GeoDRL: A Self-Learning Framework for Geometry Problem Solving using Reinforcement Learning in Deductive Reasoning

Shuai Peng[1], **Di Fu**[2], **Yijun Liang**[2], **Liangcai Gao**[1]*, **Zhi Tang**[1]
[1]Peking University, Beijing, China
[2]ByteDance, Beijing, China
`{pengshuaipku, gaoliangcai, tangzhi}@pku.edu.cn`
`{fudi.01, liangyijun}@bytedance.com`

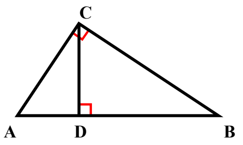
Abstract

Ensuring both interpretability and correctness is a great challenge in automated geometry problem solving (GPS), and the scarcity of labeled data hinders learning mathematical reasoning from samples. Therefore, we present GeoDRL, a self-learning geometry problem solving framework that integrates logic graph deduction and Deep Reinforcement Learning (DRL) to optimize geometry reasoning as a Markov Decision Process. GeoDRL employs a Graph Neural Network on a Geometry Logic Graph, updating the problem state using a symbolic system. Incorporating DRL into deductive reasoning enables GeoDRL to achieve unsupervised self-learning while maintaining correctness. GeoDRL, through unsupervised learning, exhibits enhanced accuracy in the Geometry3K dataset, improving by 11.1% over previous SOTA methods, and simultaneously boosts efficiency and interpretability.

1 Introduction

Automated geometry problem solving (GPS) has been a long-standing task in mathematical reasoning research, drawing significant attention from researchers (Wen-Tsun, 1986; Chou et al., 1996; Seo et al., 2015; Sachan and Xing, 2017; Lu et al., 2021, 2022b). As depicted in Figure 1, GPS can be described as follows: given a geometry problem description which typically consists of a diagram and text, the objective is to produce a flow of solution steps leading to the final answer. Solving a geometry problem consists of two key steps: *parse and reason*, where parsing involves extracting specific interpretation from the question diagram and text and reasoning involves using the interpretation to develop a solution.

There has been a long-standing debate between symbolic and probabilistic approaches in mathematical reasoning. In geometry problem solving,



Question: In triangle *ABC*, *AD* = 4 and *BD* = 9. Find *CD*.

Answer: *CD* = 6.

Solution:

Step 1: (Triangle Exterior Angle Theorem)

$$\left.\begin{array}{l}\angle BAC + \angle ACD = \angle BDC = 90^\circ \\ \angle BCD + \angle ACD = \angle ACB = 90^\circ\end{array}\right\} \Rightarrow \angle BAC = \angle BCD$$

Step 2: (Angle-Angle Similarity Theorem)

$$\left.\begin{array}{l}\angle BAC = \angle BCD \\ \angle ADC = \angle BDC\end{array}\right\} \Rightarrow \triangle ACD \backsim \triangle CBD$$

Step 3: (Side-Side-Side Similarity Theorem)

$$\triangle ACD \backsim \triangle CBD \Rightarrow \frac{AD}{CD} = \frac{CD}{BD} \Rightarrow CD = \sqrt{AD \cdot BD} = 6$$

Figure 1: A sample of plane geometry problem, including a question diagram and textural description. The figure also shows a potential solution process with three steps.

the symbolic approaches (Seo et al., 2015; Sachan and Xing, 2017; Lu et al., 2021) first parse the question diagram and text into logical expressions in a formal symbolic language, and then utilize a predefined set of geometry theorems to solve the problem. These approaches are known for the mathematical rigor, as they construct a symbolic system for logical interpretation and theorem application. In contrast, the probabilistic approaches (Chen et al., 2021, 2022; Zhang et al., 2023) view GPS as a special case of text generation with multi-modal input. These data-driven approaches learn mathematical language and solving strategies from problem-solution samples to generate human-readable solutions without prior mathematical knowledge.

Nevertheless, there are notable limitations to both categories. The symbolic approaches struggle with selecting the appropriate theorem (Gelernter, 1959; Nevins, 1975). The heuristic search strategies tend to involve redundant steps (Lu et al., 2021), which harms the readability of the solution. On the other hand, the probabilistic approaches rely heavily on manually annotated data, which is often scarce and expensive. Furthermore, the solutions generated by deep neural network are doubtful in mathematical rigor and correctness (Chen et al., 2021, 2022). Even an advanced language model like GPT-4 is prone to make arithmetic mistakes

* Corresponding Author

(Bubeck et al., 2023), which is unacceptable in mathematical problem solving.

Accordingly we propose a self-learning GEOS framework, GeoDRL, to addresses these issues. Our framework consists of two phases:

1. Parsing the question diagram and text into a Geometry Logic Graph (GLG). GLG is a heterogenous attributed graph encoding geometric information, where nodes represent geometric primitives (point, line, etc.), edges represent relationships between primitives (perpendicular, parallel, etc.), and attributes represent properties of primitives (length of line segment, measure of angle).

2. Integrating Deep Reinforcement Learning (DRL) into the reasoning procedure. Here, GLG serves as the *state* and a predefined geometry theorem set serves as the *action space*. We implement a Graph Neural Network (GNN) to learn the graph representation of GLG to estimate the Q-value of the state. At each step, DRL agent selects a geometry theorem as an action for the current state according to the Q-value. The symbolic system performs it, updates the state, and provides a reward until the problem is solved.

By introducing reinforcement learning, we construct a self-learning framework that enables unsupervised learning of problem-solving strategies, alleviating the scarcity of labeled data. Furthermore, our framework integrates neural network and symbolic system in the deduction, resulting in interpretable step-by-step solutions while maintaining mathematical correctness.

Our contributions are summarized as follows:

- We propose a self-learning geometry problem solving framework, GeoDRL, which integrates Deep Reinforcement Learning into deductive geometry reasoning. This enables training without annotated solution data.

- We integrate decision-making of neural network and symbol-manipulation of symbolic system in our framework, which enables interpretable step-by-step deduction while maintaining mathematical correctness.

- We structure geometry information as Geometry Logic Graph(GLG), which preserves geometry primitives, attributes and relations. GLG demonstrates its effectiveness in geometry interpretation modeling by outperforming sequential logical expressions.

2 Related Work

2.1 Geometry Problem Solving

Early work focus on geometry theorem proving (Gelernter et al., 1960; Wen-Tsun, 1986; Chou et al., 1996), which is to demonstrate the truth of a geometric proposition through a logical deduction. These early efforts utilized symbolic computer systems to perform logical deductions and symbol manipulation. More recently, systems for solving geometry problems, such as GEOS (Seo et al., 2015) and Inter-GPS (Lu et al., 2021), have been developed. These systems take geometry diagram and text as input and provide a problem solution, which involves parsing multi-modal information, utilizing theorem knowledge, and conducting quantitative reasoning. In recent years, there has been a trend of training language model on large-scale corpora to learn mathematical reasoning. Examples in the geometry domain include GeoQA (Chen et al., 2021) and UniGeo (Chen et al., 2022), which view GPS as a special case of text generation and use language models to generate solutions. While this approach is highly generalizable, it also carries the risk of lacking mathematical rigor.

2.2 Deep Reinforcement Learning

Deep Reinforcement Learning (DRL) is a machine learning approach that aims to learn a long-term strategy to maximize the reward signal in an optimization problem. In recent years, there have been numerous studies that apply DRL to solve mathematical problems, including math word problems (Wang et al., 2018; Lu et al., 2022a), arithmetic expression calculation (Chen et al., 2018), math theorem proving (Kaliszyk et al., 2018), symbolic reasoning (Poesia et al., 2021) and so on. But there are very few explorations that view geometry problem solving as a RL task. In this paper, we model geometry reasoning as a Markov Decision Process on a logic graph and use Deep Q-Network (DQN) (Mnih et al., 2013) to choose appropriate theorems for deductive reasoning.

3 Problem Statement and Preliminaries

3.1 Problem Definition

In this paper, we focus on Plane Geometry Problem Solving. We define the task as follows: given a

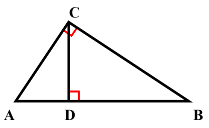
Problem	Interpretation
 **Question**: In triangle *ABC*, *AD* = 4 and *BD* = 9. Find *CD*.	**Primitives**: Point: *A, B, C, D* Line: *AB, AC, AD, BC, BD, CD* Angle: $\angle BAC, \angle ABC, \angle ACB, \angle ACD, \angle BCD, \angle ADC, \angle BDC$ Triangle: $\triangle ABC$, $\triangle ACD$, $\triangle BCD$ **Attributes**: $AD = 4, BD = 9$ $\angle ACB = 90°, \angle BDC = 90°$ **Predicates**: *D LiesOn AB*

Table 1: An example of geometry interpretation, including primitives, attributes, and predicates.

plane geometry problem $P = \{D, T\}$ including diagram D and text T, the objective is to find a deduction sequence τ that satisfies the goal g of P. The basic terms in our framework are defined below.

Definition 3.1 (Primitive) A *primitive* is a basic geometry element that constitutes the diagram, such as point, line, circle, polygon, etc.

Definition 3.2 (Attribute) An *attribute* is a property of the primitive, usually expressed as a numerical value, such as the length of a line segment, the measure of an angle, and the radius of a circle.

Definition 3.3 (Predicate) A *predicate* is a geometry relation between primitives, such as point lies on line, line 1 is perpendicular to line 2, etc.

Definition 3.4 (Interpretation) An *interpretation* is a complete description of all primitives, attributes, and predicates in a geometry problem.

Definition 3.5 (Theorem) A *theorem* is a priori geometric knowledge applied in interpretation to obtain new primitives, attributes or relations.

Table 1 shows an example of the interpretation of a geometry problem. We further define solution steps as follows: in a n-steps deduction sequence $\tau = \{p_1, p_2, ..., p_n\}$, p_t is the t-th step in τ, which is expressed as $p_t = I_{t-1} \xrightarrow{k_t} I_t$, where $k \in \mathcal{KB}$ is a theorem in geometry theorem set $\mathcal{KB}$ and I is the geometry interpretation.

Based on the above definitions, the flow of GeoDRL is described as follows:

1. The parser implements a function $Parse(\cdot)$ to extract the initial geometry interpretation I_0 and the goal g from the question diagram D and text T, which is formulated as $I_0, g = Parse(D, T)$.

2. The reasoner implements a function $Reason(\cdot)$ to produce a deduction sequence τ, given the initial interpretation I_0, the goal g and a geometry theorem set $\mathcal{KB}$, which is formulated as $\tau = Reason(I_0, g, \mathcal{KB})$

A remaining issue is the selection of the optimal solution τ^* from multiple candidate sequences $C = \{\tau_1, \tau_2, ...\}$. We prefer to choose the shortest solution, and if there are multiple solutions of the same length, choose the one with a lower time. It is formulated as $\tau^* = \text{argmin}_{\tau \in C}(|\tau|, Time(\tau))$

3.2 Reinforcement Learning Modeling

In 3.1, we described geometry problem solving, where the reasoning is modeled as solving Markov Decision Process (MDP) (Puterman, 2014). The MDP for geometry reasoning consists of all the potential geometry interpretation states and the transition probabilities between them. Hence, we adopt Q-Learning, a DRL algorithm to choose actions based on the Q-value of state. In this framework, geometry interpretations is the state space $\mathcal{S} = \{s_i | i = 1, 2, ...\}$, geometry theorem set is the action space $\mathcal{A} = \{a_i | i = 1, 2, ...\}$. DRL agent is a deep neural network that produces value $Q(s, a)$ for selecting action a in state s. Each time step environment performs an action according to $Q(s, a)$, updates to the new state s', and provides a reward r. The goal of the algorithm is to make the agent learn a policy $\pi : \mathcal{S} \to \mathcal{A}$ to maximize the accumulated reward.

4 Approach

This section introduces our proposed geometry problem solving framework, GeoDRL. The overall architecture is illustrated in Figure 2. Here we follow the flow described in 3.1 and introduce the parser and reasoner separately.

4.1 Parser

The parser is aimed to extract primitives and attributes from the question diagram and text, build relations between primitives, and parse them into the interpretation. For structured modeling of geometry information, we propose Geometry Logic Graph (GLG) as the representation of geometry interpretation in our framework. GLG is a heterogenous attributed graph $G = (V, E, \Sigma, L)$ where (1) V is the set of vertices denoting primitive types; (2) $E \subseteq V \times V$ is the set of edges and $e = (u, v) \in E$ is an edge from vertex u to vertex v, denoting geometry relation between these two primitives; (3) Σ is the domain of attributes and L is the function that assigns attributes to vertices. $l(v)$ is the attribute

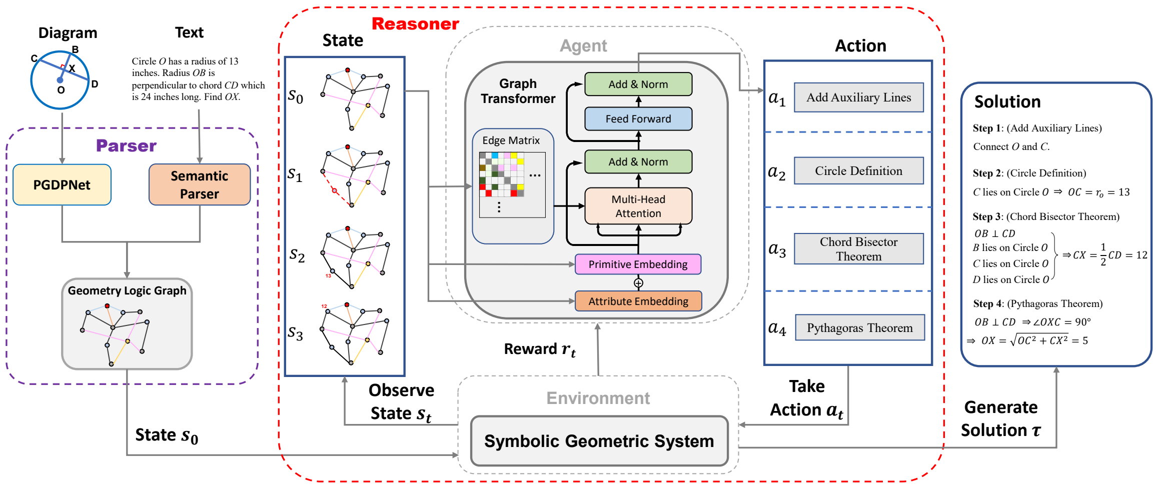


Diagram
Text
Circle O has a radius of 13 inches. Radius OB is perpendicular to chord CD which is 24 inches long. Find OX.
Parser
PGDPNet
Semantic Parser
Geometry Logic Graph
Reasoner
State
s_0
s_1
s_2
s_3
Agent
Graph Transformer
Edge Matrix
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Primitive Embedding
Attribute Embedding
Action
a_1 Add Auxiliary Lines
a_2 Circle Definition
a_3 Chord Bisector Theorem
a_4 Pythagoras Theorem
Reward r_t
Observe State s_t
Environment
Symbolic Geometric System
Take Action a_t
State s_0
Solution
Step 1: (Add Auxiliary Lines)
Connect O and C.
Step 2: (Circle Definition)
C lies on Circle $O \Rightarrow OC = r_o = 13$
Step 3: (Chord Bisector Theorem)
$OB \perp CD$, B lies on Circle O, C lies on Circle O, D lies on Circle O $\Rightarrow CX = \frac{1}{2}CD = 12$
Step 4: (Pythagoras Theorem)
$OB \perp CD \Rightarrow \angle OXC = 90°$
$\Rightarrow OX = \sqrt{OC^2 + CX^2} = 5$
Generate Solution τ

Figure 2: The overall architecture of GeoDRL. It consists of three stages: the parsing stage, where the problem is parsed into logical forms and structured into a GLG; the reasoning stage, where the actions and state transitions are incrementally deduced through interaction between the agent and symbolic geometry system; and the answering stage, where a human-readable solution is delivered from the deductive sequence.

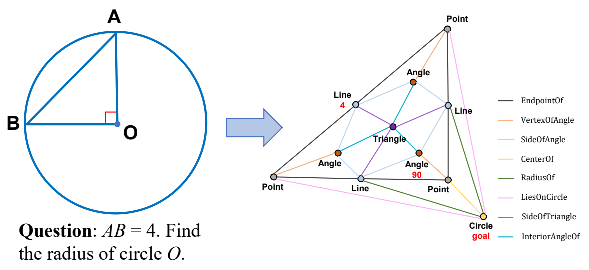


Figure 3: An example of parsing a problem into a GLG. GLG preserves geometry interpretation from the original diagram and text and adds implicit relations between primitives.

of v. For example, if v denotes a line segment AB, then $l(v)$ denotes the length of AB. GLG is a directed graph, where each edge $e = (u, v)$ between different types of primitive has its inverse edge $e^{-1} = (v, u)$. Furthermore, to represent the interpretation I and the goal g within one graph, we add the goal g as a special attribute of targeted primitives in the GLG. Figure 3 shows an example of GLG.

Here we discuss the advantages of GLG compared with symbolic logical expressions in previous works. First, GLG is structured as a graph incorporating geometry interpretation into its nodes and edges, which avoids redundant symbolic grammar. Second, in symbolic logical expression, a primitive is identified by its notation, resulting in the feature learned by the neural network being specific to its notation. It is counter-intuitive, since changing the notation of a point from A to X won't affect the problem substantially. In GLG, the feature of a primitive is only determined by its type, attribute and relations with others, instead of its notation.

Specifically, we parse the diagram and text separately. For the diagram, we use PGDPNet (Zhang et al., 2022) as the parser. PGDPNet is an end-to-end model for extracting primitives and predicates from geometry diagrams, achieving *state-of-the-art* performance on plane geometry diagram parsing. PGDPNet takes a diagram image as input, generates the instances of points, lines and circles, and the logical expressions indicating relationships between primitives. For the text, we borrow the rule-based text parser in Inter-GPS (Lu et al., 2021). The semantic parser parses text into symbolic logical expressions, usually including the goal. We combine the logical expressions from the above two branches and structure them into a GLG, consisting of an initial interpretation I_0 and the goal g.

4.2 Reasoner

Given an initial interpretation I_0, a goal g, and a geometry theorem set $\mathcal{KB}$, the reasoner is aimed to produce a deduction sequence τ where I_t satisfies the goal g. We model it as a Markov Decision Process and adopt DRL to solve the MDP. We implement an environment, i.e., a symbolic geometric system. The system parses the geometry interpretation and goal, performs actions, updates states and gives rewards. We design a DRL agent to implement the DQN algorithm (Mnih et al., 2013), using GNN to approximate the q-value function.

DRL agent selects an action for the current state according to the Q-value, and interacts with the environment to obtain the reward and the next state.

4.2.1 Environment

The environment is depicted as a symbolic geometric system. The system resolves GLG from 4.1 and stores the geometry interpretation and goal. It applies a geometry theorem, updates the interpretation and generates a reward according to the action instruction given by the agent. In an update process $s \rightarrow s'$, our reward function is set as follows:

$$Reward = \begin{cases} Solve(s', g) - \alpha e^{-\frac{t}{\sigma}} & s \neq s' \\ -1.0 & s = s' \end{cases} \tag{1}$$

$$Solve(s, g) = \begin{cases} 1.0 & s \vdash g \\ 0.0 & \text{otherwise} \end{cases} \tag{2}$$

Where t denotes the time spent in the process. The system provides a positive reward 1.0 only when the next state s' satisfies the goal g, that is, the problem is solved. We add a time penalty factor $-\alpha e^{-\frac{t}{\sigma}}$ to encourage the agent to choose more efficient actions. If the state is not changed by applying the theorem, the reward is set to -1.0 to avoid this action. In one word, the reward function encourages agent to choose a shorter and faster path with less invalid actions.

4.2.2 Action Space

Action space is defined as a geometry theorem set $\mathcal{KB}$, where each theorem k is defined as a conditional statement $p \Rightarrow q$ with a premise p and a conclusion q. At time step t, when applying theorem k on interpretation I_{t-1}, if I_{t-1} satisfies the premise p of k, then I_{t-1} is updated according to the conclusion q:

$$I_{t-1} \vdash p \Rightarrow I_t \leftarrow I_{t-1} \wedge q \tag{3}$$

If I_{t-1} does not satisfy the premise p, it will not be updated. Moreover, even if p is satisfied, I_t may remain unchanged. It often happens when a theorem is used repeatedly in succession. We call both cases invalid theorems, which leads to $r = -1.0$.

Specifically, we define 24 theorems in this work, covering various classes such as angles, circles, triangles, and polygons. In addition, we define auxiliary-line as a class to deal with some difficult geometry problems.

4.2.3 GNN Architecture

We utilize a GNN to learn the graph representation of GLG to estimate the Q-value of the state. This is accomplished through the implementation of a Heterogeneous Attributed Graph Transformer built upon a transformer-based architecture. Given a GLG $G = (V, E, \Sigma, L)$, the input of our model is a sequence of nodes with a special token [GRAPH] as the start. [GRAPH] aggregates the node features to capture the global graph representation. Each node includes node type v_i and node attribute $l(v_i)$, which are projected to node embedding h_i:

$$h_i = E_{type}(v_i) + E_{attr}(l(v_i)) \tag{4}$$

Node type is the type of primitive such as point, line and circle. Node attribute is commonly expressed as a number or math expression with variables. In practice, recording the specific value of every attribute makes little sense. For example, *AB=3.2cm, BC=3.5cm.* In this case, we only need to know that the lengths of AB and BC are two unequal numerical values. Therefore, we use several replacement words to replace the less frequent attribute values.

The model encodes the structural information of GLG by modifying the self-attention mechanism in transformer. Edges are incorporated into self-attention by adding bias term matrix B, formulated as:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d}} + B)V \tag{5}$$

$$B_{i,j} = \begin{cases} E_{edge}(e_{i,j}) & (v_i, v_j) \in E \\ 0 & i = j \\ -\infty & \text{otherwise} \end{cases} \tag{6}$$

When there exists an edge $e_{i,j}$ from v_i to v_j, the edge embedding of $e_{i,j}$ is added into the position (i, j) of attention score as a bias term. Otherwise, we block the attention channel by setting the score to -∞. Global graph token [GRAPH] attends to every node to aggregate all the features as the global graph feature. As a result, we implement a one-hop graph attention network. Finally, the graph feature vector is sent to an FFN with the output dimension of action space size $|\mathcal{A}|$ to produce the Q-value.

4.2.4 Agent Operation

DRL agent operates by interacting with the environment, i.e., the symbolic system. Algorithm

Algorithm 1 DRL Agent Operation

```
1:  $s, g \leftarrow env.init()$
2:  $M \leftarrow \{\}$
3:  $Solved \leftarrow$ False
4:  while $Solved$ = False do
5:      $Q(s, a_i) \leftarrow \mathrm{GNN}(s), i = 0, 1, 2, ...|\mathcal{A}|$
6:      $a \leftarrow epsilon - greedy(Q(s, a_i), \epsilon)$
7:      $s', r \leftarrow env.step(s, a)$
8:      if $s' \vdash g$ then
9:          $Solved \leftarrow$ True
10:     end if
11:     $M$.append($(s, a, s', r)$)
12:     if M.amount() > N then
13:         GNN.update()
14:     end if
15:     $s \leftarrow s'$;
16: end while
```

1 describes DRL agent operation. First, the system initializes an interpretation with GLG obtained from 4.1 as the initial state s_0. Then it enters a while loop, where each time step t GNN takes GLG of s_{t-1} as input and calculates $Q(s_{t-1}, a_i)$ for every action a_i. DRL agent adopts ϵ-greedy exploration strategy to choose action a_t. The symbolic system performs action a_t, updates state s_t, and returns a reward r_t. Meanwhile, the system checks if $s_t \vdash g$, and if so, assigns a flag *Solved* indicating jumping out of the loop. The data samples $\{(s_{t-1}, a_t, s_t, r_t)|t = 1, 2, ...\}$ are accumulated and stored into an experience memory. After it contains a certain amount of samples, we randomly sample one batch each time from it to train the Q-Net using the Bellman-Equation:

$$Q(s, a) = \max_{a'}(\gamma Q(s', a') + r(s, a)) \quad (7)$$

where the training objective is to minimize the SmoothL1 loss (Girshick, 2015) of $Q(s, a)$ and $\max_{a'}(\gamma Q(s', a') + r(s, a))$. In the inference stage, DRL agent chooses the action a that maximizes $Q(s, a)$ for the current state s. The symbolic system performs the action and updates the state until the problem is solved. Finally, we gather each step p_t and obtain the deduction sequence τ.

Considering the interpretability of the solution, τ needs to be further processed to deliver a human-readable answer. There might be redundant updated parts in interpretations. For example, there are several pairs of similar triangles at step 2 in Figure 1, but only $\triangle ACD \sim \triangle CBD$ leads to the goal. To address it, we record the $I_n \cap g$ and backtrack the updates that lead to it. Eventually, we get the final answer.

5 Experiments

In this section, we conduct experiments on public geometry problem datasets and compare the performance of GeoDRL against the previous methods. Furthermore, we analyze the contribution of different schemes by ablation study and discuss the superiority and limitations of our model with typical cases.

5.1 Implementation and Setup

5.1.1 Dataset and Parameter Settings

We mainly conduct experiments on Geometry3K (Lu et al., 2021), a public geometric dataset with 3000 SAT-style plane geometry problems. PGDP-Net is employed as the parser with the public available model weight[1]. Our symbolic system is built-upon Inter-GPS (Lu et al., 2021), with an improved logical parser and expanded geometry theorem set. We implement a Graph Transformer with 12-layer, 12 attention heads, a maximum of 256 nodes, and a dimensionality of 768, as the Q-Net. For fairness, transformer-based baselines share the same parameter settings and random seeds. Each experiment for GeoDRL is repeated three times and obtained the mean scores. For more parameter settings and experiment details, please refer to Appendix A.

5.1.2 Training Method

To accelerate the training process by avoiding extensive initial exploration, we adopt Imitation Learning (IL) method to make our agent learn an initial policy. Specifically, we utilize the heuristic strategy in Lu et al. (2021) to search for deduction sequences on a few training data samples (500 problems) and train a teacher model on them. We then use the teacher model to interact with the environment and generate samples to Experience Memory following Algorithm 1. A randomly-initialed student model is then pre-trained on these samples for 2000 steps to learn an initial policy from the experiences of the teacher model. Following this, we proceed to train the student model as a DRL agent on the train set.

[1]https://github.com/mingliangzhang2018/PGDP

Method	All	Angle	Length	Area	Ratio	Line	Triangle	Quad	Circle	Other
Human	56.9	53.7	59.3	57.7	42.9	46.7	53.8	68.7	61.7	58.3
Human Expert	90.9	89.9	92.0	93.9	66.7	95.9	92.2	90.5	89.9	92.3
FiLM(Perez et al., 2018)	31.7	28.7	32.7	**39.6**	33.3	33.3	29.2	33.6	30.8	29.6
FiLM-BERT(Devlin et al., 2019)	32.8	32.9	33.3	30.2	25.0	32.1	32.3	32.2	34.3	33.3
FiLM-BART(Lewis et al., 2020)	33.0	32.1	33.0	35.8	50.0	34.6	32.6	37.1	30.1	37.0
Inter-GPS(Lu et al., 2021)	57.5	59.1	61.7	30.2	50.0	59.3	66.0	52.4	45.5	**48.1**
Inter-GPS(GT)	78.3	83.1	77.9	62.3	75.0	86.4	83.3	77.6	61.5	70.4
GeoDRL(ours)	**68.4**	**75.5**	**70.5**	22.6	**83.3**	**77.8**	**76.0**	**62.9**	**53.8**	**48.1**
GeoDRL(GT)	**89.4**	**86.5**	**93.7**	**75.5**	**100.0**	**87.7**	**93.1**	**90.2**	**78.3**	**77.8**

Table 2: Accuracy(%) results of GeoDRL and compared baselines on the Geometry3K dataset. GT denotes using the ground truth parsing results, which can be considered as an individual evaluation of the reasoner.

Method	Settings	Overall Acc(%)	Avg Time(s)	Avg Step
Inter-GPS(GT)	Predict	77.5	6.35	6.68
	Low-first	77.3	9.41	15.30
GeoDRL(GT)	Beam Size=1	84.7	**2.67**	**2.27**
	Beam Size=5	**89.4**	6.03	2.34

Table 3: Evaluation results of GeoDRL and Inter-GPS with different settings. Avg Time means the average time spent on each solved problem. Avg Step means the average number of steps to solve each problem.

5.1.3 Inference

In the inference stage, we adopt the Beam Search algorithm to choose the final solution τ^*. The score of a candidate sequence τ is calculated as:

$$p(s_t|s_{t-1}, a_t) = \frac{e^{Q(s_{t-1},a_t)}}{\sum_{a' \in \mathcal{A}} e^{Q(s_{t-1},a')}} \tag{8}$$

$$score(\tau) = \sum_t log(p(s_t|s_{t-1}, a_t)) \tag{9}$$

Where $p(s_t|s_{t-1}, a_t)$ is the transition probability from s_{t-1} to s_t by taking action a_t, which is calculated by applying softmax function on the Q-value. With the beam size of k, we finally obtain k candidate sequences $\{\tau_1, \tau_2, ..., \tau_k\}$ with the top k scores and select the optimal τ^* as the result. In this study, we implement GeoDRL with the default beam size of 5.

5.2 Evaluation

We evaluate GeoDRL on the Geometry3K dataset with several baselines: three probabilistic approaches including FiLM (Perez et al., 2018), FiLM-BERT (Devlin et al., 2019), FiLM-BART (Lewis et al., 2020), and a symbolic approach Inter-GPS (Lu et al., 2021), which is the state-of-the-art method on Geometry3K. Table 2 shows the results. GeoDRL significantly outperforms the other methods in overall accuracy and most problem types. Compared to the previous state-of-the-art method Inter-GPS (Lu et al., 2021), GeoDRL obtains 11.1% and 10.9% enhancements in overall accuracy with or without the ground truth parsing results. Notablely, GeoDRL(GT) achieves comparable accuracy scores to the human expert, which demonstrates the superiority of our reasoner.

We evaluate the efficiency of GeoDRL and Inter-GPS. For fairness, both methods use ground truth parsing results of Geometry3K to avoid the impact of different parsers. As shown in Table 3, compared to Inter-GPS, GeoDRL demonstrates superior performance, achieving not only higher accuracy scores - up to 89.4% with beam size 5 versus 77.5% for Inter-GPS, but also taking less time - a minimum average of 2.67 seconds against Inter-GPS's 6.35 seconds. Additionally, GeoDRL solves each problem in fewer steps, averaging 2.27 steps with beam size 1, in contrast to Inter-GPS's 6.68 steps. Considering that our symbolic system is derived from Inter-GPS, comparing the time and steps of these two methods makes sense. Due to the heuristic algorithm in choosing theorems, Inter-GPS is found to generate solutions with redundant steps, leading to a longer solving time and more steps. Overall, GeoDRL is notably superior to the previous state-of-the-art Inter-GPS in terms of effectiveness and efficiency.

5.3 Ablation Study

We conduct an ablation study on Geometry3K to figure out the contributions of different schemes in our framework. Results are shown in Table 4. Compared to traditional symbolic expressions, GLG significantly improves the accuracy of single-step theorem prediction from 67.1% to 84.7% when the beam size is set to 1. To explore the effectiveness of unsupervised-learning, we generate solutions with random search and train our model on it. Compared to supervised learning, the self-learning RL strat-

Method	Beam Size	Overall Acc(%)	Avg Time(s)	Avg Step
GeoDRL(GT)	1	84.7	2.67	2.27
	5	**89.4**	6.03	2.34
-w/o GLG	1	67.1	2.22	1.92
	5	86.9	4.21	2.37
-w/o RL	1	81.9	3.54	2.22
	5	88.4	8.56	2.37
-w/o TP	1	84.3	2.93	2.28
	5	88.2	6.53	2.33
-w/o Agent	N/A	86.5	5.83	9.58

Table 4: Results of ablation study on Geometry3K. -w/o GLG means replacing GLG with symbolic logical expression sequence. -w/o RL means replacing RL with supervised learning on solutions from random search. -w/o TP means removing the time penalty in the reward function. -w/o Agent means replacing DRL Agent with a heuristic search strategy.

egy significantly improves prediction efficiency: the average time is reduced by 24.6% (0.87s) and 29.6% (2.53s) for beam sizes of 1 and 5, respectively, and also an improvement in accuracy. This indicates that our designed RL strategy with time penalty contributes to learning more efficient solutions. With the heuristic search strategy from Inter-GPS (Lu et al., 2021), we obtain poor-quality solutions with numerous redundant steps, averaging as high as 9.58 steps. Additionally, increasing the beam size leads to an improvement in accuracy, but also increases the average solving time.

5.4 Discussion

Here we discuss the interpretability of our solution. Compared to probabilistic approaches, GeoDRL has explicit parsing and reasoning phases. The parser generates human-readable intermediate results and a visualizable graph. The reasoner provides a step-by-step deduction procedure with mathematical justifications. Unlike symbolic approaches, GeoDRL generates an explicit probability score for each theorem to search for a high-probability solution, providing explanations for every step.

By utilizing the beam search algorithm, based on state transition probabilities, GeoDRL is able to produce multiple solutions for a single geometry problem. Figure 4 shows an example. Providing various solutions is highly beneficial, particularly in online-education applications for students of varying skill levels, where we can tailor the solutions to the student's learning progress and knowledge base.

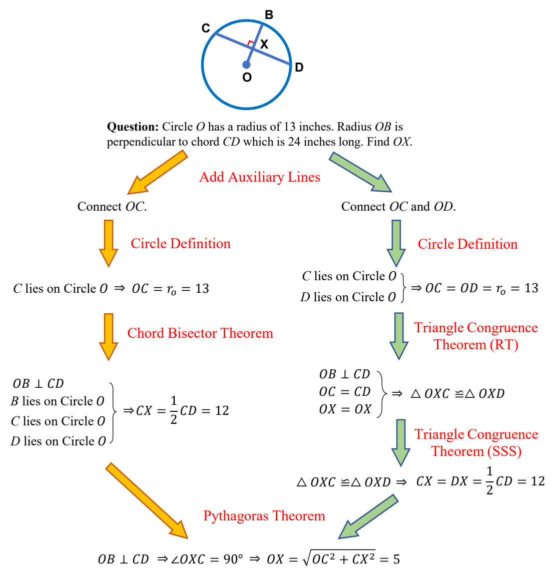

B
C
X
O
D
Question: Circle O has a radius of 13 inches. Radius OB is perpendicular to chord CD which is 24 inches long. Find OX.
Add Auxiliary Lines
Connect OC.
Connect OC and OD.
Circle Definition
Circle Definition
C lies on Circle O $\Rightarrow$ $OC = r_o = 13$
C lies on Circle O, D lies on Circle O $\Rightarrow OC = OD = r_o = 13$
Chord Bisector Theorem
Triangle Congruence Theorem (RT)
$OB \perp CD$, B lies on Circle O, C lies on Circle O, D lies on Circle O $\Rightarrow CX = \frac{1}{2}CD = 12$
$OB \perp CD$, $OC = CD$, $OX = OX$ $\Rightarrow \triangle OXC \cong \triangle OXD$
Triangle Congruence Theorem (SSS)
$\triangle OXC \cong \triangle OXD \Rightarrow CX = DX = \frac{1}{2}CD = 12$
Pythagoras Theorem
$OB \perp CD \Rightarrow \angle OXC = 90° \Rightarrow OX = \sqrt{OC^2 + CX^2} = 5$

Figure 4: Two available solutions generated by GeoDRL. The left one is more simple and concise, but for students who have not learned the chord bisector theorem, the right proof of triangle congruence will be easier to understand.

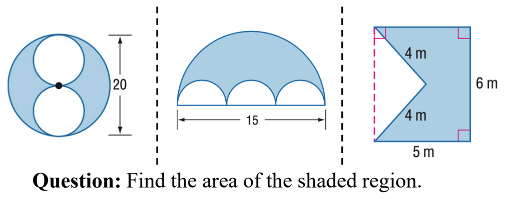


Figure 5: Failure examples of GeoDRL.

In some cases, the pipeline of parsing and reasoning can have negative impacts on problem-solving. The most common one is the shaded area problem, as shown in Figure 5. The key to solve such problems is to express the shaded area as the arithmetic results of regular shapes' areas, which requires joint efforts of parser and reasoner on the specific diagrams.

6 Conclusion and Future Work

In this paper, we propose GeoDRL, a self-learning geometry problem solving framework by integrating Deep Reinforcement Learning into deductive geometry reasoning, which enables unsupervised learning of problem-solving strategies. We structure geometry information as Geometry Logic Graph to glue discrete geometry literals together. The combination of neural network and symbolic system allows efficient solution while maintaining correctness. Experiments demonstrate that GeoDRL outperforms state-of-the-art approaches in terms of accuracy, efficiency and interpretability.

Our work presents a promising idea of integrating reinforcement learning into deductive reasoning for geometry problem solving. We believe that this framework is applicable to a majority of mathematical reasoning tasks. In the future, we intend to expand our framework to broader mathematical domains to become a general-purpose mathematical reasoning framework.

Limitations

1. Error accumulation: incorrect parsing results will affect the reasoner. In our experiments on Geometry3K dataset, incorrect parsing results lead to a substantial 21.0% performance drop.
2. The reasoner relies on manually predefined theorems, which limits its adaptability.
3. The random exploration in the RL training process leads to uncertainty in the rate of convergence.

Ethics Statement

In this research, We have made every effort to ensure that our study adheres to the ACM Ethical Principles. All data used in this study have been publicly available. This work may inspire the following research of mathematical reasoning. The potential risk of this work is that students may use the system as an auto-problem-solving tool to cheat on exams or assignments.

Acknowledgements

This work was supported by National Key R&D Program of China (No. 2021ZD0113301) (No. 2021ZD0113301).

References

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott M. Lundberg, Harsha Nori, Hamid Palangi, Marco Túlio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with GPT-4. *CoRR*, abs/2303.12712.

Jiaqi Chen, Tong Li, Jinghui Qin, Pan Lu, Liang Lin, Chongyu Chen, and Xiaodan Liang. 2022. Unigeo: Unifying geometry logical reasoning via reformulating mathematical expression. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022*, pages 3313–3323.

Jiaqi Chen, Jianheng Tang, Jinghui Qin, Xiaodan Liang, Lingbo Liu, Eric P. Xing, and Liang Lin. 2021. Geoqa: A geometric question answering benchmark towards multimodal numerical reasoning. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021*, pages 513–523.

Kaiyu Chen, Yihan Dong, Xipeng Qiu, and Zitian Chen. 2018. Neural arithmetic expression calculator. *arXiv preprint arXiv:1809.08590*.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019*, volume 1, pages 4171–4186.

Herbert Gelernter, James R Hansen, and Donald W Loveland. 1960. 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.

Herbert L. Gelernter. 1959. Realization of a geometry theorem proving machine. In *Information Processing, Proceedings of the 1st International Conference on Information Processing, UNESCO 1959*, pages 273–281.

Ross Girshick. 2015. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448.

Cezary Kaliszyk, Josef Urban, Henryk Michalewski, and Miroslav Olsák. 2018. Reinforcement learning of theorem proving. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018*, pages 8836–8847.

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

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019*.

Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. 2021. Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning. In

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, volume 1, pages 6774–6786.

Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. 2022a. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. *CoRR*, abs/2209.14610.

Pan Lu, Liang Qiu, Wenhao Yu, Sean Welleck, and Kai-Wei Chang. 2022b. A survey of deep learning for mathematical reasoning. *arXiv preprint arXiv:2212.10535*.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Arthur J. Nevins. 1975. Plane geometry theorem proving using forward chaining. *Artif. Intell.*, 6(1):1–23.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. 2018. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI 2018*, pages 3942–3951.

Gabriel Poesia, Wenxin Dong, and Noah D. Goodman. 2021. Contrastive reinforcement learning of symbolic reasoning domains. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021*, pages 15946–15956.

Martin L Puterman. 2014. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Mrinmaya Sachan and Eric Xing. 2017. 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.

Minjoon Seo, Hannaneh Hajishirzi, Ali Farhadi, Oren Etzioni, and Clint Malcolm. 2015. Solving geometry problems: Combining text and diagram interpretation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1466–1476.

Lei Wang, Dongxiang Zhang, Lianli Gao, Jingkuan Song, Long Guo, and Heng Tao Shen. 2018. Mathdqn: Solving arithmetic word problems via deep reinforcement learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI 2018*, pages 5545–5552.

Wu Wen-Tsun. 1986. Basic principles of mechanical theorem proving in elementary geometries. *Journal of automated Reasoning*, 2(3):221–252.

Ming-Liang Zhang, Fei Yin, Yi-Han Hao, and Cheng-Lin Liu. 2022. Plane geometry diagram parsing. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022*, pages 1636–1643.

Ming-Liang Zhang, Fei Yin, and Cheng-Lin Liu. 2023. A multi-modal neural geometric solver with textual clauses parsed from diagram. *CoRR*, abs/2302.11097.

A Experiment Details

The number of parameters of Graph Transformer model is around 44,000,000. The model is trained on the train set of Geometry3K dataset for 50000 steps using the AdamW optimizer (Loshchilov and Hutter, 2019), with a batch size of 32 and a learning rate of 3e-5. It costs around 90 minutes to train 1000 steps on 4 Nvidia Titan X GPUs. In reinforcement training, the parameters are set as $\epsilon = 0.1, \gamma = 0.5$ and $\alpha = 1.0, \sigma = 60$ for the reward function. In the inference stage, the time limit to solve one problem is set to 300s.

B Implementation Details

Primitive	Attribute
Point	N/A
Line	Length
Angle	AngleMeasure
Arc	ArcMeasure
Circle	RadiusLength
Triangle	Area
Polygon	Area

Table 5: The types of primitive and attribute.

Edge Type	Domain
EndPointOf(R)	<Point, Line>
LiesOnLine(R)	<Point, Line>
VertexOfAngle(R)	<Point, Angle>
SidePoint(R)	<Point, Angle>
CenterOfArc(R)	<Point, Arc>
EndPointOfArc(R)	<Point, Arc>
CenterOf(R)	<Point, Circle>
LiesOnCircle(R)	<Point, Circle>
Vertex(R)	<Point, Triangle/Polygon>
Equal	<Line, Line>
Parallel	<Line, Line>
Perpendicular	<Line, Line>
SideOfAngle(R)	<Line, Angle>
SideOf(R)	<Line, Triangle/Polygon>
RadiusOf(R)	<Line, Circle>
Equal	<Angle, Angle>
InteriorAngleOf(R)	<Angle, Triangle/Polygon>
Equal	<Arc, Arc>
Congruent	<Triangle, Triangle>
Similar	<Triangle, Triangle>
Similar	<Polygon, Polygon>

Table 6: The edge types between primitives in GLG. (R) denotes there is an inverse edge type of it.

Class	Name
Circle	Circle Definition
	Thales' Theorem
	Inscribed Angle Theorem
	Tangent Secant Theorem
	Chord Theorem
	(Chord Bisector)
	(Intersecting Chord)
Parallel	Parallel Lines Theorem
	(Alternate Interior Angles)
	(Alternate Exterior Angles)
	(Consecutive Interior Angles)
	(Consecutive Exterior Angles)
	(Corresponding Angles)
Single Triangle	Triangle Angle-Sum Theorem
	(Interior Angles)
	(Exterior Angles)
	Isosceles Triangle Theorem - Angle
	Isosceles Triangle Theorem - Side
	Equilateral Triangle Theorem
	Pythagoras Theorem
	Triangle Center of Gravity Theorem
	Angle Bisector Theorem
	Law of Sines
	Law of Cosines
Double Triangles	Triangle Congruence Theorem (Apply)
	Triangle Congruence Theorem (Prove)
	(SSS)
	(SAS)
	(ASA)
	(AAS)
	(HL)
	Triangle Similarity Theorem (Apply)
	Triangle Similarity Theorem (Prove)
	(AA)
	(SSS)
	(SAS)
Polygon	Polygon Similarity Theorem
	Median Line Theorem
	Area Equation
	Polygon Angle-Sum Theorem
Auxiliary Line	Connect Points

Table 7: Predefined geometry theorems in the symbolic system. Theorems in brackets are the sub-theorems.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 5.4 and Section Limitations

☑ A2. Did you discuss any potential risks of your work?
Section Ethics Statement

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Section 1

☑ A4. Have you used AI writing assistants when working on this paper?
ChatGPT. Rephrasing and Polishing. All sections.

B ☑ Did you use or create scientific artifacts?

Section 4

☑ B1. Did you cite the creators of artifacts you used?
Section 4

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Section 4

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Section 4

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
The public dataset used in our study is already checked and anonymized by previous work "Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning" and "Solving geometry problems: Combining text and diagram interpretation".

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Section 5.1

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 5.1

C ☑ Did you run computational experiments?

Section 5

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 5.1 Appendix A

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 5.1 Appendix A

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 5

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section 5

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.