Agentic Geometry Problem Solving via Human-like Parallel Bidirectional Reasoning

Xiaokai Zhang[1] **Ruiqing Xia**[1] **Liang Chen**[1] **Zhenhai Sun**[1] **Yuchang Yang**[1]
Tuo Leng[1,2,*]

[1] School of Computer Engineering and Science, Shanghai University
[2] School of Future Technology, Shanghai University

Abstract

Formal Geometric Problem Solving requires every step of reasoning within a strict logical framework, and has been a core challenge in artificial intelligence. This paper presents a unified neuro-symbolic reasoning framework, named **Reflect Agent Verified Solve** (RAVS), which deeply integrates large language models, agent architectures, and formal symbolic solvers. The core innovations of RAVS lie in two parts. First, we propose a role-separated neuro-symbolic collaboration mechanism: the LLM serves as a planner responsible for high-level semantic understanding and path reflection, while the symbolic solver acts as an executor responsible for formal verification and rigorous theorem application. The neural reasoning capability of the LLM and the logical completeness of the symbolic system complement each other, fundamentally eliminating the risk of hallucinations. Second, we construct the first bidirectional symbolic reasoning engine that fully unifies forward and backward solving. For each theorem, we define two reversible operations: forward application and backward decomposition. We further provide a theoretical analysis that bidirectional reasoning achieves an exponential complexity advantage over unidirectional reasoning. Together, these two mechanisms form an iterative closed loop between neural planning and symbolic execution. On the FormalGeo7K benchmark, RAVS achieves a 96.06% solving accuracy, substantially outperforming existing state-of-the-art methods, without requiring any additional problem-specific annotated data. Code and data are available at https://github.com/BitSecret/RAVS.

1 Introduction

Geometric Problem Solving has long been a fundamental challenge [26, 20] in artificial intelligence and automated reasoning. **Formal Geometric Problem Solving** requires every step of reasoning to be conducted within a strict logical framework and rigorously verified. This makes it considerably more difficult than solving geometry problems using the natural language. Specifically, the task faces two key challenges. First, geometry problems involve both image understanding and text parsing. The solving process interweaves relational reasoning, algebraic computation, and logical operations, with reasoning strategies encompassing deductive reasoning and goal decomposition. Formalizing this process in its entirety is inherently difficult. Second, the theorem search space grows exponentially with the number of reasoning steps, and the reasoning process can be represented as a large directed acyclic graph with multiple potential solution paths, further compounding the search complexity.

The research community has pursued this problem for decades, progressing through several stages: search-based methods [19, 31], computation-based methods [44, 3], geometric expert systems [11, 62], and approaches based on machine learning [38, 37]. Before the rise of deep learning, geometric reasoning primarily relied on manually designed rules and search heuristics, lacking the ability to

*Corresponding: tleng@shu.edu.cn

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

acquire knowledge automatically. In recent years, two mainstream technical paradigms have emerged for geometric problem solving. The first paradigm [57, 59, 17, 13] is natural language reasoning driven by general-purpose large language models (LLMs), which directly generates natural language solution steps. The second paradigm [40, 29, 60, 21, 56] combines specialized theorem prediction neural networks with formal symbolic solvers: the neural network predicts the theorem required for the solution, while the symbolic solver performs precise reasoning and computation. These two components interact to achieve efficient heuristic search.

However, the first paradigm suffers from a fundamental problem: natural language proofs generated directly by LLMs cannot guarantee correctness. Even with enhanced frameworks such as CoT [42] or ReAct [52], the generated steps may violate the premise of geometric theorems. The absence of a formal verification mechanism makes such approaches difficult to deploy in scenarios requiring high reliability. To fundamentally address this issue, this paper proposes **Reflect Agent Verified Solve (RAVS)**. The design of RAVS follows a core principle: let the LLM do what it excels at (high-level planning and abstract reasoning), and let the symbolic system do what it excels at (rigorous verification and precise computation). During the solving process, the LLM analyzes the current state, reflects on previously explored paths, and formulates the next plan, while the actual theorem application and algebraic computation are performed by a formal symbolic solver. Every operation in the system must pass strict verification by the solver and receive feedback, thereby eliminating hallucinations at the architectural level. In particular, RAVS introduces a memory update mechanism: when the context reaches a certain size, the LLM automatically summarizes the progress, identifies failed exploration paths, and dynamically updates its solving strategy accordingly. This closed loop of "reasoning, execution, reflection, and memory" enables the system to maintain an efficient reasoning direction on complex problems.

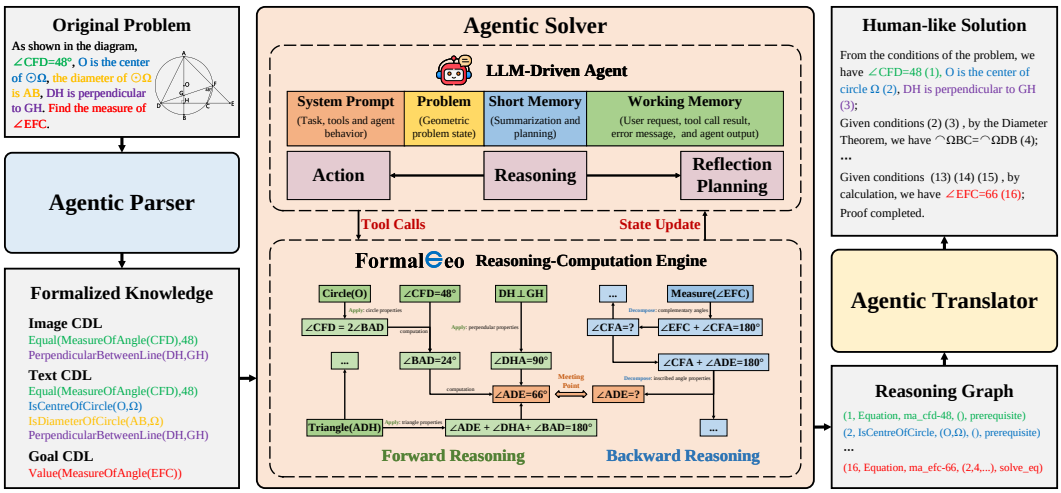

Figure 1: Overall Architecture of **Reflect Agent Verified Solve (RAVS)**. Geometric problems are parsed into formal language by an Agentic Parser, then input into an Agentic Solver for solving. After the problem is solved, the reasoning graph is transformed into a human-readable solution by an Agentic Translator. The Agentic Solver is the core method introduced in this paper. It consists of an LLM-driven agent (high-level planning and abstract reasoning) and a formal bidirectional solver based on FormalGeo (rigorous verification and precise computation).

On the other hand, while the second paradigm guarantees the correctness of reasoning, its efficiency is limited by inherent shortcomings of existing formal systems [11, 27, 5, 61]. Humans employ two modes of thinking when solving geometry problems: forward reasoning (deriving conclusions from given conditions) and backward reasoning (decomposing the goal into subgoals). Backward reasoning is often more efficient as the goal provides rich heuristic information. However, existing systems all adopt forward reasoning and have never considered the implementation of reversible operations at the formal theorem level. Given a reasoning depth d and a theorem library of size n, the worst-case complexity of search is $O(n^d)$. This exponential explosion renders such systems inadequate for problems with long reasoning chains. The bidirectional reasoning engine proposed in this paper fundamentally changes this situation. During reasoning, the system simultaneously

searches in both forward and backward direction, and terminates when the two search frontiers meet at some intermediate state. Theoretical analysis also shows that bidirectional reasoning is superior to any unidirectional approach. Furthermore, the entire solving process is organized into a clear dual-graph structure: the forward constructs a hypergraph of facts and theorems, while the backward constructs a composite graph of goals, subgoals, and theorems. This structure not only makes the reasoning process transparent but also can be naturally converted into human-readable solution steps.

In summary, this paper proposes the third technical paradigm for geometric problem solving in the deep learning era: a deep integration of large language models, agent architectures, and formal symbolic solvers. The overall framework is illustrated in Figure 1. In this framework, the LLM does not directly participate in symbolic-level reasoning; instead, it acts as a high-level planner that provides strategic guidance to the formal symbolic solver. The symbolic solver is responsible for executing rigorous formal reasoning and computation, ensuring complete correctness of the entire solving process at the architectural level. Compared with the two existing paradigms, our framework can simultaneously leverage the general knowledge and strong planning capabilities of LLMs while achieving fully reliable reasoning through the correctness verification provided by the symbolic system. Extensive experiments validate the effectiveness of the proposed method. RAVS achieves a 95.80% accuracy rate on the FormalGeo7K [63] benchmark, significantly surpassing existing state-of-the-art methods. We also provide three examples to demonstrate the advantages of RAVS, as shown in Figure 2.

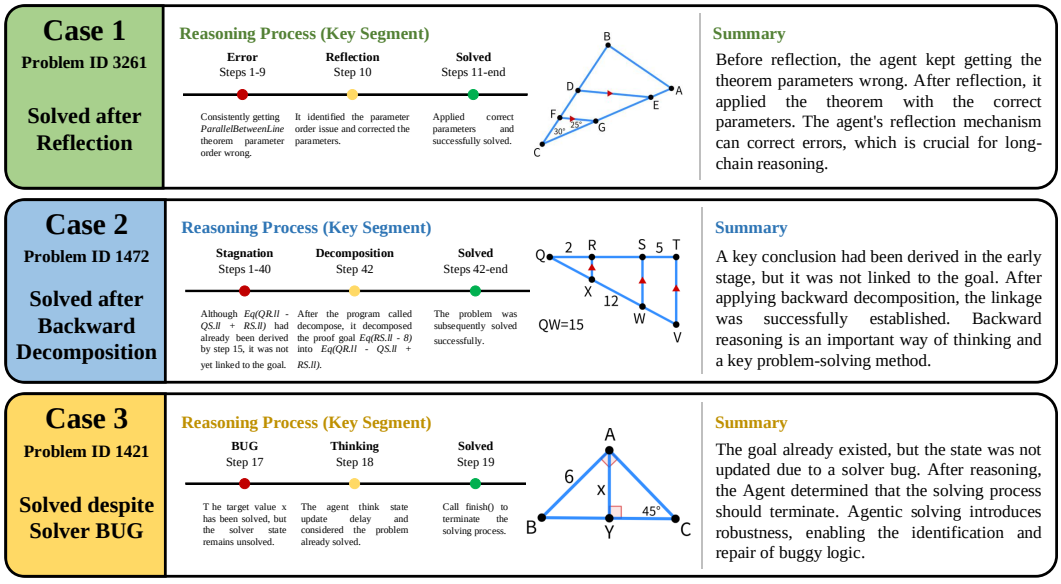

Figure 2: Three examples illustrating the advantages of RAVS. **Case 1** illustrates that the agent's reflection mechanism helps correct errors in long-chain reasoning. **Case 2** illustrates that backward solving is an important solving method. **Case 3** illustrates that the RAVS architecture can correct bugs in the solver and offers greater flexibility.

The core innovations and contributions of this paper are summarized as follows:

1. We propose the RAVS framework, a new technical paradigm that deeply integrates large language models, agent architectures, and formal symbolic solvers. By adopting a decoupled design where the LLM acts as a high-level planner and the symbolic solver performs rigorous verification, this approach achieves verifiable, hallucination-free formal geometric problem solving.
2. We introduce a bidirectional formal symbolic reasoning mechanism, which defines two reversible operations: forward reasoning (deriving conclusions from given conditions) and backward reasoning (decomposing the goal into subgoals). We theoretically analyze that bidirectional reasoning offers an exponential complexity advantage over unidirectional reasoning.
3. We conduct experiments to validate the effectiveness of the proposed method. On the FormalGeo7K [63] benchmark, our method achieves a 95.80% accuracy rate, significantly surpassing existing state-of-the-art methods Pri-TPG [65] 89.29%, without relying on any additional labeled data for specific problems.

2 Related work

Geometry is a discipline that studies numbers and shapes. Based on the different fundamental elements they process, we classify geometric problem solving methods into three categories: algebraic methods, hybrid methods, and geometric methods.

Algebraic methods [45, 3, 50] transform geometric problems into algebraic form, enabling solution via algebraic techniques. Wu's method [44] was a foundational contribution, followed by Buchberger's Gröbner bases [3], numerical parallel methods [49], polynomial triangulation elimination [18], cylindrical algebraic decomposition [12], dimensionality reduction [50]. While these methods leverage computational power, their solving processes are not easily interpretable to humans.

Hybrid methods [58, 9] bridge algebra and geometry to produce more interpretable proofs than purely algebraic approaches, though their problem coverage is limited. Point elimination method [58] uses constructive descriptions based on geometric invariants, generating concise readable proofs for many non-trivial problems. This line of work advanced rapidly [7, 9, 23], producing tools such as Geometry Explorer [43], Geometry Expert [10], and Java Geometry Expert [53], with extensions to solid [8] and non-Euclidean [51] geometry. However, the method demands substantial effort in discovering new invariants and shape constructions, limiting the range of solvable problems.

Geometric methods have been the most extensively studied, progressing through three notable stages: early symbolic reasoning [19, 31], geometric expert systems [11, 62], and machine learning-based approaches [38, 1, 37, 54]. More recently, **deep learning-based methods** have advanced the field along two routes. The first uses general-purpose LLMs for natural language reasoning [57, 16, 28, 55, 59, 25, 17, 64, 13]. This avoids building a formal system but cannot verify output correctness. The second combines specialized theorem prediction networks with formal symbolic solvers [40, 29, 60, 21, 56], using formal language (or explainable operations [48, 46, 47, 41, 32], or Python code [39]) for reasoning. This builds more reliable systems but the cost of human annotation or synthetic formal data is high.

3 Preliminaries

3.1 Problem Definition

Definition 1 (Geometric Problem). A geometric problem is defined as a tuple $Q = (F, G, E)$. Here, $F = \{f_1, f_2, \ldots, f_m\}$ is a set of known facts, $G = \{g_1, g_2, \ldots, g_n\}$ is a set of goals to be proved, where the first goal g_1 is the original solving target and the remaining goals are those derived through decomposition. Each goal is a geometry predicate logic expression tree composed of subgoals s_i and logical connectives (AND, OR, and NOT). Each fact and each subgoal is represented by a geometric predicate. $E = \{E^{(f)}, E^{(g)}, E^{(s)}\}$ denotes the set of edges connecting facts, goals, and subgoals.

Definition 2 (Theorem). A theorem is defined as a tuple $\tau = (P, C)$, where P is the premise, expressed as a geometric predicate logic expression composed of geometric predicates and logical connectives, and C is the conclusion, consisting of a single geometric predicate. The theorem library $T = \{\tau_1, \tau_2, \ldots, \tau_k\}$ is the set of all available theorems.

3.2 Task Definition

Let $Q_i = (F_i, G_i, E_i)$ denote the state of a geometric problem at time step i. The task of geometric solving is to find an ordered sequence of theorems $S = [\tau_1, \tau_2, \ldots, \tau_t]$ from the theorem library T, along with a corresponding sequence of application modes $U = [u_1, u_2, \ldots, u_t]$, such that applying these theorems in order proves the initial goal g_1, i.e., $g_1 \in F_t$. Each theorem $\tau_j = (P_j, C_j)$ can be applied in two modes:

Definition 3 (Forward Solving). If the geometric predicate logic expression P_j evaluates to true given the current fact set F_i, then C_j is added to F_i: $F_{i+1} \leftarrow F_i \cup \{C_j\}$. The theorem τ_j is also added to the edge set: $E_{i+1} \leftarrow E_i \cup \{\tau_j\}$.

Definition 4 (Backward Solving). If C_j is a subgoal of some current goal $g \in G_i$, then P_j is added to the goal set G_i: $G_{i+1} \leftarrow G_i \cup \{P_j\}$. The theorem τ_j is also added to the edge set: $E_{i+1} \leftarrow E_i \cup \{\tau_j\}$.

3.3 Interaction Process

The previous subsections describe how a geometric problem is represented and solved within the solver. We now formally define the interaction process between the solver and the agent. We model this interactive solving process as a quadruple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \pi)$.

State Space $\mathcal{S}$. The state at step t is defined as $s_t = (Q_t, M_t)$, where Q_t is the current problem state and M_t is the agent's memory.

Action Space $\mathcal{A}$. An action $a_t \in \mathcal{A}$ consists of selecting a theorem $\tau \in T$ and specifying an application mode $u \in \{\text{forward}, \text{backward}\}$. An action is valid only if the corresponding operation is executable in the current state.

State Transition Function $\mathcal{P}$. The transition $s_{t+1} = \mathcal{P}(s_t, a_t)$ is deterministic. After executing action a_t, the solver updates the state according to the predefined semantics of the theorem, producing a unique successor state.

Policy π. The policy $\pi(a_t \mid s_t; w, M_t)$ is parameterized by the LLM weights w and the memory module M_t, where M_t stores the history of reasoning and reflection.

The agent selects a theorem from the action space $\mathcal{A}$ according to the policy π. The solver applies the theorem, updates the problem state via the state transition function $\mathcal{P}$, and returns feedback to the agent. The agent incorporates this feedback into its memory, updates the state space $\mathcal{S}$, and proceeds to the next iteration until the problem is solved.

4 Method

4.1 Bidirectional Reasoning

We implement a formal bidirectional symbolic solver for geometric problems using Python, built upon the FormalGeo formal system. The solver uniformly represents the relational reasoning, algebraic computation, and logical operations involved in geometric problem solving as geometric predicate logic operations, and unifies forward and backward solving within a single formal framework.

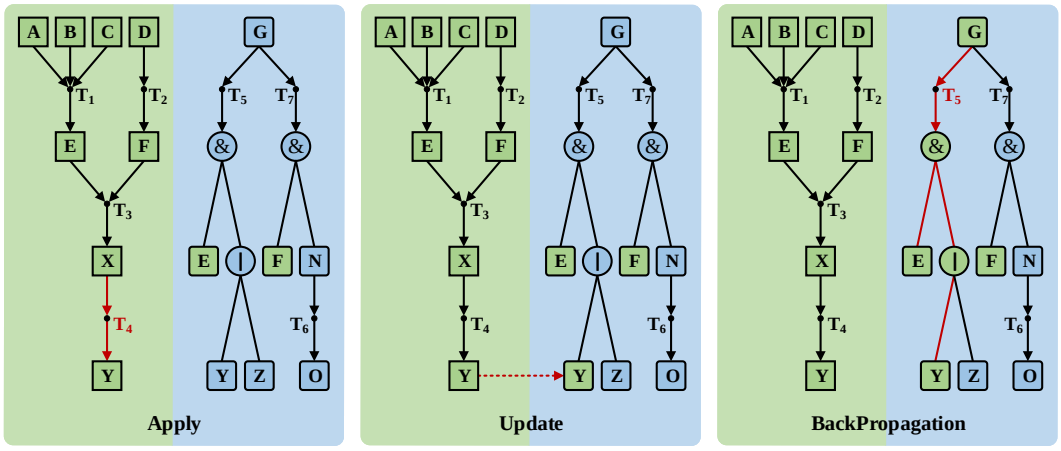

Figure 3: Reasoning graph of the bidirectional solving process. The green graph represents the forward reasoning graph, while the blue graph represents the backward reasoning graph. **Step 1: Apply.** The forward solver selects an applicable theorem from the current fact set, derives new facts, and updates the fact set. **Step 2: Update.** Whenever a new fact is produced, the system immediately queries a hash table that records all pending subgoals in the backward process. If a matching subgoal exists, its state is updated to `solved`. Since the hash table lookup has a time complexity of $O(1)$, the overhead of meeting detection is negligible. **Step 3: BackPropagation** After a subgoal state is updated, the system propagates the change upward according to logical operation rules. If all subgoals of a goal are marked as `solved`, the goal's state is updated to `solved`, and the theorem used to decompose the goal is fed back to the forward process, triggering a new round of interaction.

Humans typically employ two modes of thinking when solving geometric problems: forward reasoning (deriving conclusions from given conditions) and backward reasoning (decomposing the goal into subgoals). In certain scenarios, backward reasoning is more efficient because the goal itself provides rich heuristic information. Although the idea of backward reasoning emerged early [31] in the field of geometric reasoning, it has received far less attention than forward reasoning. Recent systems [62] that implement backward reasoning still treat forward and backward reasoning as independent modules rather than integrating them into a unified framework. Compared with forward reasoning, backward reasoning is more difficult to implement due to its complex state updates. Building a deeply integrated system that unifies both directions poses an even greater challenge. The core difficulties lie in two aspects: how to describe both reasoning modes in a unified manner, and how to accurately detect when forward and backward meet.

This paper presents the first fully unified forward and backward reasoning system for geometric problem solving. The mechanisms for meeting detection, interactive feedback, and state updates in backward solving are illustrated in Figure 3. The iterative interaction process of bidirectional reasoning is fully automatic and requires no human intervention. Through this unified bidirectional reasoning framework, the system simultaneously leverages the completeness of forward reasoning and the heuristics of backward reasoning, significantly improving search efficiency while maintaining correctness.

We theoretically show that bidirectional reasoning outperforms any unidirectional approach. Viewed as heuristic search with an LLM agent providing guidance, let D denote problem difficulty, i.e., the length of the theorem sequence needed for solving. The search tree grows exponentially with D, so larger D means harder problems.

For a problem of difficulty D, consider forward, backward, and bidirectional search. In forward search, each node expands to M children, with time a per expansion. The time to reach depth d is

$$T_{forward} = a \sum_{i=1}^{d} M^i = a(M^1 + M^2 + \cdots + M^d)$$

In backward search, each node expands to N children, with time b per expansion. From depth D toward depth 0, the time to cover the remaining distance $D - d$ is

$$T_{backward} = b \sum_{j=1}^{D-d} N^j = b(N^1 + N^2 + \cdots + N^{D-d})$$

Here M, N, a, and b depend on the theorem library: more theorems yield larger values. D reflects problem complexity, while M, N, a, and b reflect formal system complexity.

Bidirectional search outperforms unidirectional search under the following assumptions: expanding the last forward (or backward) layer takes more time than expanding the first backward (or forward) layer:

$$a \cdot M^D > b \cdot N \quad and \quad b \cdot N^D > a \cdot M$$

Then there exists a depth d where forward search is more efficient before d, and backward search after d. This switching point satisfies

$$M^d \leq N^{D-d} \quad and \quad M^{d+1} \geq N^{D-d-1}$$

Furthermore, traditional backward reasoning operates at the granularity of predicates (e.g., decomposing the goal `RightTriangle(A,B,C)`), while forward reasoning operates at the granularity of theorems (e.g., applying the theorem `pythagorean_inverse` to determine whether a `Triangle(A,B,C)` is right-angled). This mismatch in granularity makes it difficult to unify forward and backward reasoning. To address this, we explicitly require that the conclusion of each theorem consists of a single predicate. Then each theorem is associated with a specific conclusion (goal). Consequently, we can describe forward and backward reasoning in a unified manner: both apply a theorem to either expand facts or decompose goals.

4.2 Agentic Loop

Existing LLM-based geometric solving methods typically require the LLM to directly generate complete proof steps, including theorem selection, algebraic computation, and the construction of

Algorithm 1 RAVS Solving Loop

Require: Geometric Problem $Q = (F, G, E)$, Theorem Library T, Maximum Epochs $E_{\max}$
Ensure: Solving Result $status \in \{\text{solved}, \text{failed}\}$, Solving Trajectory M_t

Initialize symbolic solver SOLVER with theorem library T and problem Q
Initialize LLM Agent AGENT with system prompt and solver state SOLVER.state()
$e \leftarrow 0$
while $e < E_{\max}$ **do**
    $e \leftarrow e + 1$
    $response \leftarrow$ AGENT.reasoning() ▷ Step 1: Reasoning
    $(tool, args) \leftarrow$ ParseTool($response$)
    if $tool \neq$ summarize **then**
        $result \leftarrow$ ExecuteTool(SOLVER, $tool, args$) ▷ Step 2: Tool Execution
    else
        AGENT.summarize($args$, SOLVER.state()) ▷ Step 3: Reflection and Planning
        continue
    end if
    AGENT.$M_t \leftarrow$ AGENT.$M_t \cup [response, result]$ ▷ Step 4: Memory Update
    if SOLVER.status() = solved **then** ▷ Step 5: Termination Check
        break
    end if
end while
return SOLVER.status(), AGENT.M_t ▷ Step 6: Result Determination

reasoning chains. However, LLMs inevitably produce logical gaps or premise violations in long-sequence reasoning—a phenomenon known as "hallucination". Even with enhanced frameworks such as CoT [42] or ReAct [52], the lack of formal verification mechanisms makes such approaches difficult to deploy in high-reliability scenarios.

This section details the solving loop of RAVS. The design of RAVS follows a core principle: let the LLM do what it excels at (high-level planning and abstract reasoning), and let the symbolic system do what it excels at (rigorous verification and precise computation). This principle is realized as a four-stage closed loop "reasoning, execution, reflection, and memory", as shown in Algorithm 1:

Stage 1: Reasoning. At each loop iteration, the LLM generates reasoning based on the memory buffer M_t, which contains the system prompt, initial problem state, and full interaction history.

Stage 2: Tool Execution and Reflection (Steps 2-3). The LLM's response is parsed into a tool call. RAVS defines two core operations: forward application (`apply`) to derive new facts, and backward decomposition (`decompose`) to split a goal into subgoals. Auxiliary tools include `find_fact`, `find_goal`, and `check`. Two special control tools are provided: `summarize` triggers reflection when a bottleneck is reached, compressing the explored trajectory into a structured summary to replace redundant history; and `finish` actively terminates the solving loop when the goal is proved or deemed unsolvable.

Stage 3: Memory Update (Step 4). Tool execution results and error messages are appended to M_t, ensuring the LLM maintains complete context. A context length threshold $L_{\max}$ triggers automatic summarization when exceeded.

Stage 4: Termination Decision (Steps 5-6). At each loop end, the system checks the solver's goal status: early termination with success if proved, timeout if $E_{\max}$ is reached, or termination via `finish` with status determined by the solver.

5 Experiment

5.1 Datasets and Evaluation

We evaluate on three formal geometry benchmarks: FormalGeo7K [63], Geometry3K [27], and GeoQA [5]. Geometry3K contains 3,002 diagram-text geometry problems with formal language parsing annotations. GeoQA provides 5,010 geometry problems with annotated programs that

describe explicit solving procedures. FormalGeo7K comprises 7,000 geometry problems and has recently received widespread attention from the geometry problem-solving community [60, 30, 6, 35, 65]. It is a large-scale formal geometry dataset with natural language descriptions, high-quality geometric diagrams, GeoGebra-formatted geometric image source files, ground-truth formal language annotations, theorem sequence and theorem directed acyclic graph annotations, designed to support various tasks such as geometry problem parsing and solving. We split the dataset into training, validation, and test sets in a 4:1:1 ratio, and report our method's performance on the test set. Following FGeo-HyperGNet [60], we stratify test problems into six difficulty levels (L1–L6): L1 ($1 \leq l \leq 2$), L2 ($3 \leq l \leq 4$), L3 ($5 \leq l \leq 6$), L4 ($7 \leq l \leq 8$), L5 ($9 \leq l \leq 10$), and L6 ($l \geq 11$), where l is the number of theorems required to solve the problem. We report accuracy (%) as the proportion of solved problems in the test set. A problem is considered solved if and only if the formal solver state is `solved`.

5.2 Benchmark Methods

We compare our method against representative baselines. **Symbolic-only solvers** employ search-based approaches to solve problems, using the relationship between the problem's known conditions as heuristic information, combined with various search strategies (e.g., depth-first search, breadth-first search) to solve problems via heuristic search. **Neural-only solvers** solve geometry problems in an end-to-end fashion using LLMs. These methods directly generate natural language solutions, and the evaluation of solutions is also performed through LLMs. **Neural-symbolic solvers** integrate neural networks with a symbolic reasoning engine to perform theorem-driven search. They can be further divided into training-based and training-free. **Training-based neural-symbolic solvers** fine-tune a pre-trained neural network or train a theorem prediction network from scratch to perform theorem selection, solving problems through the interaction between a theorem predictor and a symbolic solver. **Training-free neural-symbolic solvers** leverage pre-trained LLMs for theorem prediction and apply theorems through symbolic solvers, avoiding any task-specific training.

Table 1: Comparison results between different methods on FormalGeo7K (*accuracy*, %). Our proposed method, RAVS, achieves state-of-the-art performance, significantly outperforming every baseline method across all difficulty levels.

Methods	Total	L1	L2	L3	L4	L5	L6
Symbolic-Only Methods							
Backward-DFS [62]	33.73	67.99	32.99	9.78	5.90	5.36	0.62
Backward-RS [62]	34.05	68.85	33.51	9.86	5.22	4.76	0.62
Forward-BFS [62]	38.86	61.37	37.89	32.33	17.91	9.23	2.49
Forward-RS [62]	39.71	60.74	39.33	37.22	16.55	6.25	4.05
Neural-Only Methods							
Kimi-K2 [22]	60.75	69.05	70.29	65.77	36.00	33.33	19.15
DeepSeek v3 [14]	60.79	75.72	58.81	63.26	46.29	23.44	33.87
GPT-5 mini [34]	64.79	74.11	63.30	64.66	53.50	53.23	41.46
Qwen3-VL [36]	65.93	74.53	65.43	72.18	50.96	41.94	36.67
Doubao seed 1.8 [4]	69.14	74.11	69.15	71.43	64.33	50.00	51.67
GPT-5.2 [33]	73.14	80.38	73.40	74.81	63.06	59.68	46.67
Claude4.5 Sonnet [2]	75.79	84.55	73.94	76.32	67.52	64.52	48.33
Neural-Symbolic Methods (Training-Based)							
Inter-GPS [27]	60.50	81.07	60.62	60.98	38.86	20.31	11.29
DualGeoSolver [48]	62.11	63.64	68.10	65.94	58.00	60.98	31.71
NGS [5]	62.60	62.88	65.64	71.01	57.00	60.98	36.59
FGeo-DRL [66]	80.85	97.63	92.11	70.89	61.60	39.39	31.37
FGeo-ISRL [24]	85.16	95.12	93.30	82.26	73.24	61.90	35.20
HyperGNet [60]	88.36	96.44	92.49	91.29	78.29	57.81	51.61
Neural-Symbolic Methods (Training-Free)							
Pri-TPG [65]	89.29	99.16	96.28	87.92	77.07	66.13	30.00
Ours (RAVS)	**96.06**	**99.74**	**98.72**	**94.59**	**96.00**	**91.23**	**61.70**

Table 2: Comparison results between different methods on Geometry3K, GeoQA and FormalGeo7K (*accuracy*, %). Our method achieves the best reported accuracy on every datasets. – indicates results not reported. * indicates evaluation on the subset parsed into the FormalGeo formal language.

Method	Geometry3K	GeoQA	FormalGeo7K
E-GPS [46]	90.40	–	–
DualGeoSolver [48]	–	65.20	–
FGeo-ISRL [24]	–	–	85.16
HyperGNet* [60]	91.99	85.64	88.36
Pri-TPG* [65]	95.16	85.02	89.29
Ours (RAVS)*	**98.11**	**94.64**	**96.06**

5.3 Experiment Setup

Our method runs on a personal laptop (CPU i7-13650HX, RAM 16GB, GPU RTX 4070). We set the maximum number of interaction rounds to 50 to avoid infinite loops, and the maximum memory length to 80k to force reflection and replanning when the model does not reflect autonomously. This paper uses DeepSeek-V4-Pro [15] as the base model to test the agent capabilities. We use multi-processing to improve CPU utilization and API call efficiency. For 110 problems that were not solved in the first round of interaction, we attempt to solve them again a second time.

5.4 Experimental Results

Table 1 presents the performance various methods on the FormalGeo7K dataset [63]. Our proposed method, RAVS, achieves state-of-the-art performance with an overall accuracy of 95.80%, significantly outperforming every baseline method across all difficulty levels.

Purely symbolic methods are fundamentally limited on complex problems. Purely symbolic solvers guarantee reasoning correctness but achieve only 33%–40% overall accuracy. Forward search outperforms backward search in both overall accuracy and across all difficulty levels, yet both methods drop below 10% on L5 and L6 problems, revealing their inherent weakness in long-chain reasoning without high-level guidance.

Training-based neural-symbolic methods still struggle to generalize to hard problems. State-of-the-art training-based methods like FGeo-ISRL (85.16%) and HyperGNet (88.36%) perform well on simpler problems but show significant accuracy decline on L5–L6, with HyperGNet dropping to 51.61% on L6. Purely LLM-based methods achieve over 80% on L1 but also suffer from performance degradation on hard problems and lack formal verification, making their results unreliable.

RAVS achieves superior and consistent performance across all difficulty levels. RAVS reaches 96.06% overall accuracy, with particularly striking gains on L5 (91.23% vs. 66.13%) and L6 (61.70% vs. 51.67%). Without any training, our reflection-driven neuro-symbolic framework also outperforms the training-free LLM baseline Pri-TPG by 6.77 percentage points, demonstrating that the integration of LLM planning with symbolic verification, mediated through bidirectional reasoning and agentic reflection, delivers robust and verifiable formal reasoning across the entire difficulty spectrum. However, RAVS's performance on L6-level difficulty is limited.

Table 2 report comparison results between different methods on Geometry3K, GeoQA and FormalGeo7K (*accuracy*, %). RAVS achieves state-of-the-art results across all three benchmarks, outperforming the strongest baselines by 2.95, 9, and 6.77 percentage points respectively.

Table 3: Ablation results. w/o Bidirectional denotes unidirectional reasoning only, w/o Reflection denotes no reflection mechanism, and w/o Second-pass Retry denotes only one attempt per problem.

Methods	Total	L1	L2	L3	L4	L5	L6
w/o Bidirectional	92.80	98.68	97.12	92.79	92.00	75.44	40.43
w/o Reflection	93.83	99.21	97.76	93.24	93.33	78.95	46.81
w/o Second-pass Retry	90.57	97.62	96.49	91.44	83.33	70.18	38.30
RAVS	**96.06**	**99.74**	**98.72**	**94.59**	**96.00**	**91.23**	**61.70**

Table 3 report comparison results. It can be observed that: (1) Bidirectional reasoning contributes on hard problems: removing it costs 15.79% on L5 and 21.27% on L6, compared to only marginal drops 3.26% on L1. This confirms that bidirectional reasoning is crucial for solving difficult problems. (2) Reflection and memory contributes 12.28% on L5 and 14.89% on L6, while showing negligible impact on L1, confirming that the reflection mechanism is crucial for preventing dead-end search in long-chain reasoning. (3) Second-pass retry contributes 21.05% on L5 and 23.40% on L6, yet only 2.12% on L1, serving as an effective recovery mechanism specifically for hard problems. These results demonstrate that all three components are essential, particularly for high difficulty problems.

Statistical information such as tool calls during the interaction process is shown in Figure 4. Most failed problems hit the interaction limit, resulting in balanced resource consumption across difficulty levels. For solved problems, difficulty positively correlates with resource usage. From L1 to L6, interaction count, context length, and tool calls all increase, indicating that complex problems require longer reasoning chains, richer context, and more frequent tool assistance.

Limitations. Our approach has three main limitations. First, it depends on the reasoning quality of the backbone LLM, which constitutes the primary computational cost of approximately 55 USD per full evaluation run. This limitation will diminish as LLMs continue to improve. Second, the system prompt must carefully specify the formal system and task loop, occupying substantial context windows. This challenge could be mitigated through progressive disclosure or dynamic prompt construction. Third, performance on L6 level problems with the longest reasoning chains remains limited, as longer chains demand more interaction rounds and accumulate longer contexts. Despite memory compression via reflection, context growth persists, suggesting future directions in optimized prompting and refined agent loop design.

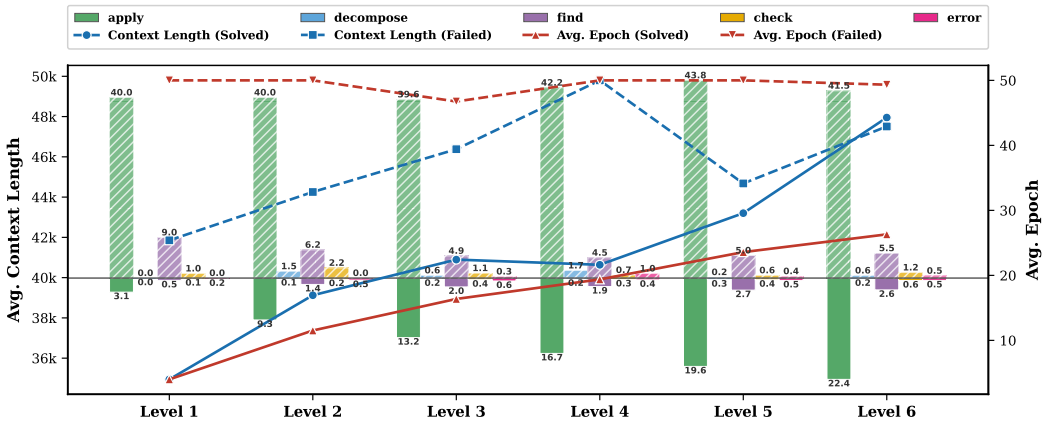

Figure 4: Statistical analysis of the interaction process. The bar chart represents the average number of tool calls, while the line charts represent the average context length and the average number of interactions. Dashed lines represent failed problems, while solid lines represent solved problems. For solved problems, difficulty positively correlates with resource usage.

6 Conclusion

We present RAVS, a unified neuro-symbolic framework that achieves 96.06% solving accuracy on FormalGeo7K by decoupling LLM planning from symbolic execution. A bidirectional reasoning engine with proven exponential efficiency advantages unifies forward and backward solving, while an agentic loop enables adaptive reflection. RAVS surpasses prior methods without any task-specific annotated data, demonstrating the power of principled neural-symbolic integration.

Acknowledgments and Disclosure of Funding

This research was supported by National Natural Science Foundation of China (NSFC) grant 12071282. The authors declare no competing interests. We thank the Xiaomi MiMo Orbit 100 Trillion Token Creator Incentive Program for providing 200 million tokens for this research.

References

[1] Alvin, C., Gulwani, S., Majumdar, R., Mukhopadhyay, S.: Synthesis of solutions for shaded area geometry problems. In: The Thirtieth International Flairs Conference (2017)

[2] Anthropic: Claude sonnet 4.5. https://www.anthropic.com/news/claude-sonnet-4-5 (2025)

[3] Buchberger, B.: Applications of gröbner bases in non-linear computational geometry. Mathematical aspects of scientific software pp. 59–87 (1988)

[4] Bytedance Seed: Seed1.8 model card: Towards generalized real-world agency (2026), https://arxiv.org/abs/2603.20633

[5] Chen, J., Tang, J., Qin, J., Liang, X., Liu, L., Xing, E., Lin, L.: GeoQA: A geometric question answering benchmark towards multimodal numerical reasoning. In: Findings of the Association for Computational Linguistics (ACL-IJCNLP). pp. 513–523. Association for Computational Linguistics, Online (8 2021). https://doi.org/10.18653/v1/2021.findings-acl.46, https://aclanthology.org/2021.findings-acl.46/

[6] Cheng, J.K., Zhang, Z., Chen, R., Deng, J., Qin, Z., Ma, J.: Geouni: A unified model for generating geometry diagrams, problems and problem solutions. In: Proceedings of the 33rd ACM International Conference on Multimedia (ACM MM). p. 3057–3066. MM '25, Association for Computing Machinery, New York, NY, USA (2025). https://doi.org/10.1145/3746027.3754965, https://doi.org/10.1145/3746027.3754965

[7] Chou, S.C., Gao, X.S., Zhang, J.Z.: Automated geometry theorem proving by vector calculation. In: Proceedings of the 1993 international symposium on Symbolic and algebraic computation. pp. 284–291 (1993)

[8] Chou, S.C., Gao, X.S., Zhang, J.Z.: Automated production of traditional proofs in solid geometry. Journal of Automated Reasoning **14**(2), 257–291 (1995)

[9] Chou, S.C., Gao, X.S., Zhang, J.Z.: Automated generation of readable proofs with geometric invariants. journal of Automated Reasoning **17**(3), 349–370 (12 1996). https://doi.org/10.1007/BF00283134, https://doi.org/10.1007/BF00283134

[10] Chou, S.C., Gao, X.S., Zhang, J.Z.: An introduction to geometry expert. In: CADE. vol. 1104, pp. 235–239 (1996)

[11] Chou, S.C., Gao, X.S., Zhang, J.Z.: A deductive database approach to automated geometry theorem proving and discovering. journal of Automated Reasoning **25**(3), 219–246 (10 2000). https://doi.org/10.1023/A:1006171315513, https://link.springer.com/article/10.1023/A:1006171315513

[12] Collins, G.E.: Quantifier elimination for real closed fields by cylindrical algebraic decomposition–preliminary report. ACM SIGSAM Bulletin **8**(3), 80–90 (1974)

[13] DeepMind: Gemini achieves gold-medal level at the international collegiate programming contest world finals (2025), https://deepmind.google/blog/gemini-achieves-gold-medal-level-at-the-international-collegiate-programming-contest-world-
accessed: 2025-12-18

[14] DeepSeek-AI: Deepseek-v3 technical report (2024), https://arxiv.org/abs/2412.19437

[15] DeepSeek-AI: Deepseek-v4: Towards highly efficient million-token context intelligence. Technical report, DeepSeek (Apr 2026), https://huggingface.co/deepseek-ai/DeepSeek-V4-Pro/blob/main/DeepSeek_V4.pdf, accessed on May 6, 2026

[16] Duan, X., Tan, D., Fang, L., Zhou, Y., He, C., Chen, Z., Wu, L., Chen, G., Gong, Z., Luo, W., Guan, Q.: Reason-and-execute prompting: Enhancing multi-modal large language models for solving geometry questions. In: Proceedings of the 32nd ACM International Conference on Multimedia (ACM MM). p. 6959–6968. MM '24, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3664647.3681484, https://doi.org/10.1145/3664647.3681484

[17] Gao, J., Pi, R., Zhang, J., Ye, J., Zhong, W., Wang, Y., Hong, L., Han, J., Xu, H., Li, Z., Kong, L.: G-llava: Solving geometric problem with multi-modal large language model. In: Proceedings of the Thirteenth International Conference on Learning Representations (ICLR) (may 2025), https://openreview.net/forum?id=px1674Wp3C

[18] Gao, X.S., Chou, S.C.: On the dimension of an arbitrary ascending chain. CHINESE SCIENCE BULLETIN-ENGLISH EDITION- **38**, 799–799 (1993)

[19] Gelernter, H.L.: Realization of a geometry theorem proving machine. In: IFIP congress. pp. 273–281 (1959)

[20] Gowers, T., Loh, P.S., Roberts, D., Smith, G., Tao, T., Sculley, D., Buzzard, K., de Moura, L., Mackey, L., Liu, P.J.: Artificial intelligence mathematical olympiad prize (aimo prize) (2023), https://aimoprize.com/

[21] Hubert, T., Mehta, R., Sartran, L., Horváth, M.Z., Žužić, G., Wieser, E., Huang, A., Schrittwieser, J., Schroecker, Y., Masoom, H., Bertolli, O., Zahavy, T., Mandhane, A., Yung, J., Beloshapka, I., Ibarz, B., Veeriah, V., Yu, L., Nash, O., Lezeau, P., Mercuri, S., Sönne, C., Mehta, B., Davies, A., Zheng, D., Pedregosa, F., Li, Y., von Glehn, I., Rowland, M., Albanie, S., Velingker, A., Schmitt, S., Lockhart, E., Hughes, E., Michalewski, H., Sonnerat, N., Hassabis, D., Kohli, P., Silver, D.: Olympiad-level formal mathematical reasoning with reinforcement learning. Nature (11 2025). https://doi.org/10.1038/s41586-025-09833-y, https://doi.org/10.1038/s41586-025-09833-y, online publication date: November 12, 2025

[22] Kimi Team: Kimi k2: Open agentic intelligence. arXiv preprint arXiv:2507.20534 (2025)

[23] Li, H.: Symbolic computation in the homogeneous geometric model with clifford algebra. In: Proceedings of the 2004 international symposium on Symbolic and algebraic computation. pp. 221–228 (2004)

[24] Li, Y., Zhang, X., Qin, C., Hu, Z., Leng, T.: Fgeo-isrl: A mcts-enhanced deep reinforcement learning system for plane geometry problem-solving via inverse search. Symmetry **18**(4) (2026). https://doi.org/10.3390/sym18040628, https://www.mdpi.com/2073-8994/18/4/628

[25] Linger, D., Zhu, L., Liu, Y., Wang, Y., Xie, Q., Wu, J., Zhang, G., Zhu, Y., Bai, X.: Theorem-validated reverse chain-of-thought problem generation for geometric reasoning. In: Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 718–735. Association for Computational Linguistics, Suzhou, China (11 2025). https://doi.org/10.18653/v1/2025.emnlp-main.38, https://aclanthology.org/2025.emnlp-main.38/

[26] Littman, M.L., Ajunwa, I., Berger, G., Boutilier, C., Currie, M., Doshi-Velez, F., Hadfield, G., Horowitz, M.C., Isbell, C., Kitano, H., et al.: Gathering strength, gathering storms: The one hundred year study on artificial intelligence (ai100) 2021 study panel report. arXiv preprint arXiv:2210.15767 (2022)

[27] Lu, P., Gong, R., Jiang, S., Qiu, L., Huang, S., Liang, X., Zhu, S.C.: Inter-GPS: Interpretable geometry problem solving with formal language and symbolic reasoning. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP). pp. 6774–6786. Association for Computational Linguistics, Online (8 2021). https://doi.org/10.18653/v1/2021.acl-long.528, https://aclanthology.org/2021.acl-long.528/

[28] Mouselinos, S., Michalewski, H., Malinowski, M.: Beyond lines and circles: Unveiling the geometric reasoning gap in large language models. In: Findings of the Association for Computational Linguistics (EMNLP). pp. 6192–6222. Association for Computational Linguistics, Miami, Florida, USA (11 2024). https://doi.org/10.18653/v1/2024.findings-emnlp.360, https://aclanthology.org/2024.findings-emnlp.360/

[29] Murphy, L., Yang, K., Sun, J., Li, Z., Anandkumar, A., Si, X.: Autoformalizing Euclidean geometry. In: Proceedings of the 41st International Conference on Machine Learning (ICML). Proceedings of Machine Learning Research, vol. 235, pp. 36847–36893. PMLR (21–27 Jul 2024), https://proceedings.mlr.press/v235/murphy24a.html

[30] Namgoong, H., Jung, J., Han, Y., Jung, S.: When does auxiliary modality matter in solving geometric problems? a comprehensive study of textual, formal, and visual modalities. In: Demberg, V., Inui, K., Marquez, L. (eds.) Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers). pp. 76–92. Association for Computational Linguistics, Rabat, Morocco (Mar 2026). https://doi.org/10.18653/v1/2026.eacl-short.4, https://aclanthology.org/2026.eacl-short.4/

[31] Nevins, A.J.: Plane geometry theorem proving using forward chaining. Artificial Intelligence **6**(1), 1–23 (1975)

[32] Ning, M., Zhou, Z., Wang, Q., Huang, X., Huang, K.: Gns: Solving plane geometry problems by neural-symbolic reasoning with multi-modal llms. Proceedings of the AAAI Conference on Artificial Intelligence (AAAI) **39**(23), 24957–24965 (apr 2025). https://doi.org/10.1609/aaai.v39i23.34679, https://ojs.aaai.org/index.php/AAAI/article/view/34679

[33] OpenAI: Introducing gpt-5.2. https://openai.com/index/introducing-gpt-5-2/ (2025)

[34] OpenAI: Openai gpt-5 system card (2026), https://arxiv.org/abs/2601.03267

[35] Pan, Y., Zhang, Z., Hu, P., Ma, J., Du, J., Zhang, J., Liu, Q., Gao, J., Ma, F.: Enhancing the geometric problem-solving ability of multimodal llms via symbolic-neural integration. In: Proceedings of the 33rd ACM International Conference on Multimedia (ACM MM). p. 5394–5403. MM '25, Association for Computing Machinery, New York, NY, USA (2025). https://doi.org/10.1145/3746027.3754571, https://doi.org/10.1145/3746027.3754571

[36] Qwen Team: Qwen3-vl technical report. arXiv preprint arXiv:2511.21631 (2025)

[37] Sachan, M., Xing, E.: Learning to solve geometry problems from natural language demonstrations in textbooks. In: Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (SEM 2017). pp. 251–261 (2017)

[38] Seo, M., Hajishirzi, H., Farhadi, A., Etzioni, O., Malcolm, C.: Solving geometry problems: Combining text and diagram interpretation. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1466–1476. Association for Computational Linguistics, Lisbon, Portugal (sep 2015). https://doi.org/10.18653/v1/D15-1171, https://aclanthology.org/D15-1171/

[39] Sharma, A., Dalmia, A., Kazemi, M., Zouaq, A., Pal, C.: GeoCoder: Solving geometry problems by generating modular code through vision-language models. In: Findings of the Association for Computational Linguistics (NAACL). pp. 7340–7356. Association for Computational Linguistics, Albuquerque, New Mexico (4 2025). https://doi.org/10.18653/v1/2025.findings-naacl.410, https://aclanthology.org/2025.findings-naacl.410/

[40] Trinh, T.H., Wu, Y., Le, Q.V., He, H., Luong, T.: Solving olympiad geometry without human demonstrations. Nature **625**(7995), 476–482 (1 2024). https://doi.org/10.1038/s41586-023-06747-5, https://www.nature.com/articles/s41586-023-06747-5

[41] Wang, Y., Wei, B., Ma, Y., Jiang, X., Ding, H., Cai, Z., Liu, J.: Reasoning step by step via a neural-symbolic geometry problem solver. Pattern Recognition **171**, 112261 (2026). https://doi.org/https://doi.org/10.1016/j.patcog.2025.112261, https://www.sciencedirect.com/science/article/pii/S0031320325009227

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

[43] Wilson, S., Fleuriot, J.D.: Geometry explorer: A tool for generating diagrammatic full-angle method proofs. In: Automated deduction in geometry: extended abstracts. pp. 144–150 (2006)

[44] Wu, W.T.: On the decision problem and the mechanization of theorem proving in elementary geometry. Scientia Sinica **21**, 157–179 (1978)

[45] Wu, W.T.: Basic principles of mechanical theorem proving in elementary geometries. journal of Automated Reasoning **2**(3), 221–252 (9 1986). https://doi.org/10.1007/BF02328447, https://doi.org/10.1007/BF02328447

[46] Wu, W., Zhang, L., Liu, J., Tang, X., Wang, Y., Wang, S., Wang, Q.: E-gps: Explainable geometry problem solving via top-down solver and bottom-up generator. In: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 13828–13837 (2024). https://doi.org/10.1109/CVPR52733.2024.01312

[47] Wu, W., Zhang, L., Zhao, B., Huang, M., Wang, Q., Liu, J.: Causal-r: A causal-reasoning geometry problem solver for optimized solution exploration. In: Advances in Neural Information Processing Systems (NeurIPS) (dec 2025), https://neurips.cc/virtual/2025/loc/san-diego/poster/116920

[48] Xiao, T., Liu, J., Huang, Z., Wu, J., Sha, J., Wang, S., Chen, E.: Learning to solve geometry problems via simulating human dual-reasoning process. In: Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI). pp. 6559–6568. International Joint Conferences on Artificial Intelligence Organization (8 2024). https://doi.org/10.24963/ijcai.2024/725, https://doi.org/10.24963/ijcai.2024/725, main Track

[49] Yang, L., Zhang, J., Li, C.: A prover for parallel numerical verification of a class of constructive geometry theorems. In: Proc. IWMM. vol. 92, pp. 244–250 (1992)

[50] Yang, L.: Practical automated reasoning on inequalities: Generic programs for inequality proving and discovering. In: Proceedings of the Third Asian Technology Conference in Mathematics. pp. 24–28 (1998)

[51] Yang, L., Gao, X.S., Chou, S.C., Zhang, J.Z.: Automated production of readable proofs for theorems in non-euclidean geometries. In: Automated Deduction in Geometry: International Workshop on Automated Deduction in Geometry Toulouse, France, September 27–29, 1996 Selected Papers 1. pp. 171–188. Springer (1997)

[52] Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K.R., Cao, Y.: React: Synergizing reasoning and acting in language models. In: The Eleventh International Conference on Learning Representations (2023), https://openreview.net/forum?id=WE_vluYUL-X

[53] Ye, Z., Chou, S.C., Gao, X.S.: An introduction to java geometry expert. In: Automated Deduction in Geometry. pp. 189–195. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)

[54] Yu, X., Gan, W., Wang, M.: Understanding explicit arithmetic word problems and explicit plane geometry problems using syntax-semantics models. In: 2017 International Conference on Asian Language Processing (IALP). pp. 247–251. IEEE (2017)

[55] Zeng, X., Liu, S.: Research on the application of knowledge mapping and knowledge structure construction based on adaptive learning model. Expert Systems with Applications **249**, 123400 (2024). https://doi.org/https://doi.org/10.1016/j.eswa.2024.123400, https://www.sciencedirect.com/science/article/pii/S0957417424002653

[56] Zhang, C., Song, J., Li, S., Liang, Y., Ma, Y., Wang, W., Zhu, Y., Zhu, S.C.: Proposing and solving olympiad geometry with guided tree search. Nature Machine Intelligence **8**(1), 84–95 (1 2026). https://doi.org/10.1038/s42256-025-01164-x, https://doi.org/10.1038/s42256-025-01164-x

[57] Zhang, J., Moshfeghi, Y.: GOLD: Geometry problem solver with natural language description. In: Findings of the Association for Computational Linguistics (NAACL). pp. 263–278. Association for Computational Linguistics, Mexico City, Mexico (6 2024). https://doi.org/10.18653/v1/2024.findings-naacl.19, https://aclanthology.org/2024.findings-naacl.19/

[58] Zhang, J.Z., Chou, S.C., Gao, X.S.: Automated production of traditional proofs for theorems in Euclidean geometry I. The Hilbert intersection point theorems. Annals of Mathematics and Artificial Intelligence **13**(1), 109–137 (3 1995). https://doi.org/10.1007/BF01531326, https://link.springer.com/article/10.1007/BF01531326

[59] Zhang, R., Jiang, D., Zhang, Y., Lin, H., Guo, Z., Qiu, P., Zhou, A., Lu, P., Chang, K.W., Qiao, Y., Gao, P., Li, H.: Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In: Computer Vision – ECCV 2024. pp. 169–186. Springer Nature Switzerland, Cham (2025)

[60] Zhang, X., Li, Y., Zhu, N., Qin, C., Zeng, Z., Leng, T.: Fgeo-hypergnet: Geometric problem solving integrating formalgeo symbolic system and hypergraph neural network. In: Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence (IJCAI). pp. 4733–4741. International Joint Conferences on Artificial Intelligence Organization (8 2025). https://doi.org/10.24963/ijcai.2025/527, https://doi.org/10.24963/ijcai.2025/527, main Track

[61] Zhang, X., Zhu, N., He, Y., Zou, J., Huang, Q., Jin, X., Guo, Y., Mao, C., Zhu, Z., Yue, D., et al.: Formalgeo: The first step toward human-like imo-level geometric automated reasoning. arXiv preprint arXiv:2310.18021 (2023)

[62] Zhang, X., Zhu, N., He, Y., Zou, J., Qin, C., Li, Y., Leng, T.: Fgeo-sss: A search-based symbolic solver for human-like automated geometric reasoning. Symmetry **16**(4) (2024). https://doi.org/10.3390/sym16040404, https://www.mdpi.com/2073-8994/16/4/404

[63] Zhang, X., Zhu, N., Qin, C., Yang, L., Zeng, Z., Leng, T.: Formal representation and solution of plane geometric problems. In: The 4th Workshop on Mathematical Reasoning and AI at NeurIPS'24 (2024), https://openreview.net/forum?id=8wDSfs1W3w

[64] Zhang, Z., Cheng, J.K., Deng, J., Tian, L., Ma, J., Qin, Z., Zhang, X., Zhu, N., Leng, T.: Diagram formalization enhanced multi-modal geometry problem solver. In: ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 1–5 (2025). https://doi.org/10.1109/ICASSP49660.2025.10889286

[65] Zhao, J., Zhang, T., Li, C., He, W., Wang, J., Huang, H.: Non-parametric structural priors for geometry theorem prediction. In: Forty-third International Conference on Machine Learning (2026), https://openreview.net/forum?id=XRzeAIhVaS

[66] Zou, J., Zhang, X., He, Y., Zhu, N., Leng, T.: Fgeo-drl: Deductive reasoning for geometric problems through deep reinforcement learning. Symmetry **16**(4) (2024)

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction accurately state the paper's core contributions, including the proposed RAVS architecture and its reported performance on the FormalGeo7K benchmark, and clearly define the scope as formal geometry problem solving.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

Justification: RAVS's performance on L6-level difficulty is limited.

Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

   Justification: We conducted a theoretical analysis of the superiority of bidirectional reasoning.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We have set up a "Experimental Setup" section to specify the experimental environment in detail.

   Guidelines:

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

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

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

   Answer: [Yes]

   Justification: We provide all code, data, and reproduction guidelines in the appendix. A GitHub link will be included upon paper acceptance.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We have set up a "Experimental Setup" section to specify the experimental environment in detail.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We report mean accuracy for all main experiments.

   Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide all code, data, and reproduction guidelines in the appendix. A GitHub link will be included upon paper acceptance.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research fully complies with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [N/A]

    Justification: There is no societal impact of the work performed.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [N/A]

Justification: The paper poses no such risks.

Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: All assets used in this paper (code, data, models) are properly cited with their original sources.

Guidelines:

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

13. **New assets**

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

Answer: [Yes]

Justification: We have thoroughly revised and improved the FormalGeo7K dataset, with complete documentation and an open-source license.

Guidelines:

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

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

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

Answer: [N/A]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

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

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

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

Answer: [N/A]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [Yes]

Justification: We explicitly state that the work in this paper is based on an LLM-driven agent.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.