E-GPS: Explainable Geometry Problem Solving via Top-Down Solver and Bottom-Up Generator

Wenjun Wu[1,3], Lingling Zhang[1,3]*, Jun Liu[1,3], Xi Tang[1,3], Yaxian Wang[1,3]
Shaowei Wang[1,3], Qianying Wang[2]

[1]School of Computer Science and Technology, Xi'an Jiaotong University, China [2]Lenovo Research
[3]Key Laboratory of Intelligent Networks and Network Security, Ministry of Education, Xi'an, Shaanxi, China

Abstract

Geometry Problem Solving has drawn growing attention recently due to its application prospects in intelligent education field. However, existing methods are still inadequate to meet the needs of practical application, suffering from the following limitations: 1) explainability is not ensured which is essential in real teaching scenarios; 2) the small scale and incomplete annotation of existing datasets make it hard for model to comprehend geometric knowledge. To tackle the above problems, we propose a novel method called Explainable Geometry Problem Solving (E-GPS). E-GPS first parses the geometric diagram and problem text into unified formal language representations. Then, the answer and explainable reasoning and solving steps are obtained by a Top-Down Problem Solver (TD-PS), which innovatively solves the problem from the target and focuses on what is needed. To alleviate the data issues, a Bottom-Up Problem Generator (BU-PG) is devised to augment the data set with various well-annotated constructed geometry problems. It enables us to train an enhanced theorem predictor with a better grasp of theorem knowledge, which further improves the efficiency of TD-PS. Extensive experiments demonstrate that E-GPS maintains comparable solving performances with fewer steps and provides outstanding explainability.

1. Introduction

Geometry Problem Solving (GPS) aims to obtain the answer of problem based on the given geometric diagram and textual problem description. It has drawn growing attention recently [4, 15, 22, 31] due to its application prospects in intelligent education field in high schools [2, 20]. Different from general question answering (QA) tasks, GPS requires the model to possess the abilities of symbolic abstraction, logical reasoning and algebraic calculation simultaneously [6, 24], making it a challenging task even for large multimodal models (LMMs) like GPT-4V [37]. Therefore, recent works attempt to combine the procedural power of symbolic models with the general power of neural models. Among these, symbolic-based approaches [22, 25, 29, 32] first parse the geometric diagram and problem text into formal language representations, and then continuously predict and apply predefined theorem rules to obtain the final answer. Neural-based approaches [4, 5, 40] tend to transfer the original problem into multi-modal features, and feed them into generative models to acquire an executable program sequence for an answer. However, they both suffer from the following two limitations which hinder their application in practical scenarios.

At methodological level, previous methods are unable to ensure basic explainability which is essential. As shown in Fig. 1, examples of two mainstream GPS methods are listed. The first one is a typical symbolic-based model Inter-GPS

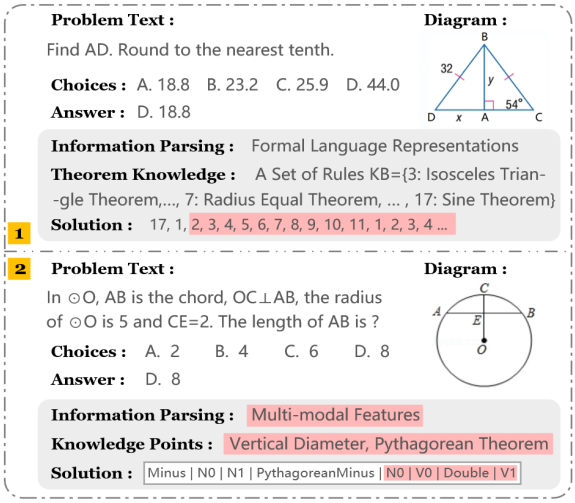

Figure 1. Output examples of two mainstream GPS methods. Case 1 and case 2 are chosen from Inter-GPS [22] and NGS [4], respectively. Content with red background is seen as inexplicable.

*Corresponding author. Email: zhanglling@xjtu.edu.cn

Dataset	Scale	Annotation		
		Solution	Theorem	Parsing
GEOS [32]	186			
GEOS++ [30]	1,406			
GEOS-OS [29]	2,235			
Geometry3K [22]	3,002		✓	✓
GeoQA [4]	5,010	✓		

Table 1. Information of popular GPS datasets. Common basic annotation such as problem text and answer is not listed for brief. Here, *Scale* refers to the number of problem pairs, *Solution* represents the appropriate solving steps to the answer, *Theorem* refers to the explicit theorem knowledge defined in mathematical rules, and *Parsing* denotes problem parsing results in formal languages.

[22] that generates a theorem application sequence. It can be seen that such solution is actually redundant and vague. Many theorems (*e.g.* 7: radius equal theorem) are unhelpful to solving the problem but included in the sequence, which is mostly caused by the high-level reasoning gulf between the known information and final problem target. Moreover, no explicit procedures on specific primitives are given for each theorem application. As for the second neural-based model NGS [4], how the diagram and text are parsed and used for solving the problem is a black box. Furthermore, the generated sequence is unreliable due to the inherent issues of generative models, which makes it even more difficult to conclude precise mathematical theorem knowledge. These drawbacks greatly reduce the interpretability, bringing much confusion to the learners. Therefore, GPS models should generate not only the correct answer, but also the explainable reasoning and solving steps as supporting evidence. It is an essential prerequisite for the employment of solver models in real application scenarios such as teaching assistance.

At data level, the small scale and incomplete annotations of existing datasets make neural models struggle with learning geometric knowledge. Compared with regular QA datasets, the construction of GPS datasets is much more difficult due to the shortage of samples and tremendous labour power demanded for high-standard annotation works. This leads to the limited scale and incomplete annotation of existing GPS datasets, as recorded in Tab. 1. For instance, GEOS [32] and GEOS++ [30] contains only 186 and 1,406 problems, respectively. The scale is expanded by Geometry3K [22] and GeoQA [4], but merely to 3,002 and 5,010 problems with annotations that are still unsatisfactory. The appropriate problem solving sequence is absent in Geometry3K, while the illustration of geometric theorems is not considered in GeoQA. The data issue causes great troubles for training of data-driven models and weakens their mastery of geometric knowledge. It further reduces their performances in tasks like theorem prediction and solution program generation, hence limiting the application potential of GPS models.

To overcome the above two problems, we propose a new method named as *Explainable Geometry Problem Solving* (E-GPS). It first parses the geometric diagram and problem text into unified formal language representations. The problem target (*e.g.* `Find(Line(AB))`) is also specified during parsing. Inspired by how human experts solve geometry problems, starting from the target itself to seek for needed conditions could largely narrow the reasoning gap between the two, compared to starting from the known conditions as previous methods did [22, 25, 32]. That is, we should focus on what is needed instead of what could be used. To this end, the explainable Top-Down Problem Solver (TD-PS) is designed to transfer the task into tree-searching in a top-down manner for solutions with two mechanisms: 1) *target decomposition* decomposes the parent target into multiple child targets according to premise conditions of a theorem rule; 2) *condition processing* is responsible for checking whether needed conditions are acquirable. In this way, all conditions are used for a purpose, which ensures the explainability of final generated reasoning and solving steps. To improve the efficiency of TD-PS, neural-guided theorem prediction is considered to narrow the search space, which also faces the scale and annotation issues of existing datasets. Interestingly, we find that values generated by the forward theorem application of symbolic models could be treated as answers with known solutions. Hence, the Bottom-Up Problem Generator (BU-PG) is devised to continuously apply theorems on the original problem and convert the geometric primitives with newly obtained values into a series of new problem targets. It automatically augments the original data set with a large quantity of well-annotated constructed geometry problems, making it possible for us to train an enhanced theorem predictor for TD-PS.

Our contributions can be summarized as three-folds:

- We introduce a novel top-down problem solver that generates the answer and explainable reasoning and solving steps. To the best of our knowledge, it is the first work that focuses on what is needed for the problem target.
- We propose a bottom-up problem generator that is able to augment the data set with a large quantity of constructed geometry problems with explicit solutions.
- Experiment results demonstrate that our method is able to maintain comparable performances with fewer steps and provide outstanding explainability.

2. Related Works

2.1. Geometry Problem Solving

Developing geometry problem solving systems has been studied for long [11], but mostly on geometry theorem proving [7, 10, 36, 38] and problem formalization [9]. Until re-

cently, research on GPS has been further promoted and can be roughly categorized into two types: symbolic-based and neural-based. Symbolic-based methods continuously update the parsed conditions until the problem is solved. Some work [32] focuses on selecting a most related relation set to solve an optimization problem. Others use extracted theorem knowledge which is represented as horn clause rules [30, 31] or predefined theorem rules [22, 25] to deduce new conditions. There are also works [12, 17] paying attention to the optimization of parsed formal language information. Neural-based methods [4, 5] learn implicit patterns and generate an executable program to obtain a final answer. Some works [18, 40] realize that such generative approach relies heavily on the features extracted from raw problems, and improve the fusion of multi-modal knowledge.

However, these works neglect the high-level reasoning gap between the known conditions and the problem target. It makes their models prone to introduce incorrect or redundant information in the solution, weakening the explainability. Moreover, for neural-based methods, the reasoning from features to executable programs is still a black box.

2.2. Explainable Math Problem Solving

While the explainability of geometry problem solvers has not yet been explored much, there are works investigating that in general math problem solving field. Some prior works generate interpretable solutions of math problems by intermediate structural results [1, 14, 16, 27, 34, 35]. Other works propose to perform logical reasoning based on symbolic language parsed from the problem [23, 28], providing a reference for the establishment of symbolic geometry problem solving systems. Moreover, the ideas of structured image parsing [13, 33, 41] have greatly inspired existing methods [22, 32] on the interpretability of information extraction from geometry problems. However, the explainability of solutions generated by GPS solver models are still in need of further research.

3. Methodology

In a geometry problem (D, T), D is a geometric diagram and T is a textual problem description which includes the problem target (*e.g.* `Find AB?`). With given (D, T), geometry problem solving aims at obtaining the correct answer a^* of target. For more explainable GPS, the corresponding solution steps $\mathcal{S}$ should also be generated. To achieve this goal, we design E-GPS which is illustrated in Fig. 2. Specifically, E-GPS first transfers the raw problem into unified formal language propositions by two parsers in Sec. 3.1. Then the Top-Down Problem Solver is introduced in Sec. 3.2, which generates the answer a^* as well as explainable reasoning and solving steps $\mathcal{S}$. In Sec. 3.3, how the Bottom-Up Problem Generator constructs augmented problems is described in details. The augmented data set is then used for training an enhanced theorem predictor in Sec. 3.4.

3.1. Geometry Problem Parsing

The content integration of geometric diagram D and problem text T is crucial for understanding the entire geometry problem but faces a great modality gulf. Inspired by previous works [22, 25, 32], we utilize a diagram parser and a text parser to transfer the modal-specific knowledge into unified formal language representations automatically.

Diagram Parser. Following previous work [25], the mature neural-based model PGDPNet [39] is utilized to detect the geometric primitives (*i.e.* `line`, `angle` and `arc`) as it has been well solved. The relations between these geometric primitives, symbols and texts in the diagram are further generated as propositions in formal language. We define the proposition set of D as $\mathcal{P}^D$, which includes the relations between geometric primitives and values. For example, `Triangle(ABC)` is a proposition describing the combination of `Line(AB)`, `Line(BC)` and `Line(AC)`. And `Equals(Line(AB),5)` is also a proposition which establishes equation between primitive and value.

Text Parser. For more precise parsing results, we use rule-based text parser with regular expressions [22, 32] to translate the problem text. It automatically parses the content into primitives, values and their relations. Similarly, the text parser also generates a proposition set $\mathcal{P}^T$. Normally, the problem target (*e.g.* `Find(Line(AB))`) is also parsed from problem text, defined as t^*.

Therefore, it is feasible for us to combine the visual and textual information into proposition set $\mathcal{P} = \{\mathcal{P}^D; \mathcal{P}^T\}$. We further integrate offline programs into a predefined theorem knowledge base set $\mathcal{KB} = \{k_1, k_2, \cdots, k_N\}$ the same as in [22], where each conditional rule k_i stands for a geometry theorem and builds equations between primitives and values. For instance, if lengths of two right sides of a right triangle are known, length of its hypotenuse can be calculated through an equation according to the Pythagorean Theorem. Specifically, the proposition set $\mathcal{P}$ can be seen as updated by theorem i once the needed primitives in premise q_i of k_i meet conditions and new proposition p is generated:

$$p \leftarrow k_i\left(q_i, \mathcal{P}\right), k_i \in \mathcal{KB}, \tag{1}$$

$$\mathcal{P} \leftarrow p \vee \mathcal{P}. \tag{2}$$

It is noteworthy that the application of k_i introduces new equations between primitives and values. We encapsulate the above process into the following function:

$$a, \mathcal{P} \leftarrow \texttt{SolveEquations}\left(\mathcal{P}, t, k_i\right). \tag{3}$$

Hence, the above function always performs three tasks: 1) apply theorem k_i if it's given; 2) solve the established equations; 3) calculate a numerical answer a of given target t.

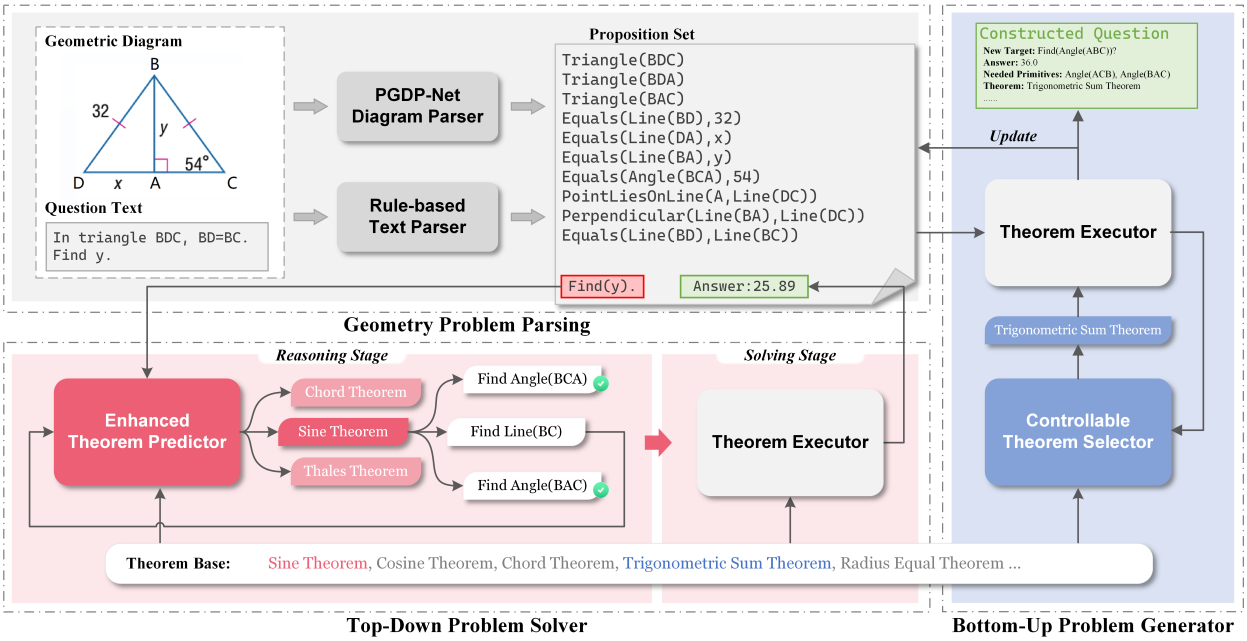

Figure 2. The overall framework of E-GPS which includes three parts: 1) the proposition set in formal language is generated by the diagram parser and text parser; 2) TD-PS generates the answer of the problem target, as well as explainable reasoning and solving steps; 3) BU-PG automatically augments the data set with constructed geometry problems, which is served for training the enhanced theorem predictor.

3.2. Top-Down Problem Solver

Previous works mostly attempted to solve geometry problems by starting from the known information (*e.g.* $\mathcal{P}$ in this work). However, the natural reasoning gap between it and the final target is ignored, which inevitably caused the introduction of redundant or wrong steps. We find that the gap could be narrowed if we start from the target and focus on what is actually needed for solving it rather than what could be used. To achieve this, the explainable **T**op-**D**own **P**roblem **S**olver (TD-PS) first reasons for the solution based on the proposition set $\mathcal{P}$, knowledge base set $\mathcal{KB}$ and target t^* in reasoning stage, and outputs the answer a^* along with explainable reasoning and solving steps $\mathcal{S}$ in solving stage.

3.2.1 Reasoning Stage

Two collaborative mechanisms are implemented to first reason for the solution, namely *target decomposition* and *condition processing*.

Target Decomposition. For geometry problems that require multi-step theorem applications, directly calculating the answer of final target is not ideal. Nevertheless, once the premise condition of a theorem is met, it is most likely that applying this theorem would lead to an answer. Take the geometry problem in Fig. 2 as an example, if the values of `Line(BC)`, `Angle(ACB)` and `Angle(BAC)` are known, obtaining the value of `Line(AB)` is promising according to Sine Theorem. Hence, we transfer the task of finding the answer of t into finding the values for those primitives needed in premise condition of k_i, which we call a decomposition of target. For instance, to solve parent target `Find(Line(AB))`, finding the value of `Line(BC)` is transferred into `Find(Line(BC))` as one of the child targets. These child targets are further packed into a target group $\mathcal{T}_1^{\wedge}$ and they should all be solved for obtaining the answer of parent target. Such decomposition is actually the reverse process of theorem application. Moreover, alternative decompositions are allowed such as target group $\mathcal{T}_2^{\wedge}$ which also leads to a solution according to the same theorem: `(Find(Line(BD)),Find(Angle(ADB)),Find(Angle(BAD)))`$_2^{\wedge}$. Therefore, the process of target decomposition is defined as follows:

$$(\mathcal{T}_1^{\wedge}, \cdots, \mathcal{T}_n^{\wedge})^{\vee} \leftarrow \texttt{TargetDecompose}(t, k_i), \quad (4)$$

where symbol $\wedge$ represents all child targets within the group need to have solutions simultaneously and symbol $\vee$ denotes that at least one target group needs to have a solution. The theorem $k_i (k_i \in K^t)$ is selected by the following function which will be further discussed in Sec. 3.4:

$$K^t \leftarrow \texttt{TheoremPrediction}(\mathcal{P}, \mathcal{KB}, t). \quad (5)$$

Condition Processing. The logical relationship between parent target and child targets needs to be handled carefully to ensure the feasibility of solutions. For example in Fig. 2, `Find(Line(AC))` is solvable by applying Cosine Theorem on `Line(BC)`, `Angle(ACB)`, and `Angle(BAC)`,

where all values of primitives are known in $\mathcal{P}$. We have summarized the processing logic into the following three rules. For parent target t and its decomposed child targets:

- t returns True if it has known numerical answer or is solvable by equations;
- $\mathcal{T}^{\wedge}$ returns True if all child targets in it return True, otherwise returns False;
- t returns True if any target group $\mathcal{T}^{\wedge}$ returns True, otherwise returns False.

In this way, the original task of solving problem target t^* is transferred into tree-searching in a top-down manner, whose goal is to find a path where all values of needed primitives are known in $\mathcal{P}$ or can be obtained by theorem applications. Such process is defined as reasoning. For comprehensive and explicit explanations, child targets $\mathcal{T}^{\wedge}$, theorem k_i and parent target t are recorded as reasoning steps in $\mathcal{S}$:

$$\mathcal{S} \leftarrow \mathcal{S}\texttt{.Append}(\mathcal{T}^{\wedge}, k_i, t). \tag{6}$$

3.2.2 Solving Stage

Once the reasoning steps of problem target t^* is confirmed, the solving orders of theorem application and needed primitives are also determined. Function `SolveEquations()` is then called sequentially by the theorem executor according to solving steps to obtain final answer a^*, which is the solving stage. The whole process of TD-PS is displayed in details in Algorithm 1. We claim that the explainability of TD-PS is reflected in the following aspects:

- TD-PS generates not only the answer, but also appropriate reasoning and solving steps $\mathcal{S}$ without redundancy, which provides human readable solutions;
- Each step in the reasoning and solving stage has specific target and required primitives as supporting evidences;
- Each theorem is defined by rules, which shows clear mathematical procedures of how the theorem is applied;
- The motivation and algorithm design of TD-PS is explainable. It imitates the process of human solving geometry problems to first reason and then solve.

3.3. Bottom-Up Problem Generator

Existing GPS datasets are insufficient in both quantity of samples and annotations of solving steps, posing difficulties for neural models to find patterns of complex theorem applications. To alleviate this issue, the **B**ottom-**U**p **P**roblem **G**enerator (BU-PG) is devised to automatically augment any geometry problem (D, T) parsed in formal languages with a series of constructed geometry problems $\{(D, T_1^c, \mathcal{S}_1^c), (D, T_2^c, \mathcal{S}_2^c), \cdots, (D, T_L^c, \mathcal{S}_L^c)\}$. Each text T_i^c includes a new problem target t_i^c and could be solved by corresponding solving steps $\mathcal{S}_i^c$.

As introduced in Sec. 3.1, $\mathcal{KB}$ is defined as explicit mathematical rules of geometry theorems. Theoretically, continuously applying $\mathcal{KB}$ on $\mathcal{P}$ could acquire many values of unknown primitives. Therefore, given a sequence of theorems, it is possible for the theorem executor to apply theorems one after another and eventually obtain a numerical value of certain primitive. Such primitive is then transferred as a new problem target t_i^c, where the value can be obtained by solution path $\mathcal{S}_i^c$. Similar to TD-PS, BU-PG is able to record explicit primitives and theorem in the solving steps $\mathcal{S}_i^c$, which ensures the comprehensiveness of annotation. For generation of more diverse geometry problems, we define certain terms to control the target type (*e.g.* line and angle), com-

Algorithm 1 Explainable Top-Down Problem Solver

Input: Proposition set $\mathcal{P}$, knowledge base $\mathcal{KB}$, target t^*
Output: Numeric answer a^*, explainable solving steps $\mathcal{S}$

```
function Main(P, KB, t*):
    global KB, P and initialize a* = ∅, S = ∅, time = 0;
    while a* = ∅ and time within restrictions do
        F, S ← TargetDecomposition(t*, S);
        if F is True then
            for k_i, t_i in S do
                a_i, P ← SolveEquations(P, t_i, k_i);
            return a*, S
    return None
function TargetDecomposition(t, S):
    a, P ← SolveEquations(P, t);
    if a ≠ ∅ then
        S ← S.Append(t);
        return True, S
    K^t ← TheoremPrediction(P, KB, t);
    for k_i^t in K^t do
        T^∨ ← TargetDecompose(t, k_i^t);
        F, S, T^∧ ← ConditionProcessing(T^∨, S);
        if F is True then
            S ← S.Append(T^∧, k_i^t, t);
            return True, S
    return False, S
function ConditionProcessing(T, S):
    for T^∧ in T do
        set hasSolution as True;
        for target t in T^∧ do
            F, S ← TargetDecomposition(t, S);
            if F is False then
                S.Backward();
                set hasSolution as False;
                break
        if hasSolution is True then
            return True, S, T^∧
    return False, S, None
```

plexity (*e.g.* multi-target) and difficulty (*e.g.* multi-theorem application) of the constructed geometry problems. To ensure that the solution is reasonable and non-redundant, every latter theorem to be applied should cover at least one primitive value obtained by the former theorem for problems whose difficulty is larger than 1.

With any version of $\mathcal{KB}$ and GPS datasets in formal languages, BU-PG is able to re-generate a new one with a large quantity of samples. Unlike the other augmentation methods such as symbol modification and diagram flip [40], BU-PG directly augments the multi-step theorem application samples for high-level reasoning learning, which shows undeniable value to many downstream tasks. In this work, the augmented data set aug-Geo3K is utilized to train an enhanced theorem predictor, which will be described next.

3.4. Enhanced Theorem Predictor

For symbolic-based models, one of the key challenges is to predict the theorems needed for obtaining the correct answer. A simple implementation is traversing the theorems in $\mathcal{KB}$ until the problem is solved. However, when the scale of $\mathcal{KB}$ becomes large for more complicated geometry problems in the future, such traversal method might obviously reduce the efficiency of the solver model. Inspired by previous works [3, 21, 22], applying neural guided search to predict theorems and speed up the search process of TD-PS is feasible. Given the propositions $\mathcal{P}$, target t and $\mathcal{KB}$, the theorem predictor aims at predicting the needed theorems $K^t = \{k_1^t, \cdots, k_M^t\}$, as introduced in Eq. (5). In practice, a token sequence $E^t = \{e_1, \cdots, e_M\}$ is generated instead, where e_i refers to the index token of k_i^t in $\mathcal{KB}$. The mature transformer-based model [19] is used to accomplish such sequence-to-sequence (Seq2Seq) task. The optimization goal is defined as a negative log-likelihood loss:

$$\mathcal{L}_{M-TP} = -\sum_{i=1}^{M} \log_\theta \left(e_i | e_1, e_2, \cdots, e_{i-1}\right). \quad (7)$$

We also train the model on one-step prediction $(M = 1)$, considering the difficulty of generating a multi-step sequence which involves high-level reasoning and intermediate values. However, sufficient well-annotated samples are required for neural models to learn the logical correlation between propositions and target. In previous work [22], the pseudo-optimal theorem application sequences were constructed as ground-truth by random sampling in $\mathcal{KB}^1$, which introduces redundant theorems. Instead, we utilize our auxiliary data set aug-Geo3K generated by BU-PG to train the theorem predictor. In details, the constructed geometry problems whose difficulty are labeled as 1 and M $(M \geq 1)$ are used as training samples for one-step and multi-step prediction, respectively. In this way, both quantity and quality of samples for theorem prediction training are guaranteed, benefited from the generation mechanism of BU-PG.

4. Experiments

4.1. Datasets

For geometry problem solving task, we mainly conduct experiments on Geometry3K [22]. The theorem prediction is further compared on the augmented data set aug-Geo3K due to the absence of solution annotation in Geometry3K.

Geometry3K: includes 3,002 geometry problems in total and is divided into 2,101 for training, 300 for validation and 601 for testing. Each problem is accompanied with a geometric diagram, a problem text and explicit parsing annotations in formal language. Problems in it cover many geometric shapes, such as lines, triangles, circles, quadrilaterals and other polygons, providing a more realistic benchmark.

aug-Geo3K: is a large scale constructed geometry problem data set augmented from Geometry3K with $\mathcal{KB}^1$, used for auxiliary training and evaluation. It contains a total of 17,607 problems, where each problem is equipped with a diagram, a problem text, the answer and corresponding explicit solutions. The problems are divided into 12,291 for training, 1,492 for validation and 3,824 for testing according to the original data split in Geometry3K.

4.2. Baselines and Evaluation Metrics

Baselines. We mainly compare with the methods on Geometry3K. FiLM [26] is a highly effective model for visual reasoning on abstract images that requires multi-step processing abilities. Two improved models, namely FiLM-BERT [8] and FiLM-BART [19], are also chosen as competitors which use better encoders. Inter-GPS [22] is a popular approach that applies predefined theorem rules until the final problem target is solved. The recent work GeoDRL [25] incorporates deep reinforcement learning into reasoning for better theorem prediction based on a larger theorem set.

Metrics. For geometry problem solving, we evaluate the model performances from two aspects: 1) accuracy: the answer is considered correct if it is the closest to the ground-truth; 2) average steps: the average theorem application steps of solutions. For theorem prediction, we adopt R@K, which shows the percentage of geometry problems whose ground-truth theorem application sequence is contained within the top-K sequences predicted by the model.

4.3. Implementation Details

For fair comparison, the same theorem knowledge base sets $\mathcal{KB}^1$ [22] and $\mathcal{KB}^2$ [25] in previous works are used which include 17 and 24 commonly used geometry theorems, respectively. The solver is set to search for solutions within 5 steps. In our theorem predictor, the transformer model is designed with 6 layers, 12 attention heads and the size of hidden embedding layer is 768. The maximum length of predicted sequence are set to 20. For generation of aug-Geo3K, all 17 theorems in $\mathcal{KB}^1$ are considered and the diffi-

Methods	Question Type				Geometric Shape					Accuracy	Steps
	Measure	Length	Area	Ratio	Line	Triangle	Quad	Circle	Other		
Human	53.7	59.3	57.7	42.9	46.7	53.8	68.7	61.7	58.3	56.9	–
Human Expert	89.9	92.0	93.9	66.7	95.9	92.2	90.5	89.9	92.3	90.9	–
FiLM [26]	28.7	32.7	39.6	33.3	33.3	29.2	33.6	30.8	29.6	31.7	–
FiLM-BERT [8]	32.9	33.3	30.2	25.0	32.1	32.3	32.2	34.3	33.3	32.8	–
FiLM-BART [19]	32.1	33.0	35.8	50.0	34.6	32.6	37.1	30.1	37.0	33.0	–
Inter-GPS [22]	59.1	61.7	30.2	50.0	59.3	66.0	52.4	45.5	48.1	57.5	–
Inter-GPS (GT)	83.1	77.9	62.3	75.0	86.4	83.3	77.6	61.5	70.4	78.3	7.10
E-GPS (ours)	76.8	62.6	24.5	75.0	72.8	73.0	55.7	51.5	41.9	64.7	3.49∼4.19
E-GPS (GT)	83.8	80.0	66.7	63.9	87.7	85.3	79.2	65.9	56.8	79.8	3.35∼3.99
GeoDRL* [25]	75.5	70.5	22.6	83.3	77.8	76.0	62.9	53.8	48.1	68.4	–
GeoDRL* (GT)	86.5	**93.7**	**75.5**	100.0	87.7	93.1	**90.2**	78.3	**77.8**	89.4	2.34
E-GPS* (ours)	78.3	67.2	27.7	72.2	76.1	75.6	59.4	55.0	51.8	67.9	1.63∼2.28
E-GPS* (GT)	**90.4**	92.2	73.6	**100.0**	**91.4**	**93.1**	87.9	**81.1**	75.3	**89.8**	**1.57∼2.18**

Table 2. Comparison of geometry problem solving on Geometry3K. Methods with * mark use the larger theorem base set $\mathcal{KB}^2$ [25], while the others use theorem base set $\mathcal{KB}^1$ [22]. GT refers to using ground-truth parsing results. Best results are in bold.

Methods	One-Step			Multi-Step			rSum
	R@1	R@3	R@5	R@1	R@3	R@5	
Inter-GPS	22.1	41.8	59.5	11.8	22.3	31.7	189.2
E-GPS	**61.3**	**82.8**	**88.9**	**49.2**	**72.5**	**80.3**	**435.0**

Table 3. Comparison of theorem prediction on aug-Geo3K. Best results are in bold.

culty is set to 1 and 5 for one-step and multi-step prediction, respectively. The predictor is trained by Adam optimizer with a 0.01 learning rate and 10 training epochs. Experiments are repeated three times for more confident results.

4.4. Performance Comparison

We first compare the model performances for the regular geometry problem solving task, and then conduct extra experiments on theorem prediction.

Geometry Problem Solving. The overall accuracy, detailed accuracy rates and average steps are recorded in Tab. 2. For fair comparison, a numerical interval is recorded for E-GPS where the actual average steps lies in, in consistency with baselines [22, 25]. The first and second number represent the average used theorems and the average solving steps, respectively. We make the following observations:

- E-GPS shows comparable performances with the SOTA solvers and significantly shortens the steps. Besides the 1.5% improvement of accuracy, E-GPS reduces the steps from 7.10 in Inter-GPS to 3.35 at minimum, using $\mathcal{KB}^1$. Compared to GeoDRL with $\mathcal{KB}^2$, the steps are further reduced from 2.34 to 1.57 at minimum with a slight gain of 0.4% on accuracy. The results demonstrate that E-GPS is a more effective geometry problem solving method.
- The shortening of steps indicates the superiority of E-GPS generating better solutions. It mainly benefits from the mechanism of E-GPS which ensures the necessity of each step for better explainability, differed from previous methods. Moreover, obtaining more simple solutions would also lead to fewer average steps.
- The performance gain of GeoDRL is largely due to the introduction of the larger theorem base. Using $\mathcal{KB}^2$, E-GPS also achieves comparable performance, just as comparing with Inter-GPS based on $\mathcal{KB}^1$. Further, E-GPS is designed to be easily integrated with future version of $\mathcal{KB}$ without retraining, while it costs greater for the reinforcement learning method in GeoDRL to adjust to a new $\mathcal{KB}$.

Theorem Prediction. It is a crucial step which reflects the model's mastery of geometry theorem knowledge but remains challenging. For fair comparison, we choose baseline Inter-GPS [22] which uses the same theorem predictor settings as ours. According to Tab. 3, our theorem predictor trained with the help of BU-PG shows excellent performances with the best results over all evaluation metrics. On one-step prediction, E-GPS achieves more than 80% at both R@3 and R@5, obtaining a margin of 41.0% and 29.4% improvement compared to Inter-GPS. On the more challenging multi-step prediction which involves high-level reasoning, the performance gap is further widened . The results demonstrate the effectiveness of our BU-PG for augmenting the original data set, which better supports the neural models to learn the application of geometric knowledge.

4.5. Ablation Studies and Discussion

Theorem Selection Strategies. To evaluate the efficiency of using theorem prediction during reasoning, we choose some basic selection strategies for comparison in Tab. 4. It shows that our enhanced theorem predictors trained by one-

Strategy	Accuracy	Steps	
		Application	Search
Traversal	77.0	3.49∼4.00	22.3
Random	78.1	3.62∼4.12	16.1
Predict (M-step)	79.6	**3.33∼3.96**	12.0
Predict (1-step)	**79.8**	3.35∼3.99	**11.5**

Table 4. Comparison of different theorem selection strategies in E-GPS on Geometry3K with $\mathcal{KB}^1$. Best results are in bold.

Methods	Explainability			
	Reasoning	Solving	Theorem	Model
Inter-GPS			✓	✓
NGS		✓		
GeoDRL		✓	✓	✓
E-GPS	✓	✓	✓	✓

Table 5. Comparison of explainability with geometry solvers.

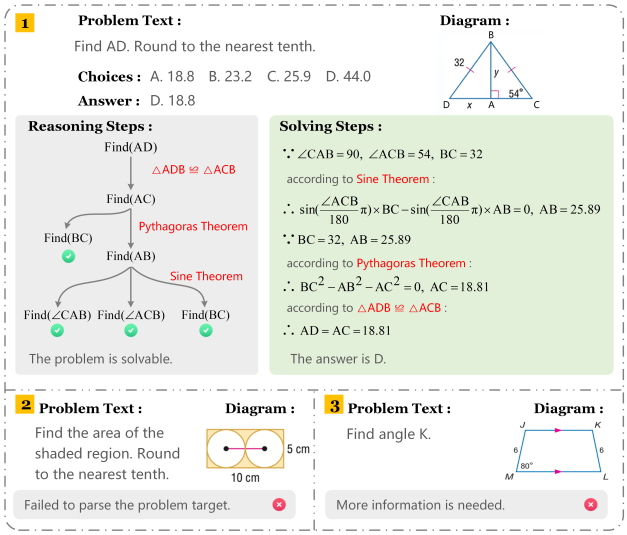

Figure 3. Some typical cases from E-GPS.

step and multi-step prediction both acquire a higher accuracy with obviously fewer steps. Compared to the traversal strategy, our predictor (1-step) even shortens the search steps by nearly half, with a drop of 10.8 steps. Based on previous experiments in Tab. 3, these results further demonstrate that our enhanced theorem predictor trained with BU-PG has a better grasp of geometric knowledge and hence largely improves the efficiency of the solver by appropriate theorem prediction.

Explainability. We further compare the explainability with some baselines in Tab. 5. It can be seen that E-GPS is the only method that is thoroughly explainable from all aspects. It is noteworthy that although GeoDRL generates human readable solutions after post processing, it is unable to avoid the introduction of redundant steps from the early stage of reasoning the same as Inter-GPS, which is caused by the method itself. In contrast, E-GPS ensures the necessity of each step during reasoning and thereby provides superior explainability of both reasoning and solving steps.

Case Analysis. We select some representative results for further analysis. Take case 1 in Fig. 3 as an example, E-GPS is able to generate the correct answer, as well as detailed reasoning and solving steps. The appropriate theorem application steps and specific primitives are confirmed during top-down reasoning, and seen as guidance for obtaining the final answer during bottom-up solving. Nevertheless, there are also limitations. While E-GPS chooses the Pythagoras Theorem and Sine Theorem to acquire the length of AC, it can be solved simply by Cosine Theorem. It is not promising for E-GPS to generate the shortest solution, which is achievable but would take longer time. Moreover, its performance is also affected by the parsing results, which is common in symbolic-based models. It failed to reason from the target in case 2 because the parser is confused about the shaded region. In case 3, E-GPS is unable to give the final solution due to the absent information specifying the region as a trapezoidal or a parallelogram.

5. Conclusion

In this paper, we present the novel method E-GPS, which includes a top-down solver and a bottom-up generator. The solver starts from the problem target to reason for solutions, and generates the final answer with solving steps. The generator uses predefined theorem rules to augment the data set with constructed geometry problems, providing sufficient well-annotated training samples. Experiments demonstrate that E-GPS maintains comparable performances with fewer steps and shows superior explainability. In the future, we will be looking into the way of obtaining the shortest solution of a problem, and constructing a larger theorem rule base for further performance improvement.

Acknowledgments

This work was supported by National Key Research and Development Program of China (2022YFC3303600), National Natural Science Foundation of China (62137002, 62293550, 62293553, 62250009, 62250066, 62176209 and 62106190), "LENOVO-XJTU" Intelligent Industry Joint Laboratory Project, Natural Science Basic Research Program of Shaanxi (2023-JC-YB-593), the Youth Innovation Team of Shaanxi Universities, Shaanxi Undergraduate and Higher Education Teaching Reform Research Program (Program No.23BY195), Project of China Knowledge Centre for Engineering Science and Technology.

References

[1] Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*, 2019. 3

[2] Richa Bajaj and Vidushi Sharma. Smart education with artificial intelligence based determination of learning styles. *Procedia computer science*, 132:834–842, 2018. 1

[3] Mislav Balunovic, Pavol Bielik, and Martin Vechev. Learning to solve smt formulas. *Advances in Neural Information Processing Systems*, 31, 2018. 6

[4] Jiaqi Chen, Jianheng Tang, Jinghui Qin, Xiaodan Liang, Lingbo Liu, Eric P Xing, and Liang Lin. Geoqa: A geometric question answering benchmark towards multimodal numerical reasoning. *arXiv preprint arXiv:2105.14517*, 2021. 1, 2, 3

[5] Jiaqi Chen, Tong Li, Jinghui Qin, Pan Lu, Liang Lin, Chongyu Chen, and Xiaodan Liang. Unigeo: Unifying geometry logical reasoning via reformulating mathematical expression. *arXiv preprint arXiv:2212.02746*, 2022. 1, 3

[6] Mohan Chinnappan. Schemas and mental models in geometry problem solving. *Educational Studies in Mathematics*, 36:201–217, 1998. 1

[7] Shang-Ching Chou, Xiao-Shan Gao, and Jing-Zhong Zhang. Automated generation of readable proofs with geometric invariants: I. multiple and shortest proof generation. *Journal of Automated Reasoning*, 17(3):325–347, 1996. 2

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 6, 7

[9] Wenbin Gan and Xinguo Yu. Automatic understanding and formalization of natural language geometry problems using syntax-semantics models. *International Journal of Innovative Computing, Information and Control*, 14(1):83–98, 2018. 2

[10] Wenbin Gan, Xinguo Yu, Ting Zhang, and Mingshu Wang. Automatically proving plane geometry theorems stated by text and diagram. *International Journal of Pattern Recognition and Artificial Intelligence*, 33(07):1940003, 2019. 2

[11] Herbert Gelernter, James R Hansen, and Donald W Loveland. Empirical explorations of the geometry theorem machine. In *Papers presented at the May 3-5, 1960, western joint IRE-AIEE-ACM computer conference*, pages 143–149, 1960. 2

[12] Fucheng Guo and Pengpeng Jian. A graph convolutional network feature learning framework for interpretable geometry problem solving. In *2022 International Conference on Intelligent Education and Intelligent Research (IEIR)*, pages 59–64. IEEE, 2022. 3

[13] Feng Han and Song-Chun Zhu. Bottom-up/top-down image parsing by attribute graph grammar. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, pages 1778–1785. IEEE, 2005. 3

[14] Yining Hong, Qing Li, Daniel Ciao, Siyuan Huang, and Song-Chun Zhu. Learning by fixing: Solving math word problems with weak supervision. In *Proceedings of the AAAI conference on artificial intelligence*, pages 4959–4967, 2021. 3

[15] Mark Hopkins, Ronan Le Bras, Cristian Petrescu-Prahova, Gabriel Stanovsky, Hannaneh Hajishirzi, and Rik Koncel-Kedziorski. Semeval-2019 task 10: math question answering. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 893–899, 2019. 1

[16] Danqing Huang, Shuming Shi, Chin-Yew Lin, and Jian Yin. Learning fine-grained expressions to solve math word problems. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 805–814, 2017. 3

[17] Pengpeng Jian, Fucheng Guo, Cong Pan, Yanli Wang, Yangrui Yang, and Yang Li. Interpretable geometry problem solving using improved retinanet and graph convolutional network. 2023. 3

[18] Pengpeng Jian, Fucheng Guo, Yanli Wang, and Yang Li. Solving geometry problems via feature learning and contrastive learning of multimodal data. *CMES-Computer Modeling in Engineering & Sciences*, 136(2), 2023. 3

[19] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 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*, pages 7871–7880, 2020. 6, 7

[20] Jinjiao Lin, Haitao Pu, Yibin Li, and Jian Lian. Intelligent recommendation system for course selection in smart education. *Procedia Computer Science*, 129:449–453, 2018. 1

[21] Sarah Loos, Geoffrey Irving, Christian Szegedy, and Cezary Kaliszyk. Deep network guided proof search. *arXiv preprint arXiv:1701.06972*, 2017. 6

[22] Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning. *arXiv preprint arXiv:2105.04165*, 2021. 1, 2, 3, 6, 7

[23] Takuya Matsuzaki, Takumi Ito, Hidenao Iwane, Hirokazu Anai, and Noriko H Arai. Semantic parsing of pre-university math problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2131–2141, 2017. 3

[24] Andi Saparuddin Nur and Evy Nurvitasari. Geometry skill analysis in problem solving reviewed from the difference of cognitive style students junior high school. *Journal of Educational Science and Technology*, 3(3):204–210, 2017. 1

[25] Shuai Peng, Di Fu, Yijun Liang, Liangcai Gao, and Zhi Tang. Geodrl: A self-learning framework for geometry problem solving using reinforcement learning in deductive reasoning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13468–13480, 2023. 1, 2, 3, 6, 7

[26] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, 2018. 6, 7

[27] Jinghui Qin, Lihui Lin, Xiaodan Liang, Rumin Zhang, and Liang Lin. Semantically-aligned universal tree-structured solver for math word problems. *arXiv preprint arXiv:2010.06823*, 2020. 3

[28] Subhro Roy and Dan Roth. Mapping to declarative knowledge for word problem solving. *Transactions of the Association for Computational Linguistics*, 6:159–172, 2018. 3

[29] Mrinmaya Sachan and Eric Xing. 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)*, pages 251–261, 2017. 1, 2

[30] Mrinmaya Sachan, Kumar Dubey, and Eric Xing. 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*, pages 773–784, 2017. 2, 3

[31] Mrinmaya Sachan, Avinava Dubey, Eduard H Hovy, Tom M Mitchell, Dan Roth, and Eric P Xing. Discourse in multimedia: A case study in extracting geometry knowledge from textbooks. *Computational Linguistics*, 45(4):627–665, 2020. 1, 3

[32] Minjoon Seo, Hannaneh Hajishirzi, Ali Farhadi, Oren Etzioni, and Clint Malcolm. Solving geometry problems: Combining text and diagram interpretation. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1466–1476, 2015. 1, 2, 3

[33] Kewei Tu, Meng Meng, Mun Wai Lee, Tae Eun Choe, and Song-Chun Zhu. Joint video and text parsing for understanding events and answering queries. *IEEE MultiMedia*, 21(2): 42–70, 2014. 3

[34] Lei Wang, Yan Wang, Deng Cai, Dongxiang Zhang, and Xiaojiang Liu. Translating a math word problem to an expression tree. *arXiv preprint arXiv:1811.05632*, 2018. 3

[35] Lei Wang, Dongxiang Zhang, Jipeng Zhang, Xing Xu, Lianli Gao, Bing Tian Dai, and Heng Tao Shen. Template-based math word problem solvers with recursive neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7144–7151, 2019. 3

[36] Wu Wen-Tsun. Basic principles of mechanical theorem proving in elementary geometries. *Journal of automated Reasoning*, 2:221–252, 1986. 2

[37] Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. The dawn of lmms: Preliminary explorations with gpt-4v (ision). *arXiv preprint arXiv:2309.17421*, 9, 2023. 1

[38] Xinguo Yu, Mingshu Wang, Wenbin Gan, Bin He, and Nan Ye. A framework for solving explicit arithmetic word problems and proving plane geometry theorems. *International Journal of Pattern Recognition and Artificial Intelligence*, 33 (07):1940005, 2019. 2

[39] Ming-Liang Zhang, Fei Yin, Yi-Han Hao, and Cheng-Lin Liu. Plane geometry diagram parsing. *arXiv preprint arXiv:2205.09363*, 2022. 3

[40] Ming-Liang Zhang, Fei Yin, and Cheng-Lin Liu. A multimodal neural geometric solver with textual clauses parsed from diagram. *arXiv preprint arXiv:2302.11097*, 2023. 1, 3, 6

[41] Song-Chun Zhu, David Mumford, et al. A stochastic grammar of images. *Foundations and Trends® in Computer Graphics and Vision*, 2(4):259–362, 2007. 3