symmetry

MDPI

Article

A Hybrid Framework for Automated Geometric Problem-Solving by Integrating Formal Symbolic Systems and Deep Learning

Zhengyu Hu [1], Xiaokai Zhang [1], Cheng Qin [1,2], Yang Li [1] and Tuo Leng [1,2,*]

[1] School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China
[2] School of Future Technology, Shanghai University, Shanghai 200444, China
* Correspondence: tleng@shu.edu.cn

Abstract

Geometric problem-solving (GPS) has been a long-standing challenge in the fields of formal mathematics and artificial intelligence. To address the limitations of unidirectional approaches, we developed a neuro-symbolic system that integrates forward and backward reasoning. The neural component employs a gating-enhanced attention network to select candidate theorems, guiding the heuristic search and pruning irrelevant branches. The symbolic component is a bidirectional solver built on FormalGeo, which performs rigorous geometric relational reasoning and algebraic computation. The neural component predicts the theorems based on the current problem state, while the symbolic component applies these theorems and updates the problem state. These two parts interact iteratively until the problem is solved. The solving process is organized as a graph structure where facts and goals serve as nodes and theorems as edges, thereby generating a human-readable solution. The proposed neuro-symbolic system achieved an 89.63% problem-solving success rate (PSSR) on the FormalGeo7K dataset, surpassing the previous best result.

Keywords: symmetry in geometry; formal mathematics; geometric problem-solving; bidirectional reasoning

1. Introduction

Symmetry is observed in many phenomena, such as biology [1], physics [2], economics [3], and also in geometric problems (GPs). It exists not only in geometric shapes but also in the solving process of GP: namely, the symmetrical forward [4] and backward [5] reasoning processes. Geometric problem-solving (GPS) has long been a long-standing challenge [6,7] in the fields of formal mathematics and artificial intelligence due to the complexity of integrating symbolic reasoning and algebraic computation with visual perception. GPS typically integrates knowledge in both visual and textual forms [8,9], and the solving process involves relational reasoning and algebraic computation, making it difficult to incorporate into a unified framework. A typical geometric problem consists of both text and an image, as shown in Figure 1. Solving it requires multi-step geometric reasoning or algebraic calculation to achieve the goal.

Early methods for solving geometric problems can be broadly categorized into three types: algebraic methods [10], hybrid methods [11], and geometric methods [12]. Algebraic methods transform geometric problems into algebraic forms, enabling the use of algebraic tools for solving. Representative methods include Wu's method [10] and Gröbner bases method [13]. However, algebraic methods typically fail to produce human-readable proof processes. Hybrid methods [11,14] situate at the intersection of algebraic and geometric

Academic Editor: Alexei Kanel-Belov

Received: 25 February 2026
Revised: 24 March 2026
Accepted: 27 March 2026
Published: 30 March 2026

approaches, partially transforming geometric problems into algebraic forms by defining certain geometric invariants. This maintains interpretability while applying algebraic ideas for automated reasoning. However, the problem-solving capability of hybrid methods is limited by the defined geometric invariants, restricting them to solving specific classes of geometric problems. Geometric methods have undergone several development stages, each with distinct characteristics. The early stage featured search-based methods [4,15], such as backward reasoning and forward chaining. Subsequently, geometry expert systems, like the deductive database method [12], emerged. With advances in machine learning and optimization techniques, these methods [16–18] were gradually applied to geometric problem-solving, leading to a significant improvement in the level of automation. Particularly in recent years, with the rise in deep learning, data-driven connectionist methods [19–21] have gradually become mainstream. Nevertheless, numerous challenges persist, hindering further progress in this field.

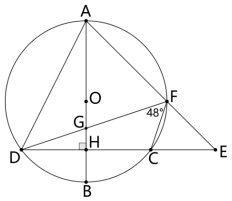

Image CDL
Equal(MeasureOfAngle(CFD),48)
PerpendicularBetweenLine(DH,GH)

Text CDL
Equal(MeasureOfAngle(CFD),48)
IsCentreOfCircle(O,O)
IsDiameterOfCircle(AB,O)
PerpendicularBetweenLine(DH,GH)

Construction CDL
Shape(OAD,DA)
Shape(AD,DG,GO,OA)
Shape(AO,OG,GF,FA)
Shape(OFA,AF)
Shape(GD,DH,HG)
Shape(FG,GH,HC,CF)
Shape(OCF,FC)
Shape(OCF,CE,EF)
Shape(BH,HD,ODB)
Shape(CH,HB,OBC)
Collinear(AOGHB)
Collinear(DGF)
Collinear(AFE)
Collinear(DHC)
Cocircular(O,ADBCF)

Problem Text
English: As shown in the diagram, ∠CFD=48°, O is the center of ⊙O, the diameter of ⊙O is AB, DH is perpendicular to GH. Find the measure of ∠EFC.

Image Annotation (parsed from .gbb)
Points: {A:[919,980], B:[1378,827], C:[919,1103], ..., O:[919,827]}

Theorem Sequences
circle_property_chord_perpendicular_bisect_arc(1,ODC,OHB)
arc_addition_length(1,ODB,OBC)
similar_arc_judgment_cocircular(1,ODB,ODC)
similar_arc_property_length_ratio(1,ODB,ODC)
similar_arc_property_measure_ratio(1,ODB,ODC)
arc_property_circumference_angle_external(1,ODB,A)
arc_property_circumference_angle_external(1,OAC,F)
arc_property_circumference_angle_external(1,ODC,F)
triangle_property_angle_sum(1,ADH)
arc_property_circumference_angle_internal(1,OAC,D)
adjacent_complementary_angle(1,EFC,CFA)

Goal CDL
Value(MeasureOfAngle(EFC))

Answer
66

Figure 1. Examples of geometric problems and formal annotations.

First, the field lacks network architectures specifically tailored for the formal solving of geometric problems, which must simultaneously address the unique demands of multimodal integration and the structural characteristics of formal geometric language. Most current methods suffer from a significant modality gap [8,22–25] by employing a sequential pipeline that separates image parsing, formal language processing, and theorem prediction. This approach not only leads to information loss and distortion but also allows early-stage errors to propagate downstream without global optimization or cross-validation mechanisms [26]. Furthermore, existing models typically rely on generative large language model architectures that lack specialized optimization for formal language. Such language is rigorously structured yet saturated with repetitive, templated tokens (e.g., brackets, quantifiers, and standard predicates) that act as logical connectors rather than core knowledge; traditional Transformer attention mechanisms, which treat all tokens equally, struggle to filter these redundancies, thereby wasting computational resources and impairing the perception of critical geometric entities. To overcome these challenges, we construct a unified multimodal framework equipped with a dynamic gating mechanism [27]. Our model takes both raw geometric diagrams and formal textual descriptions as input, performing joint encoding through a deep network to achieve feature-level fusion and end-to-end mapping from inputs to proof steps. Simultaneously, the integrated gating structure acts as a learnable information filter that adaptively suppresses non-essential structural tokens while enhancing attention toward core geometric elements. This tailored architecture not only enables cross-modal fusion and error correction but also ensures efficient extraction of

key reasoning information, significantly improving both the accuracy of theorem selection and the robustness of the overall solving process.

Second, the mainstream data-driven methods for solving geometric problem models, theorem prediction or proof-step generation as a sequence generation task [19,20]. Typically, models take the problem conditions as input and directly generate a long sequence of theorems until the final conclusion is reached. This one-step generation approach essentially reflects the current research's insufficient exploration of the dynamic formal process in geometry [21], failing to fully consider the unique structure of geometric reasoning. GPS is a structured state-space search process. The initial state of a problem evolves into a new intermediate state through the application of a theorem; this process repeats until the state satisfies the solving objective. At each specific intermediate state, the number of applicable theorems for the next step is limited. Therefore, modeling theorem prediction as a stepwise multi-class classification task is more natural and accurate than modeling it as a generation task. We transform the geometric problem-solving task into a theorem prediction task, further model it as a classification task, and implement a neuro-symbolic system that enables interaction between the neural network and the solver. Compared to existing systems, our approach not only allows step-by-step solving with interaction, resulting in clearer and more reliable solving processes, but also greatly reduces the difficulty of searching for and generating long sequences. It decomposes the complex global generation problem into a series of more manageable local decision-making problems.

Third, existing automated geometric solving systems generally adopt a unidirectional forward reasoning paradigm [12,19,28]: starting from the given conditions, they repeatedly apply theorems to derive new facts until the target conclusion is included in the derived set. While this approach is intuitive, it is often inefficient, especially for complex targets, as it can easily lead to blind searches in irrelevant directions, consuming substantial computational resources. In practice, human geometric problem-solving is both bidirectional and flexible. We not only explore forward from the given conditions but also perform backward analysis and decomposition starting from the target. Through this top-down, stepwise decomposition of goals, humans can more quickly focus on the key path, thereby devising more efficient solving strategies. We propose a unified bidirectional geometric reasoning framework. The core of this framework lies in integrating forward reasoning (expanding the state space from facts) and backward reasoning (decomposing the goal into subgoals) within a unified search space. The problem is considered solved when the solving paths from both directions meet at an intermediate state. This design enables the model to emulate the human analytical process, achieving higher efficiency and stronger strategic capability in solving complex problems.

In summary, we construct a neuro-symbolic framework for solving geometric problems, as illustrated in Figure 2. The neural component is a multimodal classification model enhanced with a gating mechanism, which at each step acts as a precise geometric state perception and decision unit. This model first performs deep fusion encoding of the input image and text, avoiding potential information loss that may arise from early unimodal parsing. It then employs a specially designed gated attention mechanism to actively filter structural redundancies in formal geometric language, focusing on the entities and relationships most critical to the current reasoning state. Finally, it precisely models the selection of theorems at each step as a classification task, predicting the most likely solving direction from a finite set of candidate theorems to advance the proof. The symbolic component is a formal geometric system built on FormalGeo, serving as a rigorous reasoning and verification execution engine. It receives decision instructions from the neural model and operates under a unified logical framework capable of both forward deduction and backward decomposition. Through the efficient synergy of a bidirectional search strategy, it

quickly constructs a proof path. More importantly, the system organizes the entire solving process—including all reasoning steps, algebraic computations, and intermediate conclusions—into a structured, fully traceable proof state graph. This graph not only ensures the logical rigor of each step but also naturally generates a clear, step-by-step, human-readable solving process. Consequently, while pursuing solving efficiency, it fundamentally guarantees the verifiability and explainability of geometric proofs.

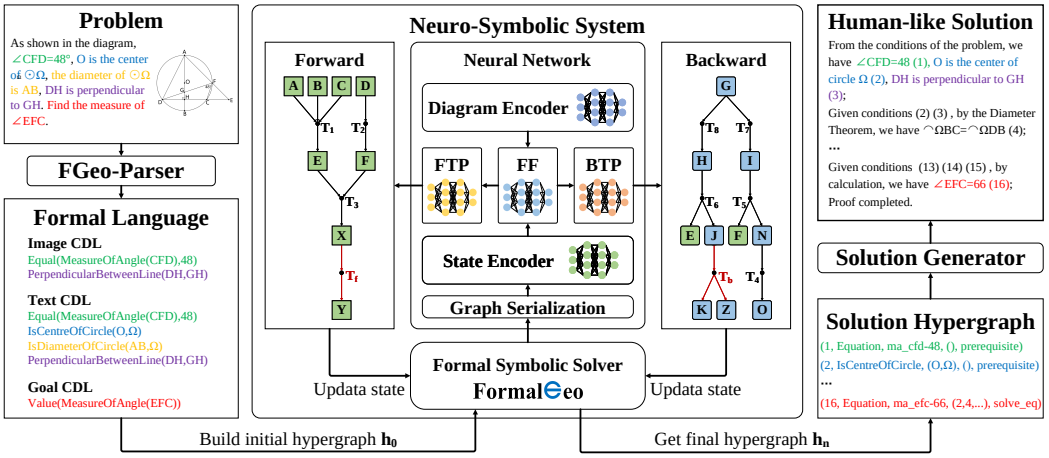

Figure 2. The neuro-symbolic system architecture proposed in this paper consists of three main components: the parser on the left, which formalizes geometry problems; the central neuro-symbolic problem-solving system, where the neural part is a gating-enhanced attention network and the symbolic part is a bidirectional solver based on FormalGeo; and the solution process generator on the right, which converts structured solution steps into human-readable output. The problem is initially parsed into a formal representation and fed into the formal symbolic solver. Within the solver, the problem state is characterized by a dual-graph structure comprising a forward hypergraph and a backward composition graph. In the neuro-symbolic module, both the current problem state and the associated geometric image are jointly encoded by a neural network to predict the theorem required for the next deductive step. The predicted theorem is then dispatched to the solver, where it is applied within both the forward and backward reasoning processes to update the problem state. This iterative predict-apply cycle continues until the problem is resolved. Upon completion, the final problem state is passed to a solution generator to synthesize a human-readable solution. A detailed description of the symbolic solver is provided in Section 3.1. The network architecture and training configurations are detailed in Sections 3.2 and 4.3.

Furthermore, we qualitatively compare our method with existing approaches across three dimensions comprising twelve criteria: system architecture (neuro-symbolic, multimodal input), solving modes (automatic and interactive mode and forward- and backward-solving), and user-friendly features (human-like solutions, traceable solutions, and extensible formal systems), as shown in Table 1. Our method is the only one that fully supports all evaluated features, setting it apart from prior works that lack comprehensive coverage. We also wish to clarify the distinctions between our approach and existing neuro-symbolic methods. First, unlike prior approaches that formulate geometric problem-solving as theorem generation [19,29] or sequence generation [20,30] tasks, we formulate it as a step-wise theorem classification problem. This formulation enables finer-grained control over the neuro-symbolic reasoning process. Second, whereas existing neural networks [31,32] often lack specialized mechanisms for formal languages, we introduce a gating mechanism specifically designed to filter out redundancy and markers within formal language. We also employ a multimodal model to fully integrate and align the image-text information inherent in geometric problems. In contrast, most existing methods [21,33] adopt a pipeline that first parses images into text, a process prone to information loss. Finally, in contrast to

existing symbolic systems [34,35] built upon FormalGeo, our method implements a unified bidirectional solving process, thereby significantly enhancing problem-solving efficiency.

Table 1. Feature comparison between our method and other geometry systems. (✓, – and × indicate full support, partial support, and no support, respectively. NS: neuro-symbolic system, MI: multimodal input, AM: automatic mode, IM: interactive mode, FS: forward-solving, BS: backward-solving, HLS: human-like solution, TS: traceable solution, EFS: extensible formal system).

Methods	System		Solving Mode				User-Friendly		
	NS	MI	AM	IM	FS	BS	HLS	TS	EFS
Backward [5]	×	×	✓	×	×	✓	×	×	×
Forward [4]	×	×	✓	×	✓	×	×	×	×
Wu's method [10]	×	×	✓	×	✓	×	×	×	×
JGEX [36]	×	×	✓	✓	✓	×	–	✓	–
GCLC [37]	×	×	✓	×	✓	×	–	–	×
GeoGebra [38]	×	×	✓	✓	✓	×	–	–	×
LeanEuclid [39]	✓	×	×	✓	✓	×	✓	✓	✓
NGS [20]	✓	✓	×	✓	✓	×	×	×	×
Inter-GPS [19]	✓	×	✓	✓	✓	×	×	×	×
DualGeoSolver [30]	✓	✓	✓	✓	✓	×	×	×	×
FGeo-TP [34]	✓	×	✓	✓	✓	×	✓	✓	✓
FGeo-DRL [35]	✓	×	✓	✓	✓	×	✓	✓	✓
FGeo-HyperGNet [21]	✓	×	✓	✓	✓	×	✓	✓	✓
DeepSeek v3 [40]	×	×	✓	✓	✓	✓	✓	×	×
FGPS [28]	×	×	✓	✓	✓	–	✓	✓	✓
DDAR [41]	×	×	✓	–	✓	×	✓	✓	–
Ours	✓	✓	✓	✓	✓	✓	✓	✓	✓

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

1. This paper presents a unified framework to integrate both forward and backward processes for geometry problem-solving. At every intermediate state of the solution, the system allows for the arbitrary selection of applying theorems to expand known conditions or decompose goals while achieving automatic information updates between the two directions: specifically, when a new node is added in the forward process, the corresponding goal's state in the backward process is automatically updated; conversely, once a sub-goal tree in the backward process is solved, the relevant theorem is automatically applied to the forward process, adding the goal's root node to the forward deduction chain.
2. This paper introduces a neural network architecture specifically designed for geometric problem-solving. Formal languages are characterized by significant redundancy and marker information. Within standard attention architectures, such noise tends to reduce the attention scores allocated to critical tokens. To address this, we integrate a gating mechanism into the attention framework, enabling it to dynamically filter out redundant content based on the input, thereby directing the network's focus toward essential tokens. The core of this architecture consists of attention modules enhanced with gating mechanisms, which support multimodal input. This design effectively captures multimodal geometric information while filtering out redundancy, enabling more efficient theorem prediction. We also propose a novel method for synthesizing training data based on our bidirectional reasoning process, which significantly enhances the network's generalization capabilities.
3. This paper constructs a neuro-symbolic system for solving geometric problems. The neural component is a multimodal classification model enhanced with a gating mechanism, which predicts potentially applicable theorems at each problem state during

the solving process. The symbolic component is a formal geometric system built on FormalGeo, serving as a rigorous reasoning and verification engine that performs forward deduction and backward decomposition while organizing the entire solving process into a structured graph. The neural and symbolic components interact iteratively until the problem is successfully solved.

4. This paper examines the differences among existing GPS methods from four systematic dimensions: information fusion mechanism (multimodal vs. unimodal), problem modeling paradigm (classification task vs. generation task), methodological architecture (neuro-symbolic methods vs. purely symbolic/purely neural methods), and solving-process modeling (bidirectional reasoning vs. unidirectional reasoning). Experiments show multimodal methods outperform unimodal methods; modeling theorem prediction as a classification task outperforms generation tasks; neuro-symbolic frameworks outperform purely symbolic or purely neural methods; and bidirectional reasoning outperforms forward or backward reasoning alone. These experimental findings not only confirm the validity of the improvements proposed in this work but also provide empirical design guidelines for subsequent research.
5. The proposed neuro-symbolic system achieved an 89.63% problem-solving success rate (PSSR) on the FormalGeo7K dataset, surpassing the previous best result of FGeo-HyperGNet (88.36%).

2. Preliminaries

This chapter categorizes geometric problem-solving methods from four dimensions: information fusion mechanism (multimodal vs. unimodal), problem modeling paradigm (classification vs. generation tasks), methodological architecture (neuro-symbolic vs. purely symbolic/purely neural methods), and reasoning strategy (bidirectional vs. unidirectional forward reasoning). For each category, the characteristics, strengths, and limitations of the methods are clarified. Experiments demonstrate that in the context of geometric problem-solving, multimodal fusion outperforms unimodal pipelines; modeling theorem prediction as a classification task outperforms generation tasks; neuro-symbolic frameworks outperform purely symbolic or purely neural approaches; and bidirectional reasoning strategies outperform forward or backward reasoning alone.

2.1. Multimodal Methods vs. Unimodal Methods

From the perspective of information fusion and representation, existing geometric problem-solving methods are primarily categorized into two types based on their input modalities: multimodal methods and unimodal methods.

Unimodal methods rely exclusively on text as input. These approaches either use the problem's textual description directly or attempt to first parse the geometric diagram and convert it into a textual format (such as natural language or formal descriptions of geometric entities and their relationships), which is then merged with the original problem text and fed into the solver. However, this "image-to-text" process is essentially a form of lossy compression. The high-density visual information inherent in geometric diagrams—such as precise spatial positions, subtle geometric constraints, and intuitive topological structures—is highly prone to loss or distortion when converted into discrete textual symbols. Even with advanced parsing algorithms, the resulting textual representation struggles to fully reproduce the rich semantic details of the original image, preventing the model from fully leveraging critical cues within the diagram.

Multimodal methods, in contrast, take both the raw image and the textual description (whether in natural language or formalized form) as joint inputs, achieving deep fusion at the feature level through joint encoding mechanisms. The core advantage of this approach

lies in the sufficiency of information representation. As a high-bandwidth information carrier, images can preserve the complete visual semantics of geometric problems with extremely high density. The multimodal framework avoids the forced discretization of continuous, dense visual signals into sparse text, enabling the model to directly capture geometric intuitions and implicit relationships that are difficult to describe precisely with language. By directly utilizing raw pixel information, multimodal methods achieve complementarity between textual semantics and visual features, thereby constructing a more comprehensive and robust problem representation than relying solely on text (whether original or parsed).

Current research trends indicate that, compared to the unimodal path, which attempts to transfer images into text through complex parsing pipelines, the multimodal paradigm that directly fuses images and text better unleashes the intrinsic value of the data. No longer constrained by the limitations of textual expression, it fully leverages the naturally high information density of images, providing a more solid informational foundation for solving complex geometric reasoning tasks.

2.2. Classification Task vs. Generation Task

The process of solving geometric problems, or proving geometric theorems, can be viewed as the sequential application of known theorems, and thus can be represented as a theorem sequence. In deep learning-based systems for geometric problem-solving, geometric problems are often transformed into theorem sequence prediction tasks. For the problem of geometric theorem sequence prediction, current mainstream methods primarily revolve around two modeling paradigms.

The first model's theorem sequence prediction as a multi-step classification task. This method deconstructs the proof process into a sequential decision-making problem. At each step, the neural network predicts the theorem t most likely to be applied in the next step based on the current geometric state s, i.e., maximizing the conditional probability $P(t \mid s)$, as shown in Equation (1), where θ denotes the parameters of the neural network. This process is typically tightly coupled with a symbolic solver: after the neural network outputs a theorem choice, the symbolic solver executes the theorem, updates the geometric state, and feeds the new state (s') back to the neural network for the next prediction. The strengths of this approach are its strong controllability, high interpretability, verifiability of each step, and the traceability of states. Moreover, since the decision space is restricted to a finite set of theorems, the model converges more easily. Its limitations lie in its dependence on frequent neuro-symbolic interaction, which may reduce overall solving speed, and the need for backtracking mechanisms in case of prediction errors, which increases procedural complexity.

$$\theta^* = \arg\max_{\theta} \sum_{i=1}^{N} P(t_i \mid s_{i-1}; \theta) \tag{1}$$

The second approach models it as a sequence generation task. This method aims to directly generate the complete sequence of theorem applications $(t_1, t_2, ..., t_n)$ from the initial state of the problem using a neural network, i.e., modeling it as $P(t_1, t_2, ..., t_n \mid s_0)$, as shown in Equation (2), where θ denotes the parameters of the neural network. The generation process is usually independent of the symbolic solver. The neural network outputs multiple candidate theorem sequences at once, and the symbolic solver then verifies each sequence to determine whether it constitutes a valid proof. The advantage of this approach is its efficient, parallelizable reasoning process, which avoids repeated state transfer and interaction, making it suitable for batch processing. However, the search space for generating long sequences is extremely large, often resulting in invalid or

redundant steps. Moreover, the generation process lacks intermediate verification, meaning the logical correctness of the final output sequence is not guaranteed, leading to lower overall reliability.

$$\theta^* = \arg\max_{\theta} \prod_{i=1}^{N} P(t_i \mid t_1, t_2, \ldots, t_{i-1}, s_0; \theta) \tag{2}$$

In summary, the classification paradigm aligns more closely with the human problem-solving habit of step-by-step reasoning and careful verification, emphasizing controllability and reliability throughout the process. In contrast, the generation paradigm attempts to maximize efficiency through "one-shot prediction," yet in complex geometric scenarios, there remains substantial room for improvement in the output quality and stability of this approach.

2.3. Neuro-Symbolic Methods vs. Non-Neuro-Symbolic Methods

Current mainstream automatic geometric solving methods, rooted in the two foundational AI paradigms of symbolism (emphasizing logical reasoning) and connectionism (advocating parallel learning of neurons), often manifest as two extremes: purely symbolic methods [12] that rely on logic and symbolic computation, and purely neural methods [42,43] that depend on data and pattern matching. While purely symbolic approaches are logically rigorous and ensure proof correctness, their search space grows exponentially with problem complexity, leading to combinatorial explosion and struggles with problems requiring creative construction; conversely, purely neural methods possess strong pattern recognition capabilities to quickly generate plausible "proofs", yet their black-box reasoning lacks rigorous logical guarantees, rendering their steps potentially mathematically invalid and unverifiable. Consequently, when applying AI techniques to geometry, methods are naturally categorized into these symbolic and connectionist types, alongside a third emerging category: neuro-symbolic hybrid methods designed to bridge these inherent limitations.

Purely symbolic methods rely entirely on formal logical reasoning systems, such as rule-based theorem provers and algebraic elimination systems. Their core lies in establishing a rigorous set of geometric axioms and inference rules, performing exhaustive or heuristic searches through symbolic computation, and step-by-step deriving proofs. The greatest strength of this approach is its logical completeness and verifiability—every derivation strictly follows mathematical rules, and the generated proof process is fully traceable and verifiable. Theoretically, it can solve all provable geometric problems. However, its fatal drawback is the combinatorial explosion problem: as problem complexity increases, the search space grows exponentially. When faced with problems requiring creative auxiliary constructions or complex algebraic transformations, it often fails due to computational resource exhaustion, resulting in extremely low solving efficiency.

Purely neural methods, on the other hand, rely entirely on data-driven deep learning models, treating geometric problems as end-to-end pattern recognition tasks. Models learn the correspondence between geometric diagrams and text from large amounts of labeled data and directly output proof steps or conclusions. The advantages of this approach lie in its powerful pattern-matching capability and fast solving speed, enabling quick recognition of common geometric configurations and strong performance on simple problems without the need for explicit rule definitions. However, its fundamental limitation lies in its black-box nature: the internal reasoning process is not interpretable, and the mathematical correctness of the output steps cannot be guaranteed. When encountering complex problems outside the training data distribution, it may produce seemingly plausible but logically erroneous answers. More critically, it lacks formal verification capability, which is a fatal defect in the mathematical domain.

Neuro-symbolic methods combine the pattern recognition of neural networks with the logical reasoning of symbolic systems. This creates a framework where neural networks guide the process and symbolic systems verify the results. Specifically, the neural network learns strategies and perceives states by extracting geometric relationships from inputs to predict helpful theorems, which narrows the search space and offers guidance. Meanwhile, the symbolic system handles strict execution and verification: it takes the network's suggestions, applies theorems for formal derivation, checks each step's correctness, and feeds results back to the neural network to adjust its strategy. The neural component avoids the blind search of purely symbolic methods, while the symbolic component ensures the logical guarantees missing in purely neural approaches. In terms of solving efficiency, neural guidance enables the symbolic engine to avoid wasting resources on invalid branches, significantly improving search speed. In terms of reliability, each step is verified by the symbolic system, ensuring the strict correctness of the proof process. In terms of interpretability, the entire solving process can be organized by the symbolic system into a structured proof graph, enabling human-readable output.

2.4. Bidirectional Reasoning vs. Unidirectional Reasoning

Based on the starting point for solving geometric problems, methods can be divided into two categories: forward reasoning and backward reasoning, as shown in Figure 3. Forward reasoning starts from the given conditions and continuously applies theorems to derive new facts until the goal is included in the derived facts. Its main advantage is simplicity and ease of implementation, as it expands the state space step-by-step in a forward direction. However, its critical flaw is blind search. For complex problems, it often wastes computational resources on irrelevant paths, leading to combinatorial explosion. Backward reasoning starts from the goal, decomposing it backward into subgoals and gradually seeking prerequisite conditions that satisfy these subgoals. Its strength is strong goal orientation, allowing it to quickly focus on key paths. However, its limitation is the complexity of decomposition. It requires accurately judging which theorems to apply in reverse, and with too many subgoals, it still faces an expanding search space.

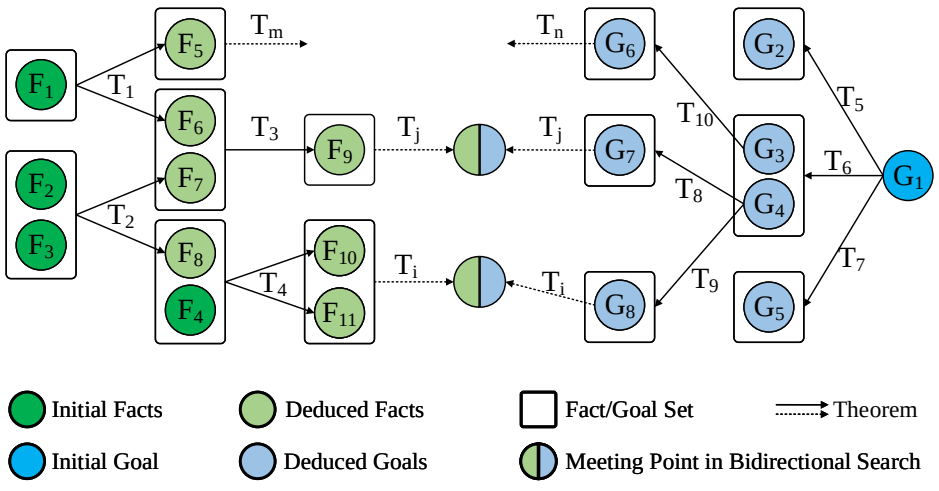

Figure 3. Forward reasoning and backward reasoning in geometric problems, and their convergence. Forward reasoning continuously expands existing facts (dark green) into new facts (light green) by applying theorems, while backward reasoning decomposes the goal (dark blue) into multiple sub-goals (light blue) through theorem application. The problem is considered solved when these two reasoning directions meet.

The common issue with unidirectional reasoning is its single strategy and lack of flexibility. Forward reasoning can be inefficient due to blind expansion, while backward

reasoning may fail due to erroneous decomposition. Both struggle to handle complex geometric problems requiring creative constructions. Bidirectional reasoning integrates forward and backward reasoning within a unified search space for collaborative problem-solving. Specifically, the system simultaneously performs forward deduction from the known conditions (expanding the fact set) and backward decomposition from the target conclusion (generating subgoals). When the search paths from both directions meet at an intermediate state, the proof chain is fully constructed. More importantly, the flexibility of the strategy is enhanced: the system can dynamically choose the more effective search direction based on the current state, simulating human problem-solving reasoning.

The core advantage of a unidirectional strategy lies in the improvement in search efficiency. First, we define a metric for problem difficulty. We represent the difficulty of a geometric problem by D, the length of the theorem sequence required to solve it. The number of nodes in the search tree increases exponentially with D; thus, a larger D implies a more difficult problem.

Consider a problem with difficulty D. We analyze forward search, backward search, and bidirectional search under this setting. Assume that in forward search, each node expands into M child nodes, and expanding a single node takes time a. The total time required for forward search to reach depth d is

$$T_{\text{forward}} = a \sum_{i=1}^{d} M^i = a(M^1 + M^2 + \cdots + M^d) \tag{3}$$

Similarly, assume that in backward search, each node expands into N child nodes, with an expansion time of b per node. Since backward search expands from depth D towards depth 0, the time required to reach a depth corresponding to $D - d$ (i.e., covering the remaining distance) is

$$T_{\text{backward}} = b \sum_{j=1}^{D-d} N^j = b(N^1 + N^2 + \cdots + N^{D-d}) \tag{4}$$

The superiority of bidirectional search over unidirectional search holds under the following assumptions. Specifically, the time to expand the last layer of the forward (or backward) search is greater than the time to expand the first layer of the backward (or forward) search:

$$a \cdot M^D > b \cdot N \quad \text{and} \quad b \cdot N^D > a \cdot M \tag{5}$$

Under these assumptions, there necessarily exists a depth d such that forward search is more efficient before this depth, while backward search becomes more efficient after it. This switching point satisfies

$$M^d \leq N^{D-d} \quad \text{and} \quad M^{d+1} \geq N^{D-d-1} \tag{6}$$

3. Neuro-Symbolic System

The architecture of the neuro-symbolic geometric problem-solving system is illustrated in Figure 2. The text and diagram of a geometric problem are first manually annotated or parsed by a parser into the FormalGeo formal language. This language, while maintaining readability, is automatically recognizable and interpretable by the symbolic solver. The symbolic solver parses the formal language and generates the initial state of the problem, with the forward process represented as a hypergraph and the backward process as a composition graph. The graph-structured state is serialized and input into the neural theorem predictor, which generates the theorems required for forward-solving and backward-solving, respectively. The symbolic solver accepts and executes the theorems,

continuously expanding facts or decomposing goals until the problem is successfully solved. After a successful solution, a human-readable solution process is generated based on the structured solving process and a rule-based solution generation algorithm.

The FormalGeo symbolic solver and the neural network are the core components of the system. This section provides a detailed description of the implementation methods for these two parts.

3.1. Symbolic System

Based on FormalGeo, we have constructed a geometric formal system that utilizes a formal language to describe geometric problems and configure solvers. The formal language comprises geometric definition language and condition declaration language. The former defines predicates and theorems for the geometric problem-solving system, while the latter is used to declare the given conditions and solve goals of geometric problems. Although serving different purposes, both adhere to the same syntactic format, similar to first-order logic. This design enables the FormalGeo formal language to be both highly readable and automatically processable by computer programs.

Based on geometric formalization theory and the FormalGeo formal geometric system, we have developed a geometric problem-solver using Python 3.10, whose architecture is shown in Figure 4. This solver accepts formally described problems as input and can execute various solving algorithms (forward and backward) using different strategies (depth-first, breadth-first, heuristic). Compared to existing solvers, this solver is the first to achieve true integration of forward and backward reasoning, addressing the challenge of determining when forward and backward processes converge.

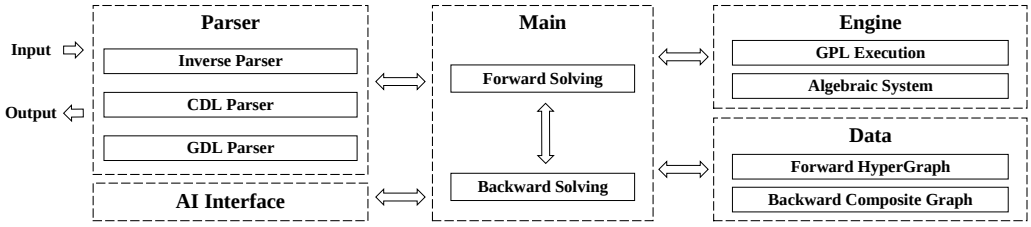

Figure 4. Architecture of the FormalGeo bidirectional symbolic solver implemented in Python. Formalized problems and formal system configurations are initially parsed by the parser module into internal data structures compatible with the solver. Subsequently, the main module runs forward and backward reasoning processes, coordinating interactions with the engine and data modules. Serving as the system's core, the engine module executes algebraic computation and logical deductions. The data module organizes the solving process into a graph structure, facilitating the generation of human-readable solutions.

In forward-solving, theorems are continuously applied to extend the set of facts. The solving process can be organized as a hypergraph, where facts serve as nodes and theorems as hyper-edges. In backward-solving, goals are recursively decomposed based on theorem definitions. The solving process can be structured as a composition graph, where goals are nodes and theorems are edges. Each goal can be represented as a tree composed of logical expressions and sub-goals, thereby representing the backward process as a composition graph. Whenever the forward process derives new facts, it checks whether the backward process includes the corresponding goal. Whenever a backward goal state is updated to known, the theorem at the root is applied forward, iteratively enabling interaction and state updates between the two directions.

Furthermore, geometric problems involve relational reasoning and algebraic computation. Internally, the solver translates theorem application into the execution of a geometric predicate logic, which is further implemented as a Cartesian product operation with constraints between two relations. The algebraic system is integrated into the unified reasoning

framework, where each algebraic relation is transformed into an equation and stored as several independent sets of dependent equation groups. This approach not only reduces redundant algebraic premises but also avoids unnecessary computations.

3.2. Neural Network

The architecture of the neural network component is illustrated in Figure 5. After the image of the geometric problem and the current state undergo tokenization and linear transformation, they are combined and augmented with positional encoding to form the embedding of the geometric problem. This embedding is then fed into multiple layers of gated enhanced attention for feature fusion, producing the overall encoding of the geometric problem. The encoding is input into a theorem prediction network composed of multiple layers of gated enhanced attention, which predicts the theorem probabilities required for forward-solving and backward-solving.

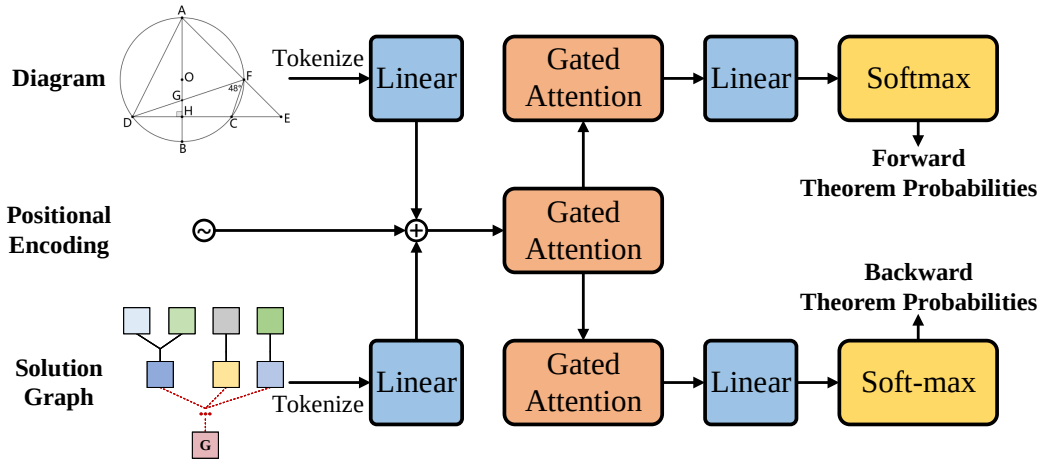

Figure 5. Architecture of the theorem prediction network. The image is patched, flattened, and projected via a linear transformation to obtain image embeddings. Meanwhile, the problem state is serialized and embedded to generate state embeddings. These two representations are concatenated, and positional encodings are added to form the final problem embedding. This problem embedding is then fed into separate forward and backward theorem prediction networks, yielding probability distributions for theorem application in both the forward and backward directions.

The core of the network is the gated enhanced attention module, which effectively filters redundant information in the state of the geometric problem. Its mathematical description is as follows:

QKVG Linear Projections: The input X is linearly transformed into queries $Q \in \mathbb{R}^{n \times d_k}$, keys $K \in \mathbb{R}^{n \times d_k}$, values $V \in \mathbb{R}^{n \times d_v}$, and gates $G \in \mathbb{R}^{n \times d_{\text{model}}}$ using learned weight matrices $W_Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_V \in \mathbb{R}^{d_{\text{model}} \times d_v}$, $W_G \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$.

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V, \quad G = XW_G \tag{7}$$

Scaled Dot-Product Attention: Computes attention scores between queries and keys, followed by a softmax normalization. $\frac{QK^T}{\sqrt{d_k}} \in \mathbb{R}^{n \times n}$ represents the scaled dot-product similarity matrix, and softmax$(\cdot)$ ensures the attention weights are non-negative and sum to 1 across each row.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{8}$$

Multi-Head Concatenation: In multi-head attention, the above process is repeated for h heads, with each $\text{head}_i = \text{Attention}(QW_Q^i, KW_K^i, VW_V^i)$ having its projection matrices W_Q^i, W_K^i, W_V^i. All heads' outputs are concatenated:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h) \tag{9}$$

Gating mechanism: The gating score is computed using G, where $\sigma(\cdot)$ is the sigmoid activate function, followed by element-wise multiplication with $\text{MultiHead}(Q, K, V)$ to obtain Y. This not only introduces non-linearity but also filters irrelevant information in the current problem state.

$$Y = \text{MultiHead}(Q, K, V) \odot \sigma(G) \tag{10}$$

Final Output Layer: The gated multi-head attention output Y is passed to the output layer $W_o \in \mathbb{R}^{hd_v \times d_{\text{model}}}$:

$$O = YW_o \tag{11}$$

The neural network undergoes two-stage training: Encoder Masked-Pretraining and task-specific end-to-end fine-tuning. During Encoder Masked-Pretraining, the image encoder and the problem state encoder are pretrained separately by masking a small number of image patches or text tokens and requiring the encoder to reconstruct them at the output side. This enables the encoder to grasp the fundamental structure of geometric problems. In the task-specific end-to-end fine-tuning stage, a large number of state-theorem pairs are generated based on annotated theorem sequences, enabling direct end-to-end training to improve the model's theorem prediction accuracy.

4. Experiments

4.1. Benchmark Methods and Evaluation Metrics

T5-small [44] and BART-base [45] are generative models used to compare the performance differences between generative tasks and classification tasks. Inter-GPS [19], NGS [20], and DualGeoSolver [30] are all neuro-symbolic systems. Forward search and backward search [28] are both symbolic-based methods. DeepSeek V3 [40] is a neural-based method. These groups of methods are used to compare the differences between neuro-symbolic fusion methods and purely symbolic or purely neural methods. FGeo-TP [34], FGeo-DRL [35], and FGeo-HyperGNet [21] are all solving systems based on FormalGeo, used to demonstrate the impact of differences in formal systems on problem-solving systems.

Additionally, to systematically evaluate the effectiveness of the proposed method and validate key design choices, we designed detailed ablation experiments. The experiments primarily focus on two key comparisons: the performance difference between multimodal and unimodal methods, and the efficiency impact of bidirectional versus unidirectional reasoning. To this end, we constructed several control models by adjusting the model architecture based on the baseline configuration, including a unimodal variant that only accepts text input and a unidirectional reasoning variant that only enables forward-solving. Furthermore, the effect of model parameter size and the effectiveness of the gating mechanism are also emphasized in the ablation analysis to comprehensively evaluate the contribution of each component to overall performance.

We tested the above models and our proposed method on the FormalGeo7K dataset. The FormalGeo7K dataset contains 7000 geometry problems with comprehensive annotations, which we divided into training, validation, and test sets in a 4:1:1 ratio. Based on

the length of the theorem sequences required to solve the problems, we categorized the test set into six difficulty levels: L_1 (length ≤ 3), L_2 ($4 \leq$ length ≤ 6), L_3 ($7 \leq$ length ≤ 9), L_4 ($10 \leq$ length ≤ 12), L_5 ($13 \leq$ length ≤ 15), and L_6 (length ≥ 16). The rationale for using theorem sequence length as a measure of difficulty has been sufficiently discussed in existing literature. For each geometry problem, successful solving is defined as obtaining the correct answer within a limited time. The final evaluation metric is the successful solving rate, calculated as the proportion of problems successfully solved in the test set for each method.

4.2. Data Synthesis

During the forward and backward-solving processes, the conclusions of certain theorems serve as premises for others, establishing structural relationships among theorems that can be organized into a graph structure called the Theorem Directed Acyclic Graph (TDAG). Specifically, in the forward hypergraph and backward composition graph, nodes correspond to derived geometric facts (forward) or sub-goals to be decomposed (backward), while directed edges represent the direction of theorem application. By ignoring the nodes in the graph and elevating the edges to new nodes, a directed acyclic graph with theorems as nodes, expressing reasoning dependencies, can be naturally constructed. The TDAGs obtained from the forward and backward processes are highly symmetrical.

As shown in Figure 6, for any intermediate state in the solving process, theorem nodes in the graph with no predecessors (zero in-degree) indicate that the corresponding theorem can be successfully executed in the forward direction under the current state. Similarly, theorem nodes with no successors (zero out-degree) indicate that the theorem can be used for backward decomposition in the current state. Each time a theorem is successfully applied, the corresponding theorem node and its associated edges are removed from the graph. This process of incrementally removing executable theorem nodes essentially performs alternating bidirectional topological sorting on the same DAG, which we refer to as bidirectional topological traversal of the TDAG. Any theorem sequence generated through this traversal is a valid solving sequence for the original geometric problem.

The same TDAG can generate a large number of valid topological sequences through different orders of bidirectional topological traversal, corresponding to diverse solving paths. Computing the number of topological traversals of a directed acyclic graph is a #P-complete problem, with no known polynomial-time algorithm for an exact solution. For the TDAG shown in Figure 6, which contains only eight theorems, bidirectional topological traversal yields 5818 valid theorem paths. Leveraging this property, we perform multiple random samplings of the bidirectional topological traversal process for each problem, specifically generating 50 different valid theorem sequences per problem. This approach constructs a larger, more comprehensive, and more diverse training and evaluation dataset, fully utilizing this combinatorial diversity to enhance data coverage and model generalization capability.

We clarify that the TDAG data synthesis strategy is not a generic augmentation strategy but an intrinsic component specifically designed for our bidirectional, step-by-step neural-symbolic framework. Consequently, it is technically incompatible with the baseline methods due to fundamental architectural mismatches. Pure symbolic methods (forward/backward search) and pre-trained LLMs (DeepSeek-V3) do not involve training phases. Sequence generation models (T5-small, BART-base, FGeo-TP) formulate geometry solving as a theorem sequence generation from the initial state. They lack the iterative, step-by-step reasoning mechanism required to utilize the TDAG data synthesis strategy. Methods like Inter-GPS, NGS, and DualGeoSolver rely on fundamentally different formal symbolic systems that cannot parse or execute the state-theorem pairs generated by our

TDAG data synthesis strategy. FGeo-DRL employs a reinforcement learning paradigm based on reward signals, whereas our synthesis is tailored for supervised learning. FGeo-HyperGNet employs a similar data augmentation strategy, consisting only of a forward process with no backward mechanism. However, for both our method and the baselines, the original dataset was split into the same training, validation, and testing sets using a consistent 4:1:1 ratio. Our TDAG-based data synthesis was used only to expand the training and validation sets for our proposed method. Crucially, the test set remained completely untouched and identical for all methods.

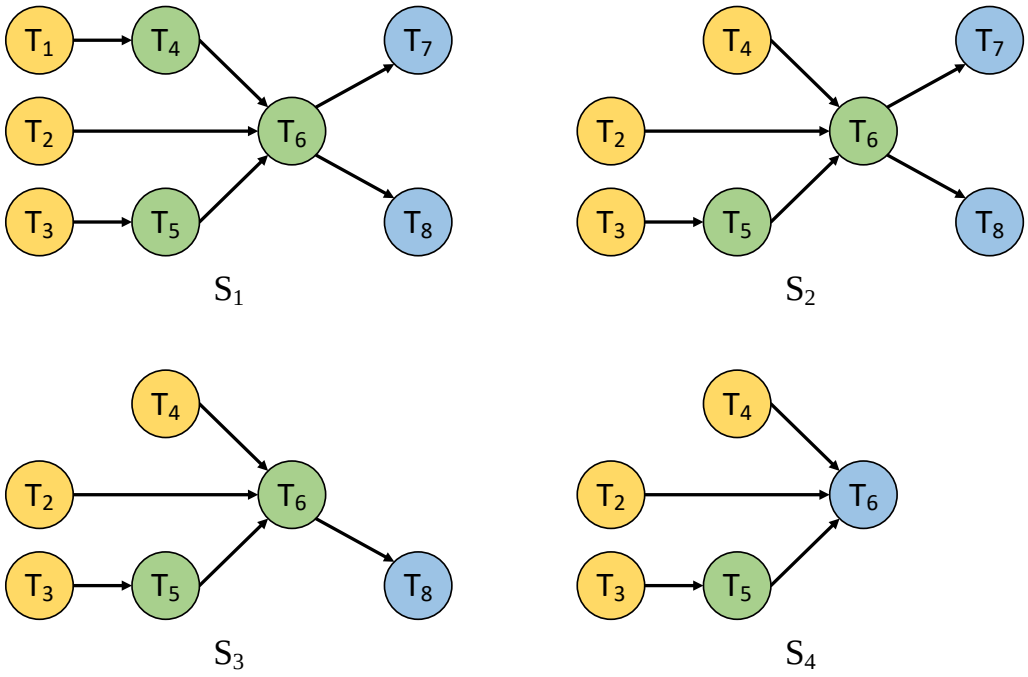

Figure 6. Schematic illustration of the synthetic data generation process using a bidirectional topological traversal. Yellow indicates theorems applicable to the forward process at the current state, while blue indicates those applicable to the backward process. Applying theorem T_1 to state S_1 expands the facts to yield state S_2; subsequently, applying theorem T_7 to S_2 decomposes the goal to obtain state S_3, and applying theorem T_8 to S_3 further decomposes the goal to reach state S_4. This process is repeated iteratively until all applicable theorems have been utilized.

4.3. Training Environment and Hyperparameters

All experiments were conducted on a Linux workstation equipped with an NVIDIA GeForce RTX 4090 GPU and an Intel i9-10900X CPU. To ensure experimental reproducibility, we fixed the random seed to 0 during both the data loading and model initialization stages.

The input images were uniformly resized to 256×256 pixels and partitioned into a sequence of 16×16 patches via a patch embedding layer. The maximum length of the sequence was set to 2048, while the maximum length of the state vector was limited to 800. The dataset was randomly split into training, validation, and testing sets in a ratio of 4:1:1. To accelerate data synthesis, we adopted a multi-process data loading strategy with a process utilization rate set to 0.9, and configured a task timeout threshold of 600 s to prevent deadlocks. Additionally, a random sampling strategy with 2 samples per augmentation was employed to increase sample diversity.

The core embedding dimension of the model, d_{model}, and the image feature dimension, d_{image}, were both set to 256. The architecture comprises $M = 4$ feature fusion modules and $N = 3$ theorem prediction modules. The number of heads for the multi-head attention mechanism, h, was set to 8. Under this setting, the model has only 8.85 million parameters.

To prevent overfitting, we applied dropout during training, with the dropout rate p_{drop} set to 0.5.

The model was trained for a total of 10 epochs. We utilized the adaptive learning rate algorithm Adam as the optimizer, with an initial learning rate set to 5×10^{-4}. The batch size during training was fixed at 32. During the testing phase, the process utilization rate was maintained at 0.9 to ensure evaluation efficiency. Under this setting, one epoch takes approximately 6 h.

During inference, we employ beam search to enhance the fault tolerance of neural network predictions. Let b denote the beam width and s_i represent the score of the i-th beam. For each beam, the neural network predicts the application probability p_j for each theorem j. If theorem j is successfully applied, a new beam is extended from the current one with an updated score of $s_i \times p_j$. All newly generated beams are then ranked by their scores, and the top b beams are selected to initiate the next iteration of AI-assisted beam search. The process terminates with a success status if any beam yields a valid solution; conversely, it returns a failure status if the number of active beams drops to zero.

4.4. Experimental Results

The experimental results presented in Table 2 demonstrate the performance comparison across different methods on geometric problem-solving with varying difficulty levels from L1 to L6. Our method achieves the highest overall success rate of 89.63%, outperforming all baseline approaches. Several notable patterns emerge from the data. First, methods based on large language models, namely DeepSeek V3, exhibit relatively consistent performance across different difficulty levels. However, despite this stability, their overall performance remains below that of our approach and several neuro-symbolic methods. Second, a clear performance degradation is observed across all methods as problem difficulty increases from L1 to L6, which is expected given the inherent complexity growth in geometric reasoning tasks.

Table 2. PSSR of different methods on the FormalGeo7K dataset. The timeout is set to 600 s.

Method	Total	L_1	L_2	L_3	L_4	L_5	L_6
Forward-BFS [28]	38.86	56.87	39.21	32.81	22.77	11.69	6.32
Forward-DFS [28]	36.16	52.51	40.42	24.72	18.10	17.53	8.80
Forward-RS [28]	39.71	55.52	42.10	35.41	20.88	12.34	8.80
Backward-BFS [28]	35.44	67.95	35.43	12.40	8.32	3.57	1.35
Backward-DFS [28]	33.73	65.72	32.76	11.73	7.45	3.25	1.13
Backward-RS [28]	34.05	66.94	32.55	11.58	7.30	3.57	1.13
T5-small [44]	36.14	47.66	40.00	35.45	22.54	15.52	1.96
BART-base [45]	54.00	73.83	56.39	48.88	35.21	25.86	9.80
Inter-GPS [19]	60.50	77.66	64.44	57.09	44.37	34.48	13.73
NGS [20]	62.60	65.07	68.15	66.45	58.11	54.29	37.50
DualGeoSolver [30]	62.11	65.75	67.52	67.10	59.46	42.86	35.42
FGeo-TP [34]	80.86	96.43	85.44	76.12	62.26	48.88	29.55
FGeo-DRL [35]	80.85	97.97	88.36	72.61	60.39	55.42	39.62
FGeo-HyperGNet [21]	88.36	95.96	94.17	90.67	74.65	65.52	58.82
DeepSeek v3 [40]	60.79	73.19	58.89	61.19	45.07	43.10	41.18
Ours	89.63	96.92	93.06	92.41	80.99	72.55	48.44

The comparison between generative models and classification-based approaches reveals significant advantages of the latter for geometric problem-solving. T5-small and BART-base, both designed for generative tasks, achieve overall success rates of only 36.14%

and 54.0%, respectively, substantially lower than our classification-based method. This performance gap can be attributed to the nature of geometric reasoning, which requires precise logical deduction rather than open-ended text generation. Generative models tend to produce plausible but potentially incorrect reasoning steps, whereas classification approaches can more accurately select from well-defined solution paths. The particularly poor performance of T5-small on harder problems (L5: 15.52%, L6: 1.96%) further emphasizes the limitation of generative approaches in handling complex geometric reasoning that demands exact logical consistency.

When examining the comparison between neuro-symbolic fusion methods and purely symbolic or purely neural approaches, the advantages of combining both paradigms become evident. Purely symbolic methods, including forward search and backward search variants, achieve overall success rates ranging from 33.73% to 39.71%. In the realm of purely neural approaches, strong general-purpose models like DeepSeek V3 reach 60.79%, demonstrating significant capabilities in pattern recognition. In contrast, neuro-symbolic methods such as Inter-GPS, NGS, DualGeoSolver, and our approach consistently demonstrate superior performance, with our method achieving 89.63%. We view this contrast not as a direct head-to-head benchmark under strictly equivalent conditions but rather as an illustration of fundamental paradigm differences: it underscores the unique value of integrating formal reasoning capabilities for specific domains like geometry. This suggests that while neural components provide effective heuristic guidance, the addition of symbolic components is crucial for ensuring logical rigor and interpretability. This suggests that neural components provide effective pattern recognition and heuristic guidance, while symbolic components ensure logical rigor and interpretability. The particularly strong performance of neuro-symbolic methods on harder problems (L5 and L6) indicates that the symbolic reasoning component becomes increasingly valuable as problem complexity grows, where pure neural approaches may struggle with maintaining logical consistency over extended reasoning chains.

The comparison among methods based on the FormalGeo formal system highlights the importance of the underlying formal representation in geometric problem-solving. FGeo-TP, FGeo-DRL, FGeo-HyperGNet, and our method all leverage FormalGeo, and they collectively outperform methods based on other formalisms or no explicit formal system. Among these, FGeo-HyperGNet achieves an 88.36% overall success rate, while our method further improves this to 89.63%. The consistent high performance across FormalGeo-based methods, particularly on easier problems where FGeo-TP reaches 96.43% on L1, demonstrates that a well-designed formal system provides a solid foundation for geometric reasoning. The improvement of our method over other FormalGeo-based approaches suggests that while the formal system is crucial, the specific architecture and reasoning strategy built upon it also significantly impact final performance. This is particularly evident in the L6 difficulty level, where our method achieves 48.44% compared to 58.82% for FGeo-HyperGNet, indicating room for improvement in handling the most challenging geometric problems even within the same formal framework.

4.5. Ablation Study

The ablation study results presented in Table 3 provide comprehensive insights into the contribution of each component in our proposed geometric problem-solving framework. The standard model achieves the highest overall success rate of 78.15% with beam size 5, outperforming all ablated variants across different difficulty levels. This confirms the effectiveness of our complete architecture design. Additionally, a consistent trend is observed across all experimental settings where increasing the beam size from 1 to 5 leads to improved performance. For instance, the standard model improves from 72.07% to

78.15%, while the text-only variant increases from 62.30% to 70.78%. This demonstrates that beam search effectively enhances the fault tolerance of theorem prediction by exploring multiple reasoning paths, which is particularly valuable in geometric reasoning where a single incorrect theorem selection can lead to solution failure.

Table 3. Ablation study results of the proposed method. All ablation experiments used the beam search strategy, with a timeout set to 60 s. Text-only refers to inputting only the problem state without the problem image; forward-only denotes solving the problem using solely the forward process; no gate indicates the exclusion of the gating mechanism defined in Equation (11); and small model refers to using a model with fewer parameters (2.29 million parameters compared to 8.85 million in the standard model).

Method	Beam Size	Total	L_1	L_2	L_3	L_4	L_5	L_6
Text-Only	1	62.30	83.33	66.88	56.70	35.54	17.65	17.19
	3	69.41	86.41	71.92	70.98	42.98	37.25	23.44
	5	70.78	87.95	73.82	70.09	50.41	37.25	18.75
Forward-Only	1	62.64	80.77	64.35	58.04	42.98	31.37	21.88
	3	70.69	84.62	77.92	69.2	50.41	31.37	25.00
	5	71.38	84.87	79.5	69.64	52.89	31.37	21.88
No Gate	1	65.38	82.56	67.51	63.39	47.93	33.33	15.62
	3	71.72	84.36	77.60	71.88	56.20	39.22	20.31
	5	70.18	84.36	76.97	67.86	53.72	31.37	20.31
Small Model	1	67.52	83.85	70.03	65.18	47.93	39.22	23.44
	3	73.95	89.74	78.55	72.32	56.20	37.25	23.44
	5	74.81	90.00	77.29	76.34	57.85	37.25	26.56
Standard Model	1	72.07	85.90	73.19	74.11	60.33	41.18	21.88
	3	77.38	91.79	79.81	79.91	66.94	39.22	18.75
	5	78.15	94.10	81.39	79.46	63.64	39.22	18.75

The comparison between text-only and the standard model reveals the critical importance of visual information in geometric problem-solving. The text-only variant achieves a 70.78% overall success rate compared to 78.15% for the standard model, representing a 7.37 percentage point improvement when visual input is incorporated. This performance gap is particularly pronounced on easier problems, where the standard model reaches 94.10% on L1 compared to 87.95% for text-only. Geometric problems inherently contain spatial relationships and structural information that are difficult to fully capture through textual descriptions alone. The visual input provides complementary geometric constraints and relationships that enable more accurate problem understanding and theorem selection, validating our multimodal design choice.

The bidirectional search strategy proves to be a key factor in improving problem-solving success. The forward-only variant achieves a 71.38% overall success rate, which is 6.77 percentage points lower than the standard model's 78.15%. This improvement stems from the complementary nature of forward and backward reasoning. Forward search efficiently explores reachable states from the initial conditions, while backward search works from the goal to identify necessary intermediate steps. By combining both directions, the system can more effectively navigate the search space and identify valid solution paths that might be missed by unidirectional search. This is particularly evident in medium-difficulty problems (L3 and L4), where the standard model shows consistent advantages over forward-only.

We also recorded the average solving time and search depth during testing for forward-only, backward-only, and bidirectional methods. With a timeout of 60 s and a beam size of 5, the average solving times (in seconds) for Forward-only, Backward-only, and Bidirectional

were 23.4064, 23.9958, and 22.704, respectively; the average search depths were 7.7344, 7.6632, and 5.4953, respectively. In terms of average solving time, the three methods performed similarly, with bidirectional achieving the best results. Regarding average search depth, bidirectional was significantly shorter than the unidirectional approaches. Generally, solving time increases exponentially with search depth. As the timeout increases, the gap between bidirectional and unidirectional methods is expected to become more pronounced in both average solving time and average search depth.

The gate mechanism demonstrates its value in managing information flow within the neural-symbolic architecture. The no gate variant achieves a 70.18% overall success rate, underperforming the standard model by 7.97 percentage points. Geometric problem states often contain numerous symbolic elements, many of which may be irrelevant to the current reasoning step. The gate mechanism effectively filters out redundant information, allowing the model to focus on relevant geometric relationships and theorems. Without this filtering, the model may be distracted by irrelevant symbols, leading to suboptimal theorem predictions and reduced solving efficiency. This finding underscores the importance of controlled information flow in complex reasoning tasks.

An analysis of loss dynamics during neural network training further highlights the benefits of the gate mechanism. Under identical training data conditions, the standard model achieves a mean loss and variance of 0.087 and 0.096, respectively, whereas the No Gate variant records higher values of 0.125 and 0.106. On one hand, the lower mean loss of the standard model compared to the no gate variant indicates that the non-linearity introduced by the gate mechanism enhances the network's representational capacity. On the other hand, the reduced loss variance suggests that the gate mechanism stabilizes the training process, thereby allowing for the use of larger learning rates. Additionally, the incorporation of the gate mechanism promotes a more uniform distribution of token attention, encouraging the model to focus on global information rather than local artifacts. During testing, the mean attention scores for the first token were 0.136 for the standard model and 0.658 for the no gate variant. This significant difference demonstrates that the gate mechanism enables the network to filter out attention from irrelevant tokens, preventing the model from excessively concentrating its attention solely on the first token.

The small model results provide interesting insights into the scaling behavior of neuro-symbolic systems. Despite having only 2.29 million parameters, approximately one-quarter of the standard model's 8.85 million parameters, the small model achieves a 74.81% overall success rate, which is only 3.34 percentage points below the standard model. This relatively small performance gap despite significant parameter reduction suggests favorable scaling properties for our architecture. Furthermore, the small model outperforms all other ablated variants except for the standard model. This efficiency demonstrates the potential of neuro-symbolic approaches to achieve strong reasoning capabilities with modest computational resources, suggesting that future improvements in data and compute could yield substantial gains.

Several additional observations merit discussion. As shown in Figure 7, the standard model outperforms all ablated variants in total PSSR under every beam size setting, validating the critical contribution of every integrated module. All ablation experiments converge within a single training epoch, indicating that our data synthesis method can effectively generate large quantities of high-quality training data. However, this rapid convergence also suggests potential overfitting concerns, particularly given the current train-test split strategy based on geometric problem categories. Future work should consider finer-grained dataset partitioning, using (state, theorem) pairs as the minimum unit to ensure better generalization. The remarkable efficiency of our models, with the standard model at only 8.85 million parameters outperforming much larger language models, highlights the inher-

ent advantages of neuro-symbolic fusion. By combining neural pattern recognition with symbolic logical reasoning, our approach achieves strong performance without requiring massive scale, pointing to a promising direction for efficient and interpretable geometric reasoning systems.

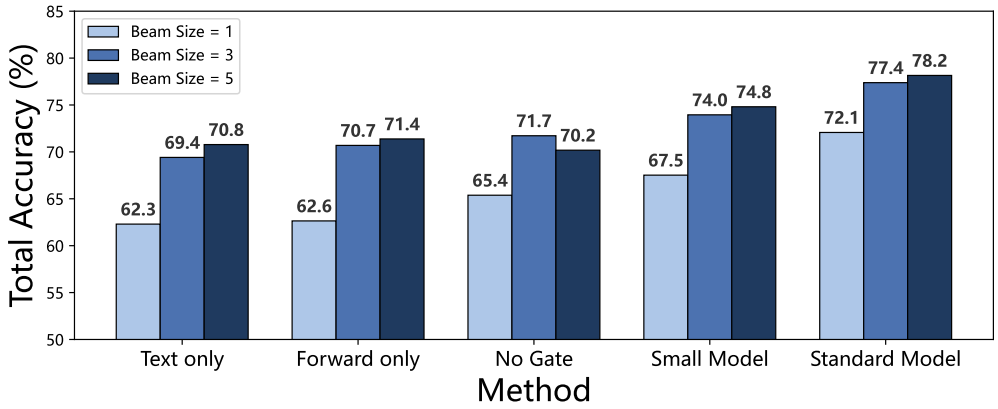

Figure 7. Impact of method and beam size on total PSSR.

5. Related Work

Geometric problem-solving and theorem proving have long been challenging [4–7] in the fields of automated reasoning and artificial intelligence due to their inherent complexity. Based on different technical approaches, methods for solving geometric problems can be broadly categorized into three stages: algebraic methods, synthetic methods, and geometric methods.

5.1. Algebraic Methods

Algebraic methods [10,13,46] transform geometric problems into algebraic form, thereby enabling their solution with algebraic techniques. Wen-Tsun proposed the Wu's method [47]. Then, by utilizing the properties of algebraic computation, these algebraic expressions are simplified and solved, transforming complex algebraic expressions into forms understandable by humans, and thus interpreting their geometric meanings. The study of algebraic approaches to geometry problems has given rise to a range of research achievements, such as Buchberger's Gröbner bases method [13], numerical parallel methods [48], polynomial system triangulation elimination algorithms [49], cylindrical algebraic decomposition for solving inequalities [50], dimensionality reduction methods [46], and software tools like GEOTHER [51]. These methods fully leverage the computational power of computers, but their problem-solving processes are not easily comprehensible to humans.

5.2. Synthetic Methods

Synthetic methods [11,52] lie at the intersection of algebra and geometry. They employ algebraic ideas to achieve automated reasoning, and their proofs are more readable than those produced by purely algebraic methods, although the range of problems they can represent is limited. Zhang proposed the point elimination method based on geometric invariants [11]. This approach employs constructive methods to describe problems and is capable of generating concise and meaningful readable proofs for a large number of non-trivial geometric problems. Subsequently, research on machine proofs of geometric theorems based on geometric invariants rapidly advanced [53–55], leading to the development of practical software tools such as Geometry Explorer [56], Geometry Expert [57] and Java Geometry Expert [36]. The method based on geometric invariants can also be extended to solid geometry [14] and non-Euclidean geometry [58]. The point elimination

method is a mechanized method but requires a lot of effort in discovering new invariants and shape construction, and the types of geometric problems that can be solved are limited.

5.3. Geometric Methods

Geometric methods have gone through several notable stages, starting from early search-based methods [4,5] to geometric expert systems [12,28]. Gelernter et al. developed the pioneering automated GPS system known as the Geometry Theorem Prover [15], which employed a backward search approach to solve pre-formalized problems. Nevins pointed out that the forward chaining method [4] can also be effective by efficiently representing the known conditions of the problem and limiting the typical application of those conditions.

In the early 2000s, with the advancement of machine learning and optimization theory, geometric methods based on these technologies began to emerge, significantly enhancing the automation level of problem-solving systems. The development of geometry problem-solving has led to the emergence of various downstream tasks, including geometry problem formalization [59,60], geometric knowledge extraction [18,61–64], geometric diagram parsing [8,65,66], geometric theorem proving [67–69], and geometry problem-solving [16,17,70–73].

In the current deep learning era, there is a debate between two methodological approaches. The first approach [22,24,42,74–79] employs general-purpose models with natural language reasoning. This approach generally entails pretraining a base model on extensive mathematical or reasoning datasets, followed by post-training fine-tuning to augment its reasoning performance. Although this strategy avoids constructing a formal system, it faces challenges in verifying the correctness of the model's outputs and cannot supply dependable supervision signals. The second approach [21,39,80] adopts specialized models with formal language (or an explainable sequence of operations and numbers [30,33,81–83], even Python code [84]) reasoning. This approach depends on human experts to pre-construct a formal system, which is then employed to provide supervision signals for model training or to automatically generate training data. Such a framework enables the development of more reliable systems and facilitates integration with self-improving algorithms, ultimately constructing problem-solving systems capable of exceeding human performance.

6. Conclusions

We present a neuro-symbolic framework for geometric problem-solving, where a gating-enhanced multimodal neural network guides a formal symbolic solver through iterative, bidirectional reasoning. The neural component actively filters geometric language and predicts theorems, while the symbolic component executes rigorous deduction within a structured proof graph. This synergy yields an 89.63% problem-solving success rate on FormalGeo7K and produces human-readable, verifiable proofs.

Author Contributions: Conceptualization, Z.H. and X.Z.; methodology, Z.H.; software, Z.H.; validation, Z.H., C.Q. and Y.L.; formal analysis, Z.H.; investigation, Z.H.; resources, Z.H.; data curation, Z.H.; writing—original draft preparation, Z.H.; writing—review and editing, Z.H., X.Z., C.Q., Y.L. and T.L.; visualization, Z.H.; supervision, T.L.; project administration, Z.H.; funding acquisition, T.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Natural Science Foundation of China (NSFC) grant 12071282.

Data Availability Statement: The code and datasets are avaiable at https://github.com/huzhengyu2016/FGeo-NSS (accessed on 24 March 2026).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Ocklenburg, S.; Mundorf, A. Symmetry and asymmetry in biological structures. *Proc. Natl. Acad. Sci. USA* **2022**, *119*, e2204881119. [CrossRef]
2. Sundermeyer, K. *Symmetries in Fundamental Physics*; Springer: Cham, Switzerland, 2014; Volume 176.
3. Fukang, F. The symmetry approach on economic systems. *Chaos Solitons Fractals* **1996**, *7*, 2247–2257. [CrossRef]
4. Nevins, A.J. Plane geometry theorem proving using forward chaining. *Artif. Intell.* **1975**, *6*, 1–23. [CrossRef]
5. Gelernter, H.; Hansen, J.R.; Loveland, D.W. Empirical explorations of the geometry theorem machine. In *IRE-AIEE-ACM '60 (Western), Proceedings of the Western Joint IRE-AIEE-ACM Computer Conference*; Association for Computing Machinery: New York, NY, USA, 1960; pp. 143–149. [CrossRef]
6. 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* **2022**, arXiv:2210.15767. [CrossRef]
7. XTX Markets. Artificial Intelligence Mathematical Olympiad Prize (AIMO Prize). 2023. Available online: https://aimoprize.com/ (accessed on 24 March 2026).
8. Zhang, M.L.; Yin, F.; Hao, Y.H.; Liu, C.L. Plane Geometry Diagram Parsing. In Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI-22), Vienna, Austria, 23–29 July 2022; pp. 1636–1643. [CrossRef]
9. Zhao, J.; Zhang, T.; Sun, J.; Tian, M.; Huang, H. Pi-GPS: Enhancing Geometry Problem Solving by Unleashing the Power of Diagrammatic Information. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Honolulu, HI, USA, 19–20 October 2025; pp. 1526–1536.
10. Wu, W.T. Basic principles of mechanical theorem proving in elementary geometries. *J. Autom. Reason.* **1986**, *2*, 221–252. [CrossRef] [PubMed]
11. 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. *Ann. Math. Artif. Intell.* **1995**, *13*, 109–137. [CrossRef]
12. Chou, S.C.; Gao, X.S.; Zhang, J.Z. A Deductive Database Approach to Automated Geometry Theorem Proving and Discovering. *J. Autom. Reason.* **2000**, *25*, 219–246. [CrossRef]
13. Buchberger, B. Applications of Gröbner bases in non-linear computational geometry. In *Trends in Computer Algebra*; Springer: Berlin/Heidelberg, Germany, 1988; pp. 59–87.
14. Chou, S.C.; Gao, X.S.; Zhang, J.Z. Automated production of traditional proofs in solid geometry. *J. Autom. Reason.* **1995**, *14*, 257–291. [CrossRef]
15. Gelernter, H.L. Realization of a geometry theorem proving machine. In *Proceedings of the International Conference on Information Processing (Proceedings of the First International Conference on Information Processing, Paris, France, 15–20 June 1959)*; UNESCO: Paris, France, 1959; pp. 273–281.
16. Alvin, C.; Gulwani, S.; Majumdar, R.; Mukhopadhyay, S. Synthesis of geometry proof problems. In *AAAI'14, Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*; AAAI Press: Washington, DC, USA, 2014; pp. 245–252.
17. 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), Lisbon, Portugal, 17–21 September 2015; pp. 1466–1476. [CrossRef]
18. Sachan, M.; Dubey, A.; Hovy, E.H.; Mitchell, T.M.; Roth, D.; Xing, E.P. Discourse in Multimedia: A Case Study in Extracting Geometry Knowledge from Textbooks. *Comput. Linguist.* **2020**, *45*, 627–665. [CrossRef]
19. 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), Online, 1–6 August 2021; pp. 6774–6786. [CrossRef]
20. 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 Proceedings of the Findings of the Association for Computational Linguistics (ACL-IJCNLP), Online, 1–6 August 2021; pp. 513–523. [CrossRef]
21. 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), Montreal, QC, Canada, 16–22 August 2025; pp. 4733–4741. [CrossRef]
22. Mouselinos, S.; Michalewski, H.; Malinowski, M. Beyond Lines and Circles: Unveiling the Geometric Reasoning Gap in Large Language Models. In Proceedings of the Findings of the Association for Computational Linguistics (EMNLP), Miami, FL, USA, 12–16 November 2024; pp. 6192–6222. [CrossRef]
23. Li, Z.Z.; Zhang, M.L.; Yin, F.; Liu, C.L. LANS: A Layout-Aware Neural Solver for Plane Geometry Problem. In Proceedings of the Findings of the Association for Computational Linguistics (ACL), Bangkok, Thailand, 11–16 August 2024; pp. 2596–2608. [CrossRef]

24. Zhang, R.; Jiang, D.; Zhang, Y.; Lin, H.; Guo, Z.; Qiu, P.; Zhou, A.; Lu, P.; Chang, K.W.; Qiao, Y.; et al. MATHVERSE: Does Your Multi-modal LLM Truly See the Diagrams in Visual Math Problems? In *Computer Vision—ECCV 2024, Proceedings of the 18th European Conference, Milan, Italy, 29 September–4 October 2024*; Springer: Berlin/Heidelberg, Germany, 2025; pp. 169–186.
25. Guo, Z.; Liu, M.; Wang, Q.; Ji, Z.; Bai, J.; Zhang, L.; Zuo, W. Integrating Visual Interpretation and Linguistic Reasoning for Geometric Problem Solving. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Honolulu, HI, USA, 19–20 October 2025; pp. 3988–3998.
26. Chen, J.; Li, T.; Qin, J.; Lu, P.; Lin, L.; Chen, C.; Liang, X. UniGeo: Unifying Geometry Logical Reasoning via Reformulating Mathematical Expression. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP), Abu Dhabi, United Arab Emirates, 7–11 December 2022; pp. 3313–3323. [CrossRef]
27. Qiu, Z.; Wang, Z.; Zheng, B.; Huang, Z.; Wen, K.; Yang, S.; Men, R.; Yu, L.; Huang, F.; Huang, S.; et al. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. *arXiv* **2025**, arXiv:2505.06708. [CrossRef]
28. 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* **2024**, *16*, 404. [CrossRef]
29. Peng, S.; Fu, D.; Liang, Y.; Gao, L.; Tang, Z. GeoDRL: A Self-Learning Framework for Geometry Problem Solving using Reinforcement Learning in Deductive Reasoning. In Proceedings of the Findings of the Association for Computational Linguistics (ACL), Toronto, ON, Canada, 9–14 July 2023; pp. 13468–13480. [CrossRef]
30. 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), Jeju, Republic of Korea, 3–9 August 2024; pp. 6559–6568. [CrossRef]
31. 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 *MM '25, Proceedings of the 33rd ACM International Conference on Multimedia (ACM MM)*; Association for Computing Machinery: New York, NY, USA, 2025; pp. 3057–3066. [CrossRef]
32. 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 *MM '25, Proceedings of the 33rd ACM International Conference on Multimedia (ACM MM)*; Association for Computing Machinery: New York, NY, USA, 2025; pp. 5394–5403. [CrossRef]
33. 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 Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 16–22 June 2024; pp. 13828–13837. [CrossRef]
34. He, Y.; Zou, J.; Zhang, X.; Zhu, N.; Leng, T. FGeo-TP: A Language Model-Enhanced Solver for Euclidean Geometry Problems. *Symmetry* **2024**, *16*, 421. [CrossRef]
35. Zou, J.; Zhang, X.; He, Y.; Zhu, N.; Leng, T. FGeo-DRL: Deductive Reasoning for Geometric Problems through Deep Reinforcement Learning. *Symmetry* **2024**, *16*, 437. [CrossRef]
36. Ye, Z.; Chou, S.C.; Gao, X.S. An Introduction to Java Geometry Expert. In *Automated Deduction in Geometry, Proceedings of the 7th International Workshop, ADG 2008, Shanghai, China, 22–24 September 2008*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 189–195.
37. Janičić, P. GCLC—A Tool for Constructive Euclidean Geometry and More Than That. In *Mathematical Software—ICMS 2006, Proceedings of the Second International Congress on Mathematical Software, Castro Urdiales, Spain, 1–3 September 2006*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 58–73.
38. Botana, F.; Hohenwarter, M.; Janičić, P.; Kovács, Z.; Petrović, I.; Recio, T.; Weitzhofer, S. Automated Theorem Proving in GeoGebra: Current Achievements. *J. Autom. Reason.* **2015**, *55*, 39–59. [CrossRef]
39. 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), PMLR, Vienna, Austria, 21–27 July 2024; Volume 235, pp. 36847–36893.
40. DeepSeek-AI. DeepSeek-V3 Technical Report. *arXiv* **2024**, arXiv:2412.19437. [CrossRef]
41. Chervonyi, Y.; Trinh, T.H.; Olšák, M.; Yang, X.; Nguyen, H.; Menegali, M.; Jung, J.; Kim, J.; Verma, V.; Le, Q.V.; et al. Gold-medalist Performance in Solving Olympiad Geometry with AlphaGeometry2. *arXiv* **2025**, arXiv:2502.03544. [CrossRef]
42. Zhang, J.; Moshfeghi, Y. GOLD: Geometry Problem Solver with Natural Language Description. In Proceedings of the Findings of the Association for Computational Linguistics (NAACL), Mexico City, Mexico, 16–21 June 2024; pp. 263–278. [CrossRef]
43. Zheng, K.; Han, J.M.; Polu, S. miniF2F: A cross-system benchmark for formal Olympiad-level mathematics. In Proceedings of the Tenth International Conference on Learning Representations (ICLR), Online, 25–29 April 2022.
44. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* **2020**, *21*, 1–67.
45. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020*; Jurafsky, D., Chai, J., Schluter, N., Tetreault, J., Eds.; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020; pp. 7871–7880. [CrossRef]

46. Yang, L. Practical automated reasoning on inequalities: Generic programs for inequality proving and discovering. In Proceedings of the Third Asian Technology Conference in Mathematics, Tsukuba, Japan, 24–28 August 1998.
47. Wu, W.T. On the decision problem and the mechanization of theorem proving in elementary geometry. *Sci. Sin.* **1978**, *21*, 157–179.
48. Yang, L.; Zhang, J.; Li, C. A prover for parallel numerical verification of a class of constructive geometry theorems. *Proc. IWMM* **1992**, *92*, 244–250.
49. Gao, X.S.; Chou, S.C. On the dimension of an arbitrary ascending chain. *Chin. Sci. Bull.-Engl. Ed.* **1993**, *38*, 799.
50. Collins, G.E. Quantifier elimination for real closed fields by cylindrical algebraic decomposition–preliminary report. *ACM SIGSAM Bull.* **1974**, *8*, 80–90. [CrossRef]
51. Wang, D. GEOTHER: A geometry theorem prover. In *Automated Deduction—Cade-13, Proceedings of the 13th International Conference on Automated Deduction, New Brunswick, NJ, USA, 30 July–3 August 1996 Proceedings*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 166–170.
52. Chou, S.C.; Gao, X.S.; Zhang, J.Z. Automated generation of readable proofs with geometric invariants. *J. Autom. Reason.* **1996**, *17*, 349–370. [CrossRef]
53. Chou, S.C.; Gao, X.S.; Zhang, J.Z. Automated geometry theorem proving by vector calculation. In *ISSAC '93: Proceedings of the 1993 International Symposium on Symbolic and Algebraic Computation*; Association for Computing Machinery: New York, NY, USA, 1993; pp. 284–291.
54. Chou, S.C.; Gao, X.S.; Zhang, J.Z. *A Collection of 110 Geometry Theorems and Their Machine Proofs Based on Full-Angles*; TR-94-4; Computer Science Department, Wichita State University: Wichita, KS, USA, 1994.
55. Li, H. Symbolic computation in the homogeneous geometric model with Clifford algebra. In *ISSAC '04: Proceedings of the 2004 International Symposium on Symbolic and Algebraic Computation*; Association for Computing Machinery: New York, NY, USA, 2004; pp. 221–228.
56. Wilson, S.; Fleuriot, J.D. Geometry Explorer: A tool for generating diagrammatic full-angle method proofs. In *Proceedings of the 6th International Workshop on Automated Deduction in Geometry, Narbada, India, 25–27 August 2006*; Narbada: Narbada, India, 2006; pp. 144–150.
57. Chou, S.C.; Gao, X.S.; Zhang, J.Z. An introduction to geometry expert. *Proc. CADE* **1996**, *1104*, 235–239.
58. 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: Proceedings of the International Workshop on Automated Deduction in Geometry, Toulouse, France, 27–29 September 1996 Selected Papers 1*; Springer: Berlin/Heidelberg, Germany, 1997; pp. 171–188.
59. Gan, W.; Yu, X. Automatic understanding and formalization of natural language geometry problems using syntax-semantics models. *Int. J. Innov. Comput. Inf. Control.* **2018**, *14*, 83–98.
60. Rao, Y.; Xie, L.; Guan, H.; Li, J.; Zhou, Q. A Method for expanding predicates and rules in automated geometry reasoning system. *Mathematics* **2022**, *10*, 1177. [CrossRef]
61. Sachan, M.; Dubey, K.; Xing, E. From Textbooks to Knowledge: A Case Study in Harvesting Axiomatic Knowledge from Textbooks to Solve Geometry Problems. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP), Copenhagen, Denmark, 9–11 September 2017; pp. 773–784. [CrossRef]
62. Yu, W.; Wang, M.; Wang, X.; Zhou, X.; Zha, Y.; Zhang, Y.; Miao, S.; Liu, J. GeoRE: A relation extraction dataset for chinese geometry problems. In Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021) Workshop on Math AI for Education (MATHAI4ED), Online, 6–14 December 2021.
63. Huang, L.; Yu, X.; He, B. A novel geometry problem understanding method based on uniform vectorized syntax-semantics model. In Proceedings of the 2022 International Conference on Intelligent Education and Intelligent Research (IEIR), Wuhan, China, 18–20 December 2022; pp. 78–85.
64. Zhou, W.; Xu, R.; Guan, H.; Zhao, J.; Rao, Y. Research on geometry problem text understanding based on bidirectional LSTM-CRF. In Proceedings of the 2022 9th International Conference on Digital Home (ICDH), Guangzhou, China, 28–30 October 2022; pp. 121–127.
65. Seo, M.J.; Hajishirzi, H.; Farhadi, A.; Etzioni, O. Diagram Understanding in Geometry Questions. *Proc. Twenty-Eighth AAAI Conf. Artif. Intell.* **2014**, *28*, 2831–2838. [CrossRef]
66. Wong, M.F.; Qi, X.; Tan, C.W. EuclidNet: Deep visual reasoning for constructible problems in geometry. In Proceedings of the 2nd MATH-AI Workshop at NeurIPS'22: Toward Human-Level Mathematical Reasoning, New Orleans, LA, USA, 28 November–9 December 2022.
67. Yu, X.; Wang, M.; Gan, W.; He, B.; Ye, N. A framework for solving explicit arithmetic word problems and proving plane geometry theorems. *Int. J. Pattern Recognit. Artif. Intell.* **2019**, *33*, 1940005. [CrossRef]
68. Gan, W.; Yu, X.; Zhang, T.; Wang, M. Automatically proving plane geometry theorems stated by text and diagram. *Int. J. Pattern Recognit. Artif. Intell.* **2019**, *33*, 1940003. [CrossRef]
69. Kovács, Z.; Yu, J.H. Automated discovery of geometrical theorems in geoGebra. *arXiv* **2022**, arXiv:2202.04627. [CrossRef]

70. Zhong, X.; Fu, H.; Yu, Y.; Liu, Y. Interactive learning environment based on knowledge network of geometry problems. In Proceedings of the 2015 10th International Conference on Computer Science & Education (ICCSE), Cambridge, UK, 22–24 July 2015; pp. 53–58.
71. Alvin, C.; Gulwani, S.; Majumdar, R.; Mukhopadhyay, S. Synthesis of solutions for shaded area geometry problems. In Proceedings of the Thirtieth International Flairs Conference, Marco Island, FL, USA, 22–24 May 2017.
72. 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), Vancouver, BC, Canada, 3–4 August 2017; pp. 251–261.
73. Yu, X.; Gan, W.; Wang, M. Understanding explicit arithmetic word problems and explicit plane geometry problems using syntax-semantics models. In Proceedings of the 2017 International Conference on Asian Language Processing (IALP), Singapore, 5–7 December 2017; pp. 247–251.
74. Duan, X.; Tan, D.; Fang, L.; Zhou, Y.; He, C.; Chen, Z.; Wu, L.; Chen, G.; Gong, Z.; Luo, W.; et al. Reason-and-Execute Prompting: Enhancing Multi-Modal Large Language Models for Solving Geometry Questions. In *MM '24, Proceedings of the 32nd ACM International Conference on Multimedia (ACM MM)*; Association for Computing Machinery: New York, NY, USA, 2024; pp. 6959–6968. [CrossRef]
75. Zeng, X.; Liu, S. Research on the application of knowledge mapping and knowledge structure construction based on adaptive learning model. *Expert Syst. Appl.* **2024**, *249*, 123400. [CrossRef]
76. 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), Suzhou, China, 4–9 November 2025; pp. 718–735. [CrossRef]
77. Gao, J.; Pi, R.; Zhang, J.; Ye, J.; Zhong, W.; Wang, Y.; Hong, L.; Han, J.; Xu, H.; Li, Z.; et al. G-LLaVA: Solving Geometric Problem with Multi-Modal Large Language Model. In Proceedings of the Thirteenth International Conference on Learning Representations (ICLR), Singapore, 24–28 April 2025.
78. 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 Proceedings of the ICASSP 2025—2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Hyderabad, India, 6–11 April 2025; pp. 1–5. [CrossRef]
79. DeepMind. Gemini Achieves Gold-Medal Level at the International Collegiate Programming Contest World Finals. 2025. Available online: https://deepmind.google/blog/gemini-achieves-gold-medal-level-at-the-international-collegiate-programming-contest-world-finals/ (accessed on 24 March 2026).
80. Hubert, T.; Mehta, R.; Sartran, L.; Horváth, M.Z.; Žužić, G.; Wieser, E.; Huang, A.; Schrittwieser, J.; Schroecker, Y.; Masoom, H.; et al. Olympiad-level formal mathematical reasoning with reinforcement learning. *Nature* **2024**, *651*, 607–613. [CrossRef] [PubMed]
81. 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 Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), San Diego, CA, USA, 2 –7 December 2025.
82. 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 Recognit.* **2026**, *171*, 112261. [CrossRef]
83. Ning, M.; Zhou, Z.; Wang, Q.; Huang, X.; Huang, K. GNS: Solving Plane Geometry Problems by Neural-Symbolic Reasoning with Multi-Modal LLMs. *Proc. AAAI Conf. Artif. Intell. (AAAI)* **2025**, *39*, 24957–24965. [CrossRef]
84. Sharma, A.; Dalmia, A.; Kazemi, M.; Zouaq, A.; Pal, C. GeoCoder: Solving Geometry Problems by Generating Modular Code through Vision-Language Models. In Proceedings of the Findings of the Association for Computational Linguistics (NAACL), Albuquerque, NM, USA, 29 April–4 May 2025; pp. 7340–7356. [CrossRef]