies solely on the fine-tuned Neural Reasoner via Chain-of-Thought, and "System 2 Only", which utilizes the Symbolic Validator (with SEC enabled) but treats execution failures as incorrect answers without neural backoff.

The empirical results strongly advocate for the synergistic design. Relying solely on *System 1* yields an average accuracy of 80.18%. While this represents a strong neural baseline, it still lags behind the Full Model by over 10%, indicating that pure probabilistic reasoning lacks the precision required for rigorous tasks. Conversely, *System 2* alone achieves a baseline of 84.09%, proving that symbolic execution is a primary driver of our state-of-the-art performance but still insufficient on its own. However, a critical observation is that on FOLIO, *System 2* alone (81.20%) underperforms the Full Model (86.12%), revealing that the symbolic validator is not infallible. *System 2* is inherently bounded by the "Translation Fragility" bottleneck; similarly, on AR-LSAT, the accuracy caps at 61.80%. The Full Model effectively bridges these gaps, lifting the AR-LSAT performance to 71.15% and the overall average to 90.42%. This demonstrates that the Adaptive Fusion mechanism successfully leverages *System 1* as a robust semantic backoff, salvaging cases where the strict symbolic parser fails (both in FOLIO and AR-LSAT) while maintaining the rigorous guarantees of the solver whenever possible.

Conclusion

In this paper, we addressed the "Translation Fragility" bottleneck inherent in neuro-symbolic reasoning, where the intolerance of formal solvers to minor errors often cripples the potential of Large Language Models (LLMs). We introduced Logical-SAGE, a dual-process architecture that harmonizes the linguistic intuition of neural networks with the axiomatic rigor of symbolic logic. By modeling the reasoning process not as a linear translation but as a reflexive dialogue, our framework employs a novel Socratic Error Correction mechanism. This approach fundamentally shifts the paradigm from a *generate-and-hope* strategy to a *generate-and-correct* loop, where solver feedback acts as a pedagogical signal to guide the LLM toward valid execution. Empirical evaluations across multiple benchmarks demonstrate that Logical-SAGE not only establishes a new state-of-the-art in accuracy but, more importantly, achieves high faithfulness by ensuring that answers are grounded in verifiable logical proofs rather than probabilistic hallucinations.

Our findings suggest that the path to reliable General Artificial Intelligence lies in systems that can introspect and self-correct. Logical-SAGE serves as a proof-of-concept for Grounded AI, reducing the "Black Box" opacity of deep learning by providing transparent, executable reasoning chains for high-stakes applications. However, we acknowledge that our current reliance on standard First-Order Logic (FOL) imposes limits on expressivity, particularly when dealing with the probabilistic, temporal, or fuzzy concepts common in human communication. Consequently, several promising avenues for future research emerge. First, we plan to extend the Socratic Agent to support more diverse formalisms, such as Probabilistic Soft Logic or Temporal Logic, to capture the nuances of dynamic real-world environments. Second, to mitigate the inference latency introduced by iterative refinement, we envision integrating Reinforcement Learning (RL) to fine-tune the semantic parser end-to-end, internalizing the socratic feedback directly into the model's weights. Finally, we aim to generalize this guided evolution paradigm beyond logic puzzles to broader code generation and algorithmic reasoning tasks, striving for a universally robust neuro-symbolic interface.

References

Besta, M.; Blach, N.; Kubicek, A.; Gerstenberger, R.; Podstawski, M.; Gianinazzi, L.; Gajda, J.; Lehmann, T.; Niewiadomski, H.; Nyczyk, P.; et al. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, 17682–17690.

Callewaert, B.; Vandevelde, S.; and Vennekens, J. 2025. Verus-lm: a versatile framework for combining llms with symbolic reasoning. *arXiv preprint arXiv:2501.14540*.

Cheng, Z.; Xie, T.; Shi, P.; Li, C.; Nadkarni, R.; Hu, Y.; Xiong, C.; Radev, D.; Ostendorf, M.; Zettlemoyer, L.; et al. 2023. Binding Language Models in Symbolic Languages. In *The Eleventh International Conference on Learning Representations*.

Han, S.; Schoelkopf, H.; Zhao, Y.; Qi, Z.; Riddell, M.; Zhou, W.; Coady, J.; Peng, D.; Qiao, Y.; Benson, L.; et al. 2024. FOLIO: Natural Language Reasoning with First-Order Logic. In *EMNLP*.

Hu, R.; Lin, S.; Xiu, Y.; and Liu, Y. 2025. LTRAG: Enhancing autoformalization and self-refinement for logical reasoning with thought-guided RAG. In *Findings of the Association for Computational Linguistics: ACL 2025*, 2483–2493.

Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y. J.; Madotto, A.; and Fung, P. 2023. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12): 1–38.

Kahneman, D. 2011. *Thinking, fast and slow*. macmillan.

Kirtania, S.; Gupta, P.; and Radhakrishna, A. 2024. LOGIC-LM++: Multi-step refinement for symbolic formulations. In *Proceedings of the 2nd Workshop on Natural Language Reasoning and Structured Explanations (@ ACL 2024)*, 56–63.

Lalwani, A.; Kim, T.; Chopra, L.; Hahn, C.; Jin, Z.; and Sachan, M. 2024. Autoformalizing Natural Language to First-Order Logic: A Case Study in Logical Fallacy Detection. *arXiv preprint arXiv:2405.02318*.

Liu, S.; Fan, C.; Cheng, K.; Wang, Y.; Cui, P.; Sun, Y.; and Liu, Z. 2024. Inductive meta-path learning for schema-complex heterogeneous information networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Liu, T.; Xu, W.; Huang, W.; Zeng, Y.; Wang, J.; Wang, X.; Yang, H.; and Li, J. 2025. Logic-of-thought: Injecting logic into contexts for full reasoning in large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 10168–10185.

Maynez, J.; Narayan, S.; Bohnet, B.; and McDonald, R. 2020. On faithfulness and factuality in abstractive summarization. *arXiv preprint arXiv:2005.00661*.

Olausson, T.; Gu, A.; Lipkin, B.; Zhang, C.; Solar-Lezama, A.; Tenenbaum, J.; and Levy, R. 2023. 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*, 5153–5176.

Pan, L.; Albalak, A.; Wang, X.; and Wang, W. 2023. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 3806–3824.

Qi, C.; Ma, R.; Li, B.; Du, H.; Hui, B.; Wu, J.; Laili, Y.; and He, C. 2025. Large language models meet symbolic provers for logical reasoning evaluation. *arXiv preprint arXiv:2502.06563*.

Saparov, A.; and He, H. 2022. Language Models Are Greedy Reasoners: A Systematic Formal Analysis of Chain-of-Thought. In *The Eleventh International Conference on Learning Representations*.

Srivastava, A.; Rastogi, A.; Rao, A.; Shoeb, A. A. M.; Abid, A.; Fisch, A.; Brown, A. R.; Santoro, A.; Gupta, A.; Garriga-Alonso, A.; et al. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on machine learning research*.

Tafjord, O.; Mishra, B. D.; and Clark, P. 2021. ProofWriter: Generating Implications, Proofs, and Abductive Statements over Natural Language.

Viswanadha, K.; Ghosal, D.; and Aditya, S. 2025. LOGICPO: Efficient Translation of NL-based Logical Problems to FOL using LLMs and Preference Optimization. *arXiv preprint arXiv:2506.18383*.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q. V.; Chi, E. H.; Narang, S.; Chowdhery, A.; and Zhou, D. 2022. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *The Eleventh International Conference on Learning Representations*.

Wei, J.; Tay, Y.; Bommasani, R.; Raffel, C.; Zoph, B.; Borgeaud, S.; Yogatama, D.; Bosma, M.; Zhou, D.; Metzler, D.; et al. 2023. Emergent Abilities of Large Language Models. *Transactions on Machine Learning Research*.

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

Wen, C.; Cheng, Q.; Wang, S.; Liu, Z.; Zhao, D.; and Liang, L. 2025. Logic-Thinker: Teaching Large Language Models to Think more Logically. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, 12955–12969.

Xu, F.; Wu, Z.; Sun, Q.; Ren, S.; Yuan, F.; Yuan, S.; Lin, Q.; Qiao, Y.; and Liu, J. 2024a. Symbol-llm: Towards foundational symbol-centric interface for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 13091–13116.

Xu, J.; Fei, H.; Luo, M.; Liu, Q.; Pan, L.; Wang, W. Y.; Nakov, P.; Lee, M.-L.; and Hsu, W. 2025. 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 (Volume 1: Long Papers)*, 3052–3075.

Xu, J.; Fei, H.; Pan, L.; Liu, Q.; Lee, M.-L.; and Hsu, W. 2024b. Faithful logical reasoning via symbolic chain-of-thought. *arXiv preprint arXiv:2405.18357*.

Yang, Y.; Xiong, S.; Payani, A.; Shareghi, E.; and Fekri, F. 2024. Harnessing the power of large language models for natural language to first-order logic translation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 6942–6959.

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

Zhong, W.; Wang, S.; Tang, D.; Xu, Z.; Guo, D.; Chen, Y.; Wang, J.; Yin, J.; Zhou, M.; and Duan, N. 2022. Analytical reasoning of text. In *Findings of the Association for Computational Linguistics: NAACL 2022*, 2306–2319.