




Article

FGeo-Parser: Autoformalization and Solution of Plane Geometric Problems

Na Zhu ^{1,2}, Xiaokai Zhang ², QiKe Huang ¹, Fangzhen Zhu ², Zhenbing Zeng ³ and Tuo Leng ^{1,2,*}

¹ Institute of Artificial Intelligence, Shanghai University, Shanghai 200444, China; nazhu@shu.edu.cn (N.Z.); qkhuang112@shu.edu.cn (Q.H.)

² School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China; xiaokaizhang@shu.edu.cn (X.Z.); zhufz@shu.edu.cn (F.Z.)

³ College of Sciences, Shanghai University, Shanghai 200444, China; zbzeng@shu.edu.cn

* Correspondence: tleng@shu.edu.cn

Abstract: Automatic geometric problem-solving is an active and challenging subfield at the intersection of AI and mathematics, where geometric problem parsing plays a critical role. It involves converting geometric diagram and text into certain formal language. Due to the complexity of geometric shapes and the diversity of geometric relationships, geometric problem parsing demands that the parser exhibit cross-modal comprehension and reasoning capabilities. In this paper, we propose an enhanced geometric problem parsing method called FGeo-Parser, which converts problem diagrams and text into the formal language of the FormalGeo. It also supports reverse formalization to generate human-like solutions, reflecting the symmetry between parsing and generating. Specifically, diagram parser leverages the BLIP to generate the construction CDL and image CDL, while text parser employs the T5 to produce the text CDL and goal CDL where these neural networks are both based on a symmetric encoder–decoder architecture. With the assistance of a theorem predictor, these CDLs were automatically parsed and step-by-step reasoning was executed within FGPS. Finally, the reasoning process was input into a solution generator, which subsequently produced a human-like solution process. Additionally, we re-annotated problem diagrams and text based on the FormalGeo7K dataset. The formalization experiments on the new dataset achieved a match accuracy of 91.51% and a perfect accuracy of 56.47%, while the combination with the theorem predictor achieved a problem-solving accuracy of 63.45%.



Academic Editor: Zhixun Su

Received: 22 November 2024

Revised: 17 December 2024

Accepted: 19 December 2024

Published: 24 December 2024

Citation: Zhu, N.; Zhang, X.; Huang, Q.; Zhu, F.; Zeng, Z.; Leng, T.

FGeo-Parser: Autoformalization and Solution of Plane Geometric Problems. *Symmetry* **2025**, *17*, 8. <https://doi.org/10.3390/sym17010008>

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: geometric problem autoformalization; geometry problem-solving; FormalGeo7K dataset; cross-modal

1. Introduction

Geometric problem-solving requires a system capable of multimodal fusion, diagrammatic reasoning, relational operations, logical reasoning, and algebraic computation, making it a long-standing challenge [1–3] that has drawn significant attention from both the artificial intelligence and mathematics communities. Geometric problem parsing is a critical first step in geometric problem-solving. It transforms the problem description and geometric diagrams into a formal representation that machines can understand and process, forming the foundation for subsequent problem-solving. A typical geometric problem consists of a geometric diagram and a textual problem description. A geometric problem-solving system must first parse the known conditions and the goals from both

the textual description and the diagram to enable subsequent reasoning and computation, as shown in Figure 1.

Traditional methods, such as search-based methods [4,5], algebraic methods [6–8], and point elimination methods [9], require manual preliminary formalization of geometric problems, with the solving process involving numerous manually defined rules. The development of deep learning technologies has introduced new avenues for geometric problem-solving. Recent works predominantly construct neuro-symbolic systems to address these challenges, which can be categorized into Deductive Database methods [10–12] and Program Sequence Generation methods [13–15]. Deductive Database methods parse the problem’s textual description and diagram into a formal language and subsequently solve the problem through logical reasoning. In contrast, Program Sequence Generation methods encode the problem text and diagram to create a joint problem representation, input this encoding into a decoder, and generate a program sequence, which is then executed by a program executor.

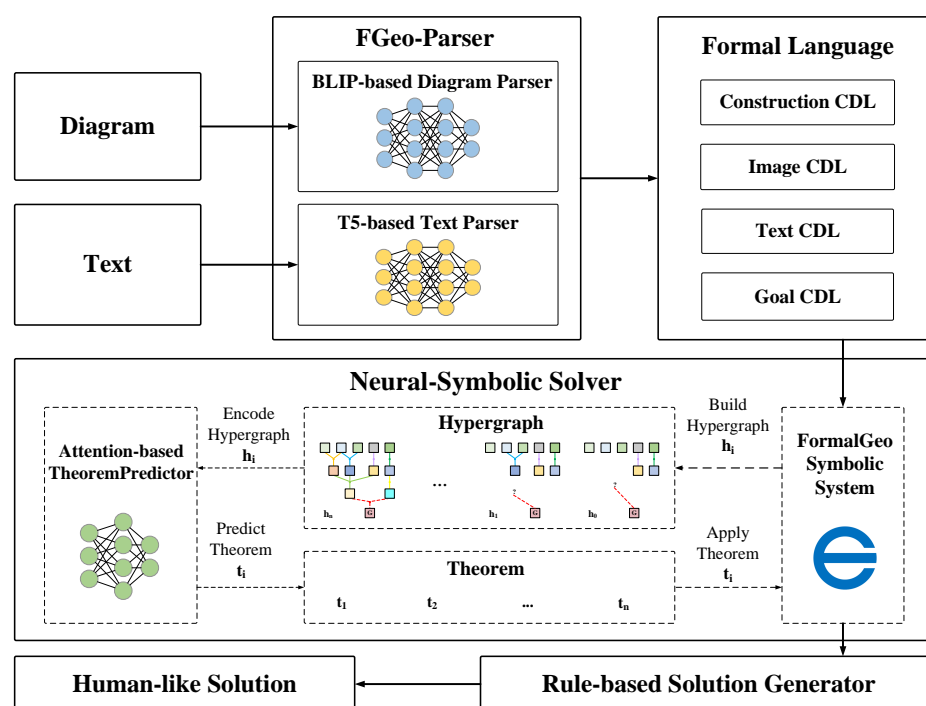


Figure 1. The architecture of automated geometry problem-solving on FGPS with FGeo-Parser and Hyper-GNet.

However, existing methods have limitations that hinder the problem-solving success rate of neuro-symbolic systems and the difficulty of geometric problems they can address. The first issue is that small-scale datasets and incomplete annotations prevent data-driven neural networks from fully acquiring geometric knowledge, making it difficult to generalize to problems outside the dataset and to higher difficulty problems, such as those at the IMO level. To address the first issue, we refined the FormalGeo7K [16] dataset by correcting annotation errors, standardizing the description style of geometric problems, and supplementing missing English and Chinese problem descriptions, resulting in the FormalGeo7K-v2 dataset. We also used GeoGebra to recreate the geometric problem diagrams, exporting them in both PNG and GGB formats. Furthermore, we designed a data augmentation method tailored to the principles of FormalGeo, which doubles problem samples and expands the training set to 14,700 samples, and the problem text is rewritten using LLMs, applying a similar twofold augmentation. Importantly, we adopted an online

data augmentation approach, ensuring that the training set generated in each epoch is different, thereby greatly increasing the dataset's diversity.

The second issue is that existing methods focus on applying deep learning to geometric problem-solving but neglect problem representation. Most studies adopt symbolic systems from NGS [13] or Inter-GPS [10]. NGS [13] uses a multimodal model to predict solutions but cannot produce human-readable ones and has limited representation. Inter-GPS [10] has a formal system yet has difficulty expanding and lacks error-checking and backtracking. To address this, we developed FormalGeo, a consistent geometry formal system based on formalization theory [17]. It defines the system with formal language, separating definitions from solver code for the easy addition of predicates and theorems. It converts texts and diagrams into a unified formal language like first-order predicate logic for good readability and computer processing. For geometric problem-solving aspects, it integrates them within a predicate logic framework. The solving process forms a hypergraph, which can be reverse-formalized into a human-like solution, reflecting semantic–syntactic and bidirectional mapping symmetries.

The third issue is that existing methods, such as FGeo-DRL [11] and FGeo-TP [18], rely on manual annotations to convert geometric problems into formal descriptions. Subsequently, geometric problem-solvers take these formal descriptions as input and output a formalized solution process. Although these methods achieve relatively high problem-solving success rates on the FormalGeo7K dataset, the formalization process requires substantial human effort, highlighting the need for research into geometric problem auto-formalization. To address the third issue, we developed FGeo-Parser, an automatic parser for geometric problems. Starting with problem diagrams and text as input, the diagram parser outputs the construction CDL and image CDL, while the text parser outputs the text CDL and goal CDL. Together, these components form a geometric problem represented in formal language. The diagram parser is implemented based on the BLIP model, while the text parser is implemented using the T5 model. These neural networks are both based on a symmetric encoder–decoder architecture. With the assistance of a predictor, FGPS solves the problem, and the rule-based generator converts the solution process into human-like solutions. The overall architecture is illustrated in Figure 1.

Our contributions are as follows:

1. We enhanced the FormalGeo7K dataset to create FormalGeo7K-v2, expanding the problem count to 7000, standardizing the problem presentation style, and correcting several annotation errors. Additionally, we used GeoGebra to relabel high-quality geometric diagrams that include detailed positional information for geometric elements such as points and lines, making them suitable for various types of diagram parsing tasks.
2. We built FGeo-Parser, an automatic parser for geometric problems, which can automatically convert a geometric diagram and text into a unified formal statement. It is divided into the diagram parser and the text parser. The diagram parser, based on the BLIP model, converts the problem diagram into construction CDL and image CDL, while the text parser, built on the T5 model, transforms problem text into text CDL and goal CDL. These CDLs provide a comprehensive formal representation of a geometric problem.
3. We developed a comprehensive framework for automatic geometric problem-solving. Given the input of raw geometric diagram and text, the framework produces a human-readable and interpretable solution process. Specifically, FGeo-Parser is employed to convert the geometric problem into formal language, followed by theorem sequence prediction using HyperGNet within FGPS. Finally, a rule-based solution generator is designed to output a human-like solution process.

4. We conducted comparative and ablation experiments on the FormalGeo7K-v2 dataset for two tasks: geometric problem parsing and geometric problem-solving. FGeo-Parser achieved a match accuracy of 91.51% and a perfect score of 56.47%. The neuro-symbolic solver obtained an overall problem-solving success rate of 63.45%. The experimental results demonstrate the effectiveness of the FGeo-Parser and neuro-symbolic solver proposed in this paper.

2. Related Works

2.1. Traditional Methods

The Geometry Theorem Prover [4] uses a backward search via the following steps: Start from problem goals, then decompose into sub-goals by rules until only known conditions are left. Vice versa, the forward chaining method [5] starts from knowns and expands till the goal is reached. Deductive Database methods [19], a data-based search type, boosts forward chaining via a structured database for efficiency. However, these search-based methods demand a substantial degree of manual rule and problem description definition; they are also time-consuming and may restrict scalability and applicability for various geometric problems.

The Wu method [6] turns geometric problems into an equation system with algebraic equalities and inequalities, making the solution similar to that of algebraic equations. Its appearance has propelled geometric problem-solving research, spawning methods like Gröbner bases [7], numerical parallel [20], polynomial system triangulation elimination [21], and dimensionality reduction [22]. Essentially, algebraic methods use computer power. But they cannot produce human-readable proof processes.

The introduction of point elimination methods [9] based on geometric invariants has advanced research into mechanical solutions for geometric problems. This method uses a constructive language to describe geometric problems, with the solution process embedded in the construction steps. Point elimination methods can produce human-like problem-solving processes and have facilitated the development of practical tools such as Geometry Explorer [23] and Java Geometry Expert [24]. The principles of point elimination methods can be readily extended to solid geometry [25] and non-Euclidean geometry [26]. However, these methods require substantial effort to discover new invariants and define point elimination rules, which limits the types of geometric problems they can represent and solve.

Since the 21st century, with tech and theory advances, some scholars aimed to build systems for auto-parsing and solving geometric problems. The GEOS [27] models the understanding task as optimization, creating a system combining text and diagram interpretation. It parses text and diagram into logical expressions and solves by checking satisfiability. GeoShader [28] focuses on shaded area problems, building an analysis hypergraph. Besides problem-solving, theorem proving [29,30], formalization [31,32], and knowledge extraction [33] have also received significant attention. But these methods rely on manual rules and human annotations, limiting problem types and generalization.

2.2. Modern Methods

With the advancement of deep learning technologies, several data-driven automatic problem-solving systems capable of learning from examples have been proposed. Recent works mostly constructed neuro-symbolic systems to solve geometric problems, which can be divided into Deductive Database methods [10–12] and Program Sequence Generation methods [13–15].

Deductive Database methods transform problem text and diagrams into formal language for logical reasoning. Inter-GPS [10] automatically converts them into formal lan-

guage via text parsing and object detection. A symbolic solver reasons stepwise with a theorem predictor. GeoDRL [34] is a self-learning framework integrating logic graph deduction and deep reinforcement learning for geometry reasoning as a Markov decision process. AlphaGeometry [12] uses a neural language model trained on synthetic data to guide a symbolic deduction engine in IMO geometry problems. FGeo-TP [18] employs a language model to predict theorem sequence, improving heuristic search. FGeoDRL [11] combines deep reinforcement learning with the FormalGeo system to obtain human-readable solutions for geometric problems.

Program Sequence Generation methods encode problem text and diagrams for joint encoding, and then input them into a decoder to generate a program sequence executed by a program executor. NGS [13] parses a multimodal geometry problem's info and generates interpretable programs with self-supervised tasks. S2G [35] maps the solving process to interpretable operation trees. Geoformer [36] tackles calculation and proving via sequence generation. DPE [37] encodes long/medium text. A symbolic character-aware model [38] tokenizes symbolic characters for better understanding. Dual-GeoSolver [15] simulates human dual-reasoning. PGPSNet [14] combines pre-training, augmentation, and decoding for geometric reasoning. GOLD [39] processes symbols and primitives to extract relations, converting them to natural language for LLM-based solving. Zeng et al. [40] developed software using LSTM and ResNet101 for encoding and self-attention for multimodal fusion and knowledge extraction.

In addition to geometric problem-solving tasks, other related geometric problems have also garnered the attention of researchers. PGDPNet [41] is an end-to-end model for geometry diagram parsing. It consists of two components: an instance segmentation model to extract geometric primitives, and a graph neural network to parse geometric relationships and classify primitives. Murphy et al. [42] introduced a neuro-symbolic framework for autoformalizing Euclidean geometry, which combines domain knowledge, SMT solvers, and large language models. Zhang et al. [43] evaluated the performance of 10 large language models and multimodal models in solving geometric problems.

3. FormalGeo7K Datasets

Several datasets of geometry problems have emerged, including Geometry3K [10], GeoQA [13], PGDP5K [44], etc. However, there are still some notable issues: (1) limited scale or absence of annotations for certain modalities; (2) low annotation quality, which hampers the training of neural networks; and (3) a lack of formal theoretical support. To address these challenges, we built a large-scale plane geometry problem dataset, FormalGeo7K, which contains 7000 plane geometry problems, covering difficulty levels from grades 6 to 12. Each problem is accompanied by fine-grained diagrams, natural language texts, formal language annotations, and theorem sequence annotations based on the FormalGeo [45].

3.1. Collection and Description

The FormalGeo7K dataset is sourced from the Geometry3K [10] and GeoQA+ [37] datasets. After removing non-planar geometry and repeated samples, we collected 2848 problems from Geometry3K and 4152 problems from GeoQA+. These problems are exercises drawn from popular textbooks from grades 6 to 12, covering almost problem types in plane geometry.

Unlike most existing methods that only address multiple-choice questions requiring candidate answers, our approach can solve numerical and relational problems to provide a problem-solving process. Given an initial problem, it can be divided into four elements, all in human-readable form: problem text t , problem diagram d , solution goal c , and problem answer a . To support the mechanization of geometry-solving, we developed a

self-consistent and expandable formal system, FormalGeo [45], which includes 88 predicates and 196 geometric theorems, all in machine-readable form. The problem description needs to be transformed into conditional declaration language CDL. It includes construction CDL cdl_c , image CDL cdl_i , text CDL cdl_t , and goal CDL cdl_g , where cdl_c is composed of the predicates Shape, Collinear, and Cocircular, representing the construction information of geometric elements and uniquely constructing the geometric diagram. cdl_i represents the conditions depicted in the problem diagram; cdl_t reflects the conditions expressed in the problem's text; and cdl_g defines the problem-solving goal. The problem-solving process is represented as theorem sequences ss , where the goal is gradually derived from the premises. The entire process is fully interpretable. Thus, a problem can be represented as $(t, d, cdl_c, cdl_i, cdl_t, cdl_g, a, s)$. A sample of the dataset can be seen in Figure 2 and our formal system is described in detail in FormalGeo [17].

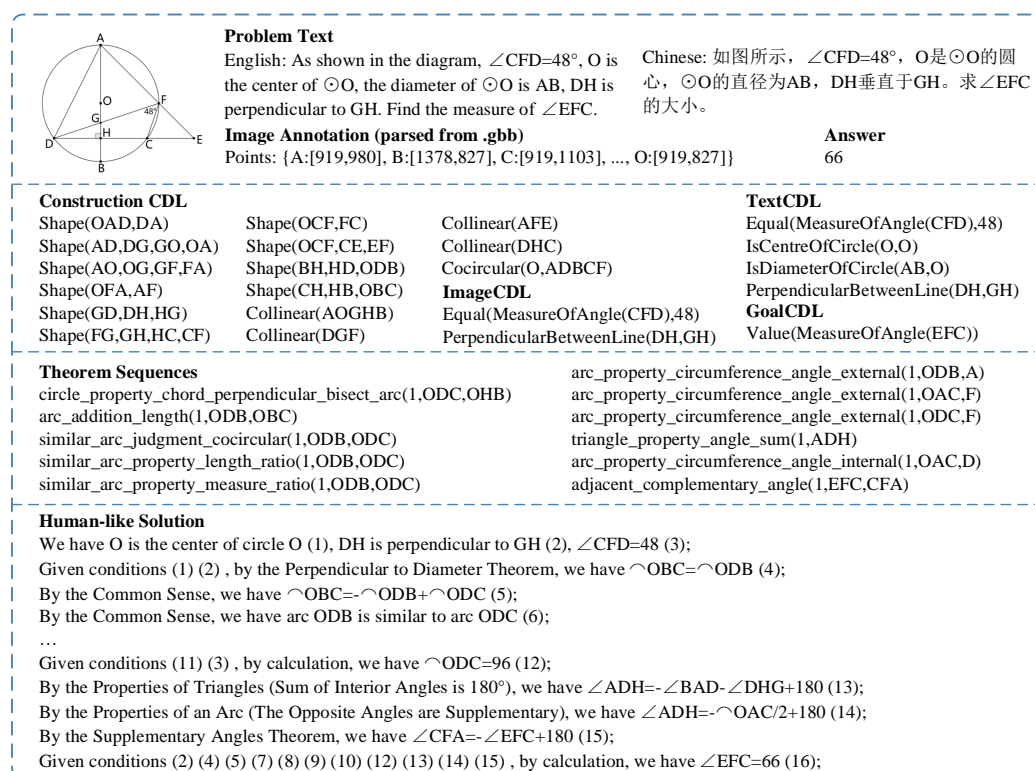


Figure 2. An example of annotated geometric problem in FormalGeo7K.

3.2. Annotation

The annotation of the FormalGeo7K dataset involves two phases. The first phase entails converting 6981 initial problems into our formal language, with further details outlined in [16]. Due to issues such as missing diagram elements and disorganized text formats in the collected initial problems, we conducted a second annotation phase. This phase involved re-annotating the corresponding problem diagram and text based on the formal language annotations. Additionally, we collected problems from GeoQA+ [37] to expand the dataset to 7000. For the annotation of diagrams, we utilized the popular mathematics education website GeoGebra [46], which provides geometric drawing tools that enabled us to create basic geometric shapes and six types of diagram conditions. To enhance the diversity of diagram representations, we incorporated some conditions represented in the text CDL into the image CDL as well. Additionally, by analyzing the GGB files of the problems in GeoGebra, we extracted the pixel locations of geometric figures and information about geometric elements. To generate problem text that is consistent with the formal language, we employed a rule-based method to deformalize the text CDL and goal

CDL back into problem text. For this annotation, we organized 14 trained master students and provided them with annotation guidelines. Each annotated problem was reviewed by the same author, and any unsatisfactory annotations were identified for correction and re-annotation until they fully met the requirements. More details on the annotations can be found on the website [47].

3.3. Statistics

The FormalGeo7K dataset consists of 7000 problems, divided into training, validation, and test sets in a ratio of 0.7:0.15:0.15. Our formal language is based on three main principles: the ordering principle, the rotation invariance principle, and the counterclockwise principle. Therefore, the CDL of each problem can have multiple representations that comply with these principles. This allowed us to perform data augmentation on the CDL. Similarly, we applied random augmentation on the problem text to enhance its robustness. Each problem was expanded to create two variations, allowing the training set to reach 14,700 samples. Notably, we opted for online data augmentation, meaning that the augmented training set was randomly regenerated for each epoch, thereby increasing the diversity of the training data.

Moreover, the difficulty of automatically formalizing geometry problems using neural networks is closely related to the complexity of the diagrams and the number of problem conditions. We roughly categorized the difficulty levels of formalizing problems in the dataset based on the number of sub-statements in the CDL. All annotated and augmented geometric problems underwent interactive verification using the FGPS [16]. As shown in Figure 3, the difficulty of formalizing problems in FormalGeo7K generally follows a normal distribution, with the majority of problems containing between 10 and 20 sub-statements. Furthermore, the difficulty of formalizing problems is approximately proportional to their problem-solving difficulty. Problems with more complex diagrams and more conditions are likely to have a higher difficulty level. This reflects a balanced difficulty distribution within FormalGeo7K, which is beneficial for evaluating the performance of different models.

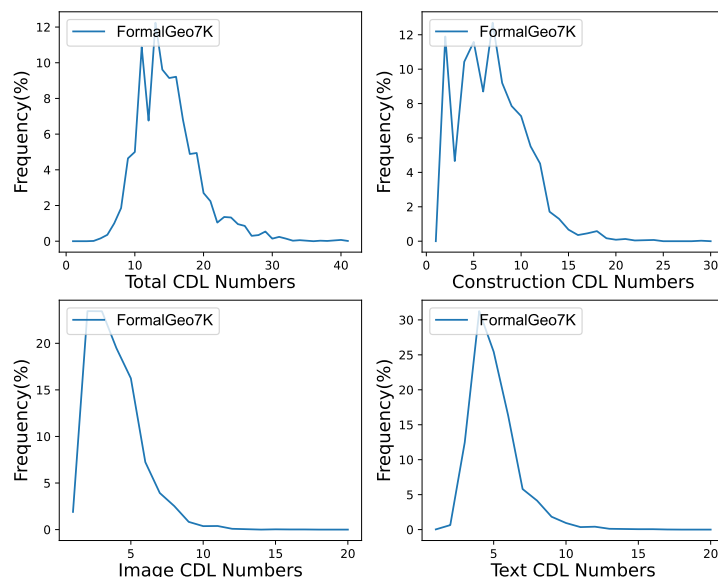


Figure 3. Difficulty distribution of the problem formalization in the FormalGeo7K.

4. FGeo-Parser

FGPS [16] can solve problems formulated in formal language through manual annotations, and with the assistance of a theorem predictor, it can achieve automated problem-solving. To fully automate geometric problem-solving, we also introduced a method termed

FGeo-Parser to facilitate the conversion between human language and machine language. The FGeo-Parser we proposed takes the diagram and text of problem as input. After processing through the diagram parser and text parser, it converts them into construction CDL, image CDL, text CDL, and goal CDL, which together constitute a geometric problem representation in formalized language. After being solved by FGPS with the assistance of a predictor, the deformalizer translates the solution process into human-readable natural language. The complete architecture is depicted in Figure 1.

4.1. Diagram Parser

The diagram contains geometric construction information along with supplementary textual conditions. Previous works [10,27,41] often involve complicated procedures or intricate models. In our work, we treat diagram parsing as image caption task and introduce an enhanced method that efficiently extracts diagram features and swiftly converts them into the corresponding formalized languages, specifically construction CDL and image CDL, which can be directly input into the FGPS.

The diagram parser is composed of a diagram encoder and a text decoder. We propose an enhanced method based on the BLIP [48], which is a vision language pre-training framework that offers greater generalization for downstream tasks. The diagram encoder utilizes a Vision Transformer (ViT) [49] that segments the image into patches and encodes them into an embedding sequence. Compared to traditional graph neural networks [50], the ViT is capable of capturing more global information. The text decoder inserts cross-attention (CA) between the feed-forward network (FFN) layer and the self-attention (SA) layer in each Transformer block to incorporate diagram information [51]. Furthermore, the bidirectional self-attention (Bi-SA) layer is replaced with a causal self-attention (Causal SA) [52] layer to accommodate text-generation tasks. The architecture of the diagram parser is demonstrated in Figure 4.

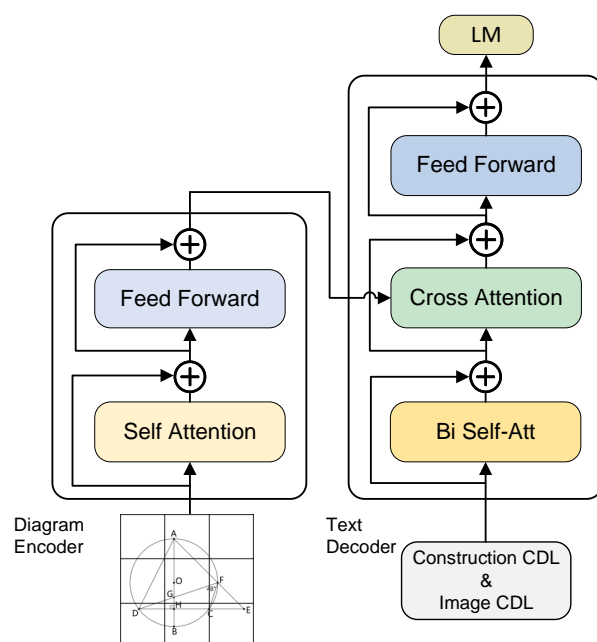


Figure 4. The architecture of diagram parser.

Additionally, we proposed a tokenization method called GeoTokenizer, which is tailored to fit our formal language. The vocabulary comprises all predicates and their sub-words from the formal system, along with all letters, commonly used symbols, and

several special tokens. We employed a Byte Pair Encoding (BPE) [53] approach to tokenize the formal language, converting it into a vector sequence.

We fine-tuned the pre-trained models using the online augmented dataset and the validation set. The fine-tuned models were then evaluated on the test set to assess the prediction results. During the fine-tuning process, cross-entropy loss was used for optimization to refine the generated outputs.

$$L_{CE} = -\frac{1}{N} \sum_{i=1}^N \sum_{c=1}^C y_{i,c} \log \hat{y}_{i,c} \quad (1)$$

L_{CE} refers to cross-entropy loss [54], where N is the total number of samples, C is the total number of classes, $y_{i,c}$ represents the true label indicating whether the i -th sample belongs to class c (with a value of 1 if true, otherwise 0), and $\hat{y}_{i,c}$ represents the predicted probability that the i -th sample belongs to class c . Additionally, cosine annealing learning rate scheduling [55] was used during the training process.

$$LR(t) = LR_{\min} + \frac{1}{2}(LR_{\max} - LR_{\min}) \left(1 + \cos\left(\frac{t\pi}{T}\right)\right) \quad (2)$$

Here, $LR(t)$ is the learning rate at time step; LR_{\min} and LR_{\max} are the minimum and maximum learning rates, respectively; t represents the current epoch; and T is the total number of epochs. The cosine function varies the learning rate smoothly between LR_{\min} and LR_{\max} over the training process. This approach enables the model to converge quickly during the early stages of prolonged training while reducing the learning rate in the later stages to prevent model oscillation.

4.2. Text Parser

The text parser is responsible for parsing the conditions and goals from the problem text. Some rule-based methods [10] struggle to handle the diversity of problem descriptions, leading to the omission of critical problem-solving information. With the support of a large-scale dataset, we employed the text-to-text unified framework of the T5 model [56] to transform the word sequences of the problem text into text CDL and goal CDL. For the input of the problem text, we used the T5Tokenizer to match the model, while the formal language output was tokenized using GeoTokenizer, consistent with the diagram parser.

Additionally, to enhance the diversity of text descriptions and reduce the impact of noise on the model, we applied an augmented training set during training, which was generated by an LLM through random rewrites of the original problem text [57] in the FormalGeo7K. Similar to the diagram parser, the text parser was also optimized using a cross-entropy loss function. The entire algorithm flow for image and text parsing is presented in Algorithm 1.

4.3. Solution Generator

To accomplish fully human-readable automated geometric problem-solving, after the parser converts the problem text and diagram into formal language readable by the FGPS and the problem is solved through theorem prediction, it is still necessary to deformalize the machine's solution process back into natural language. Our deformalizer uses the hyper-tree generated by the FGPS after successfully solving the problem as a reference. The root node represents the initial problem conditions, the intermediate nodes correspond to conditions during the solving process, and the leaf nodes represent the problem's goal, with the hyper-edges corresponding to the applied theorems. Ultimately, we used topological sorting to obtain the solution path from the initial conditions to the goal, and a rule-based method to convert the formal language into natural language.

Algorithm 1 FGeo-Parser.**Input:** *problem*: problem text and problem diagram.**Output:** *parsed_cdl*: parsed problem cdl.Initialize *text_parser*.Initialize *diagram_parser*.Initialize *parsed_cdl* as empty list.**if** *problem.text* is not *None* **then** *text_cdl* \leftarrow *text_parser.parse_text_cdl*() *goal_cdl* \leftarrow *text_parser.parse_goal_cdl*()**end if****if** *problem.diagram* is not *None* **then** *image_cdl* \leftarrow *diagram_parser.parse_image_cdl*() *cons_cdl* \leftarrow *diagram_parser.parse_cons_cdl*()**end if**Conduct a syntax check on all *cdl* and add it into *parsed_cdl*.

5. Experiments

5.1. Experimental Settings

Dataset and Implementation Details: We conducted experiments on the FormalGeo7K dataset, which included the deduplicated and fine-grained annotated Geometry3K and GeoQA+. The FormalGeo7K dataset was divided into 4700 training samples, 1050 validation samples, and 1050 test samples. Our experiments were implemented using PyTorch on an RTX 4090. In the diagram parser, the diagram encoder used a Vi-T, the processing problem diagrams were resized to 512×512 , the text decoder employed a modified Transformer, and the text parser was the T5 model. To increase dataset diversity, we applied a rule-based augmentation to the formal language and LLM rewriting to the problem text during training, expanding the training set to 14,700 samples. The loss function was the cross-entropy loss, and the optimizer was Adam. We set the initial learning rate to 1×10^{-5} , using cosine annealing learning rate scheduling to decay it to 5×10^{-6} . And the batch size was configured to 4, with a max epoch of 30.

Comparison Methods: To evaluate our FGeo-Parser, we performed two types of comparative experiments: Geometry Formal Language Parsing and FormalGeo System Problem-Solving. Given the unique formal language of the FGPS, we only compared it with several neural network models. Additionally, we evaluated problem-solving success rates under our solver system against current methods, including NGS [13] and DFE-GPS [57].

Evaluation Metrics: We evaluated these methods on two levels. The first level uses three metrics to assess the quality of the generated formal geometry language: BLEU-4 [58], match accuracy, and perfect accuracy. At the second level, the parsed CDL is input into the FGPS, combined with HyperGNet [59] to solve the problems in a fully automated process without human intervention. The final evaluation metric is problem-solving accuracy.

5.2. Geometry Formal Language Generation

Given the uniqueness of our formal language, we treated the task as a text-generation problem. Applying neural network methods for its solution, we assessed several pre-trained models for their accuracy in formal language generation. Ultimately, we selected the ResNet-LSTM model and the BLIP model as the diagram parser, and the T5 model as the text parser. As the diagram encoder, ResNet [50] effectively captured image features, which were then decoded into formal language by the LSTM [60]. In contrast, BLIP [48] employed a pre-trained ViT-BERT architecture, which provides superior generalization capabilities for specific downstream tasks. BLIP-14M refers to the model pre-trained on 14 million diagrams, while BLIP-129M refers to the model pre-trained on 129 million diagrams.

Different scales of ViT were employed, with ViT-B containing 12 Transformer blocks and ViT-L containing 24 Transformer blocks. Additionally, the T5 model [56] performed well in seq2seq tasks. The T5-base model has 220 million parameters, while the T5-large model has 770 million parameters.

We utilized the problem text and images as input, with CDL as output, to train various models using the training set, and fine-tuned them with the validation set. The final experimental results on the test set are presented in Table 1. We evaluated the model performance by BLEU-4, match accuracy, and perfect accuracy. In order to ensure that the formal language was correctly recognized by FGPS, we split each problem's CDL into several sub-statements. Match accuracy represents the ratio of correct sub-statements to the total number of sub-statements, while perfect accuracy indicates the proportion of problems where all sub-statements are correct relative to the total number of problems, with a sub-statement considered correct as long as it conforms to the FGPS principles. Since image CDL and text CDL both represent problem conditions and have a certain degree of overlap, we merged the results of the diagram parser and text parser before calculating match accuracy and perfect accuracy.

Table 1. Comparison of match accuracy and perfect accuracy on different formalization methods.

Method	BLEU-4 (%)	Match (%)	Perfect (%)
ResNet-LSTM + T5-base	87.40	62.63	11.52
BLIP _{ViT-B-14M} + T5-base	88.00	84.99	43.14
BLIP _{ViT-B-129M} + T5-base	91.40	91.31	56.19
BLIP _{ViT-L-129M} + T5-base	90.40	91.51	56.47

According to the experimental results in the Table 1, we can observe that the BLIP model significantly outperformed ResNet-LSTM overall. This difference arises because the max length of the sequence was set to 450 tokens, which limited LSTM's capacity to capture long-term dependencies, leading to a decline in generation quality. Although the BLEU-4 score remained acceptable, accurately generating formal language with multiple principles poses a substantial challenge that requires high precision. Also, the decoder of BLIP is based on a Transformer and is well suited to handling long-sequence tasks, which offers improved generalization for downstream tasks with extensive pre-training. Experimental results indicate that the BLIP model pre-trained on 129 million images outperforms the one pre-trained on 14 million images, with a 6.3% improvement in match accuracy and a 13.1% gain in perfect accuracy. This demonstrates that larger pre-training datasets can significantly amplify the model's capabilities, particularly in generating formal language. Furthermore, increasing the scale of the diagram encoder cannot yield a noticeable improvement in evaluation metrics, possibly due to the limited number of challenging problems in the dataset. Overall, FGeo-Parser can generally parse medium and lower-difficulty problems, but there remains substantial room for improvement in tackling more challenging problems.

5.3. Formalgeo System Problem-Solving

Our method serves as a bridge between human and machine reasoning, and it is essential to verify its ability to solve geometric problems. The ultimate goal is to achieve fully automated problem-solving, where the input is the original problem text and diagram, and the output is a natural language solution process. We integrated the FGeo-Parser and theorem predictor Hyper-GNet within FGPS for problem-solving, using problem-solving success accuracy as the evaluation metric. Additionally, we compared our approach with advanced methods such as NGS [13] and DFE-GPS [57]. Table 2 presents the success accuracy of each method on our FormalGeo7K dataset.

Table 2. Comparison of problem-solving accuracy on different methods.

Method	Acc (%)
NGS	30.49
DFE-GPS	64.76
Ours	63.45
Ours(gt)	85.53

Experimental results demonstrate that our method achieves a problem-solving accuracy of 63.45%. The multimodal NGS method achieves an accuracy of only 30.49% on the FormalGeo7K, which is restricted to solving multiple-choice questions and verifies its predicted solution programs for correctness. This indicates that FormalGeo7K presents certain challenges for it. DFE-GPS, leveraging a 9B-parameter LLM, achieves an accuracy of 64.76% when using text and images as input. However, it should be noted that its problem-solving accuracy is evaluated using GPT-4o-mini. In contrast, our FGPS matches the known conditions and obtains conclusions by applying theorems. The whole process is carried out step by step and is basically similar to the way humans solve problems, which has relatively strong interpretability. Moreover, the entire matching process is recorded in detail in the system, providing detailed operations for each step, which are traceable. When using the ground truth of text and diagram CDL as input, the solving accuracy reaches 85.53%, indicating substantial room for improvement in our parser to achieve higher-quality formal language generation.

5.4. Ablation Study and Discussion

In this section, we conduct four types of ablation experiments to analyze FGeo-Parser’s performance under various conditions. First, we examine the capability of the diagram parser and text parser to generate their respective CDLs by selecting the best-performing model from the second subsection. This model is tested on the test dataset for its match and perfect accuracy in generating different types of CDLs. As shown in Table 3, the text parser significantly outperforms the diagram parser, suggesting that seq2seq tasks are simpler than image2text tasks, as the latter involves cross-modal alignment and places higher demands on the model’s ability to align image and text spaces accurately. Furthermore, in diagram parsing, construction CDL primarily require general spatial information, whereas image CDL necessitate a more precise, integrated understanding of spatial and semantic content, resulting in higher parsing accuracy for construction CDL compared to image CDL.

Table 3. The match accuracy and perfect accuracy of different types of CDLs.

Method	CDL Type	Match (%)	Perfect (%)
Diagram Parser	Cons	86.99	61.23
	Image	77.48	55.61
	Cons + Image	83.04	40.10
Text Parser	Text	96.49	86.48
	Goal	96.75	96.38
	Text + Goal	96.60	86.29

Secondly, we analyzed the CDL generation quality by the model across different difficulty levels, where difficulty is defined based on the number of sub-statements in the CDL where match accuracy and perfect accuracy are used as evaluation metrics. From Figure 5, it is evident that accuracy decreases gradually as difficulty increases. In high-difficulty regions, some lines exhibit significant oscillations, likely due to the smaller

number of samples in these regions. Overall, the trend supports the feasibility of using the number of CDL sub-statements as a metric for categorizing difficulty levels.

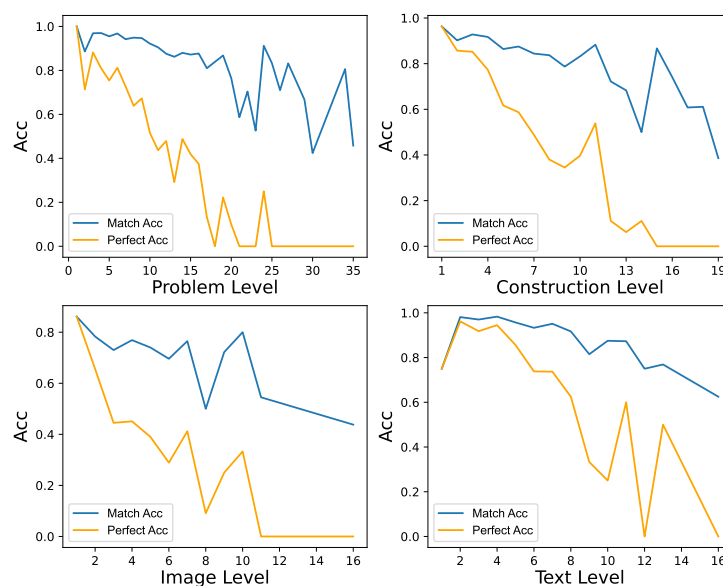


Figure 5. The match and perfect accuracy of each level in different CDLs.

Thirdly, to assess the impact of data augmentation on model performance, we conducted ablation experiments. Table 4 presents a comparison of FGeo-Parser’s results on the FormalGeo7K versus the FormalGeo7K augmented threefold. The augmented dataset boosted model performance, with match accuracy increasing by 2.9% and perfect accuracy by about 7.2%. This improvement is attributed to CDL’s inherent flexibility in representation, where online data augmentation enables the model to fully capture and recognize these variations.

Table 4. Comparison of FormalGeo7K and data-augmented FormalGeo7K.

Dataset	Match (%)	Perfect (%)
Formalgeo7k	88.48	49.04
Formalgeo7k (DA)	91.51	56.47

Finally, it is worth discussing FGeo-Parser’s performance comparison on datasets with different image sizes. Unlike complex real-world scenes, geometric figures are simpler and more intuitive, so intuitively, image resolution might not significantly impact the model’s ability to learn image features. However, our comparison between 256×256 and 512×512 pixel images showed that the latter increased match accuracy by 9.5% and perfect accuracy by approximately 15.1%, as presented in Table 5. This improvement may have stemmed from the complexity of formal language, which requires more precise positional and categorical information, making higher-resolution images beneficial for model performance.

Table 5. Comparison of different sizes of a problem image.

Image Size	Match (%)	Perfect (%)
256	81.93	41.14
512	91.51	56.47

Overall, FGeo-Parser effectively supports FGPS in achieving fully automated problem-solving, yet it is constrained by the requirement to generate formal language specific to FGPS. Furthermore, solving problems demands sufficient premise conditions, imposing strict requirements on the model to accurately generate all problem conditions to support subsequent prediction and problem-solving. Therefore, in future work, we plan to enhance FGeo-Parser's formalization accuracy, particularly when handling more complex geometric problem representations.

6. Conclusions

In this work, we focus on the automatic formalization of geometric problems and conduct a second fine-grained annotation work on problem diagram and text of the large-scale planar geometry dataset, FormalGeo7K, expanding it to 7000 problems. Additionally, we propose a neural network method, FGeo-Parser, which can automatically parse problem diagram and text, converting them into formal language within FGPS to support subsequent predictions to solve problems. Furthermore, we employ a deformalizer to convert the solution process of machine into human-readable natural language. Extensive experiments demonstrate that the FormalGeo7K dataset is challenging, and FGeo-Parser exhibits great formalization capabilities, which enables fully automated solutions with promising performance. In the future, we will conduct research on models that can obtain formal languages with higher precision since a high requirement for fully grasping the conditions of problems is essential for successfully solving them. Additionally, we plan to develop new formal systems to respond more flexibly to geometric problems of higher difficulty.

Author Contributions: Conceptualization, N.Z. and T.L.; Methodology, N.Z., X.Z., Q.H., F.Z. and T.L.; Software, N.Z.; Validation, N.Z.; Writing—original draft preparation, N.Z.; Writing—review and editing, N.Z., X.Z., Z.Z. and T.L.; Supervision, T.L.; 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, grant No. 12071282.

Data Availability Statement: The project is available at <https://github.com/RuRuo0/GeoParser> (accessed on 18 December 2024).

Acknowledgments: We thank the reviewers for their helpful comments.

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

References

1. Selsam, D.; de Moura, L.; Buzzard, K.; Barton, R.; Liang, P.; Loos, S.; Wiedijk, F. IMO Grand Challenge, 2019. Available online: <https://imo-grand-challenge.github.io/> (accessed on 14 September 2024).
2. TXMarkets. Artificial Intelligence Mathematical Olympiad Prize (AIMO Prize). 2023. Available online: <https://aimoprize.com/> (accessed on 14 September 2024).
3. 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.
4. Gelernter, H.L. Realization of a geometry theorem proving machine. In Proceedings of the IFIP Congress, Paris, France, 15–20 June 1959; pp. 273–281.
5. Nevins, A.J. Plane geometry theorem proving using forward chaining. *Artif. Intell.* **1975**, *6*, 1–23. [[CrossRef](#)]
6. Wu, W.T. On the decision problem and the mechanization of theorem proving in elementary geometry. *Sci. Sin.* **1978**, *21*, 157–179.
7. Buchberger, B. Applications of Gröbner bases in non-linear computational geometry. In *Mathematical Aspects of Scientific Software*; Springer: New York, NY, USA, 1988; pp. 59–87.
8. Wang, D. An elimination method for polynomial systems. *J. Symb. Comput.* **1993**, *16*, 83–114. [[CrossRef](#)]
9. 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](#)]

10. 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 (Volume 1: Long Papers), Online, 1–6 August 2021; pp. 6774–6786. [\[CrossRef\]](#)
11. 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. [\[CrossRef\]](#)
12. Trinh, T.H.; Wu, Y.; Le, Q.V.; He, H.; Luong, T. Solving olympiad geometry without human demonstrations. *Nature* **2024**, *625*, 476–482. [\[CrossRef\]](#)
13. 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 2021, Online, 1–6 August 2021; pp. 513–523. [\[CrossRef\]](#)
14. Zhang, M.L.; Yin, F.; Liu, C.L. A Multi-Modal Neural Geometric Solver with Textual Clauses Parsed from Diagram. In Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23, Macao, China, 19–25 August 2023; Volume 8, pp. 3374–3382. [\[CrossRef\]](#)
15. 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-24, Jeju, Republic of Korea, 3–9 August 2024; Volume 8, pp. 6559–6568. [\[CrossRef\]](#)
16. 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. [\[CrossRef\]](#)
17. Zhang, X.; Zhu, N.; He, Y.; Zou, J.; Huang, Q.; Jin, X.; Guo, Y.; Mao, C.; Li, Y.; Zhu, Z.; et al. FormalGeo: An Extensible Formalized Framework for Olympiad Geometric Problem Solving. *arXiv* **2024**, arXiv:2310.18021.
18. 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. [\[CrossRef\]](#)
19. 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. [\[CrossRef\]](#)
20. Yang, L.; Zhang, J.; Li, C. A prover for parallel numerical verification of a class of constructive geometry theorems. In Proceedings of the Proc. IWMM, St-Malo, France, 17–19 September 1992; Volume 92, pp. 244–250.
21. Gao, X.S.; Chou, S.C. On the dimension of an arbitrary ascending chain. *Chin. Sci.-Bull. Engl. Ed.* **1993**, *38*, 799.
22. Lu, Y. 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; pp. 24–35.
23. Wilson, S.; Fleuriot, J.D. Geometry Explorer: A tool for generating diagrammatic full-angle method proofs. In *Automated Deduction in Geometry: Extended Abstracts*; Dialnet: Universidad de La Rioja, Spain, 2006; pp. 144–150.
24. Ye, Z.; Chou, S.C.; Gao, X.S. An introduction to java geometry expert. In Proceedings of the Automated Deduction in Geometry: 7th International Workshop, ADG 2008, Shanghai, China, 22–24 September 2008; Revised Papers 7; Springer: Berlin/Heidelberg, Germany, 2011; pp. 189–195.
25. Chou, S.C.; Gao, X.S.; Zhang, J.Z. Automated production of traditional proofs in solid geometry. *J. Autom. Reason.* **1995**, *14*, 257–291. [\[CrossRef\]](#)
26. Yang, L.; Gao, X.S.; Chou, S.C.; Zhang, J.Z. Automated production of readable proofs for theorems in non-Euclidean geometries. In Proceedings of the Automated Deduction in Geometry: International Workshop on Automated Deduction in Geometry, Toulouse, France, 27–29 September 1996; Selected Papers 1; Springer: Berlin/Heidelberg, Germany, 1997; pp. 171–188.
27. 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, Lisbon, Portugal, 17–21 September 2015; pp. 1466–1476.
28. 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.
29. 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. [\[CrossRef\]](#)
30. 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. [\[CrossRef\]](#)
31. 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.
32. Gan, W.; Yu, X.; Wang, M. Automatic understanding and formalization of plane geometry proving problems in natural language: A supervised approach. *Int. J. Artif. Intell. Tools* **2019**, *28*, 1940003. [\[CrossRef\]](#)
33. 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. [\[CrossRef\]](#)

34. 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 2023, Toronto, ON, Canada, 9–14 July 2023; pp. 13468–13480. [CrossRef]
35. Tsai, S.H.; Liang, C.C.; Wang, H.M.; Su, K.Y. Sequence to General Tree: Knowledge-Guided Geometry Word Problem Solving. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), Online, 1–6 August 2021; pp. 964–972. [CrossRef]
36. 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, Abu Dhabi, United Arab Emirates, 7–11 December 2022; pp. 3313–3323. [CrossRef]
37. Cao, J.; Xiao, J. An Augmented Benchmark Dataset for Geometric Question Answering through Dual Parallel Text Encoding. In Proceedings of the 29th International Conference on Computational Linguistics, Gyeongju, Republic of Korea, 12–17 October 2022; pp. 1511–1520.
38. Ning, M.; Wang, Q.F.; Huang, K.; Huang, X. A Symbolic Characters Aware Model for Solving Geometry Problems. In Proceedings of the 31st ACM International Conference on Multimedia, MM '23, New York, NY, USA, 29 October–3 November 2023; pp. 7767–7775. [CrossRef]
39. Zhang, J.; Moshfeghi, Y. GOLD: Geometry Problem Solver with Natural Language Description. In Proceedings of the Findings of the Association for Computational Linguistics: NAACL 2024, Mexico City, Mexico, 16–21 June 2024; pp. 263–278. [CrossRef]
40. 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]
41. 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; Volume 7, pp. 1636–1643. [CrossRef]
42. 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, Vienna, Austria, 21–27 July 2024; Volume 235, pp. 36847–36893.
43. Zhang, J.; Li, Z.Z.; Zhang, M.L.; Yin, F.; Liu, C.L.; Moshfeghi, Y. GeoEval: Benchmark for Evaluating LLMs and Multi-Modal Models on Geometry Problem-Solving. In Proceedings of the Findings of the Association for Computational Linguistics ACL 2024, Bangkok, Thailand, 11–16 August 2024; pp. 1258–1276.
44. Hao, Y.; Zhang, M.; Yin, F.; Huang, L.L. PGDP5K: A diagram parsing dataset for plane geometry problems. In Proceedings of the 2022 26th International Conference on Pattern Recognition (ICPR), Montréal, QC, Canada, 21–25 August 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1763–1769.
45. Zhang, X.; Zhu, N.; Qin, C.; Yang, L.; Zeng, Z.; Leng, T. Formal Representation and Solution of Plane Geometric Problems. In Proceedings of the The 4th Workshop on Mathematical Reasoning and AI at NeurIPS'24, Vancouver, BC, Canada, 14 December 2024.
46. GeoGebra Geometry. 2024. Available online: <https://mat.geogebra.org/geometry> (accessed on 3 November 2024).
47. Geometry—GeoGebra—geogebra.org. Available online: <https://www.geogebra.org/geometry> (accessed on 7 November 2024).
48. Li, J.; Li, D.; Xiong, C.; Hoi, S. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In Proceedings of the International Conference on Machine Learning. PMLR, Baltimore, MD, USA, 17–23 July 2022; pp. 12888–12900.
49. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *arXiv* **2020**, arXiv:2010.11929.
50. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
51. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; MIT Press: Cambridge, MA, USA, 2017; pp. 5998–6008.
52. Yang, X.; Zhang, H.; Qi, G.; Cai, J. Causal attention for vision-language tasks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 9847–9857.
53. Sennrich, R. Neural machine translation of rare words with subword units. *arXiv* **2015**, arXiv:1508.07909.
54. Shannon, C.E. A mathematical theory of communication. *Bell Syst. Tech. J.* **1948**, *27*, 379–423. [CrossRef]
55. Loshchilov, I.; Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv* **2016**, arXiv:1608.03983.
56. 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.
57. 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. *arXiv* **2024**, arXiv:2409.04214.
58. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.J. Bleu: A method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, PA, USA, 6–12 July 2002; pp. 311–318.

59. Zhang, X.; Zhu, N.; He, Y.; Zou, J.; Qin, C.; Li, Y.; Zeng, Z.; Leng, T. FGeo-HyperGNet: Geometry Problem Solving Integrating Formal Symbolic System and Hypergraph Neural Network. *arXiv* **2024**, arXiv:2402.11461.
60. Graves, A.; Graves, A. Long short-term memory. Supervised Sequence Labelling with Recurrent Neural Networks. Ph.D. Thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2012; pp. 37–45.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.