Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations

Peiyi Wang[1†], Lei Li[3], Zhihong Shao[4], R.X. Xu[2], Damai Dai[1], Yifei Li[5], Deli Chen[2], Y. Wu[2], Zhifang Sui[1]

[1]National Key Laboratory for Multimedia Information Processing, Peking University
[2]DeepSeek-AI, [3]The University of Hong Kong
[4]Tsinghua University, [5]The Ohio State University

Presenter: Min-Han Ye, Ke Tan

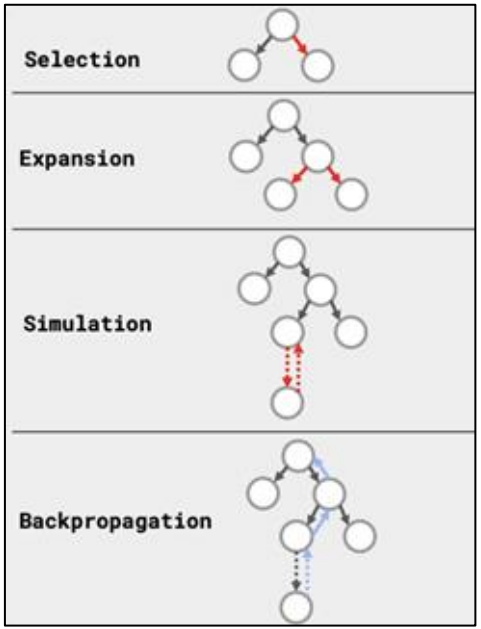
Seminar on „ Process Reward Modeling in LLMs “
Heidelberg 30.04.2026

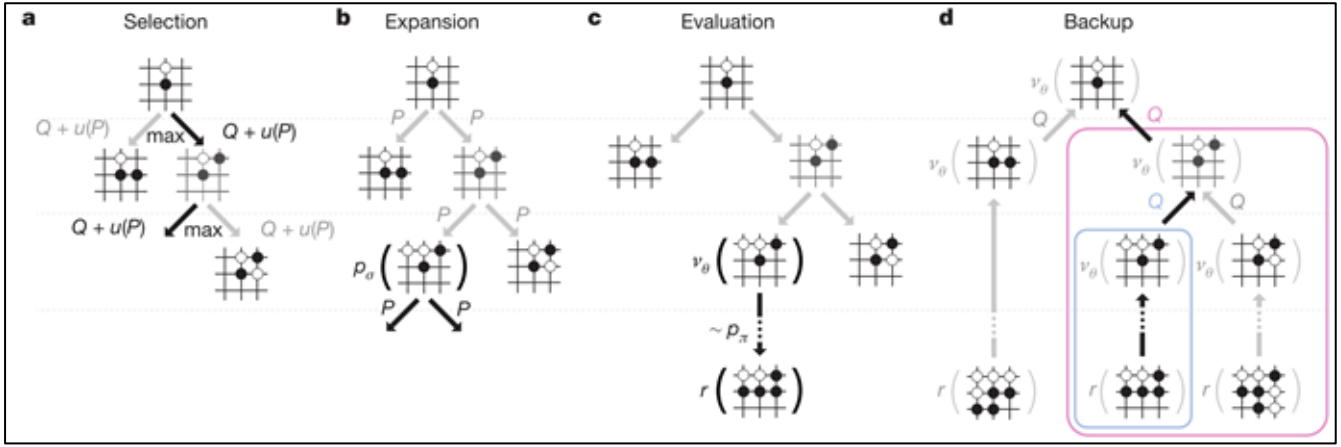
Introduction

- LLMs have demonstrated remarkable capabilities across various tasks
- Complex multi-step mathematical reasoning problems
  - pretraining (Azerbayev et al., 2023),
  - fine-tuning (Luo et al., 2023; Yu et al., 2023b; Wang et al., 2023b)
  - prompting (Wei et al., 2022; Fu et al., 2022)
  - verification (Wang et al., 2023d; Li et al., 2023b; Zhu et al., 2023; Leviathan et al., 2023).
    - accuracy, consistency, feedback
- Outcome Reward Model (ORM) vs Process Reward Model (PRM)
- Automatic process annotation framework.
  - Inspired by Monte Carlo Tree Search

Monte Carlo Tree Search (MCTS)

1. Selection: choose a move to explore.
2. Expansion: add possible next moves.
3. Simulation: estimate results through rollouts.
4. Backpropagation: update move values using outcomes.





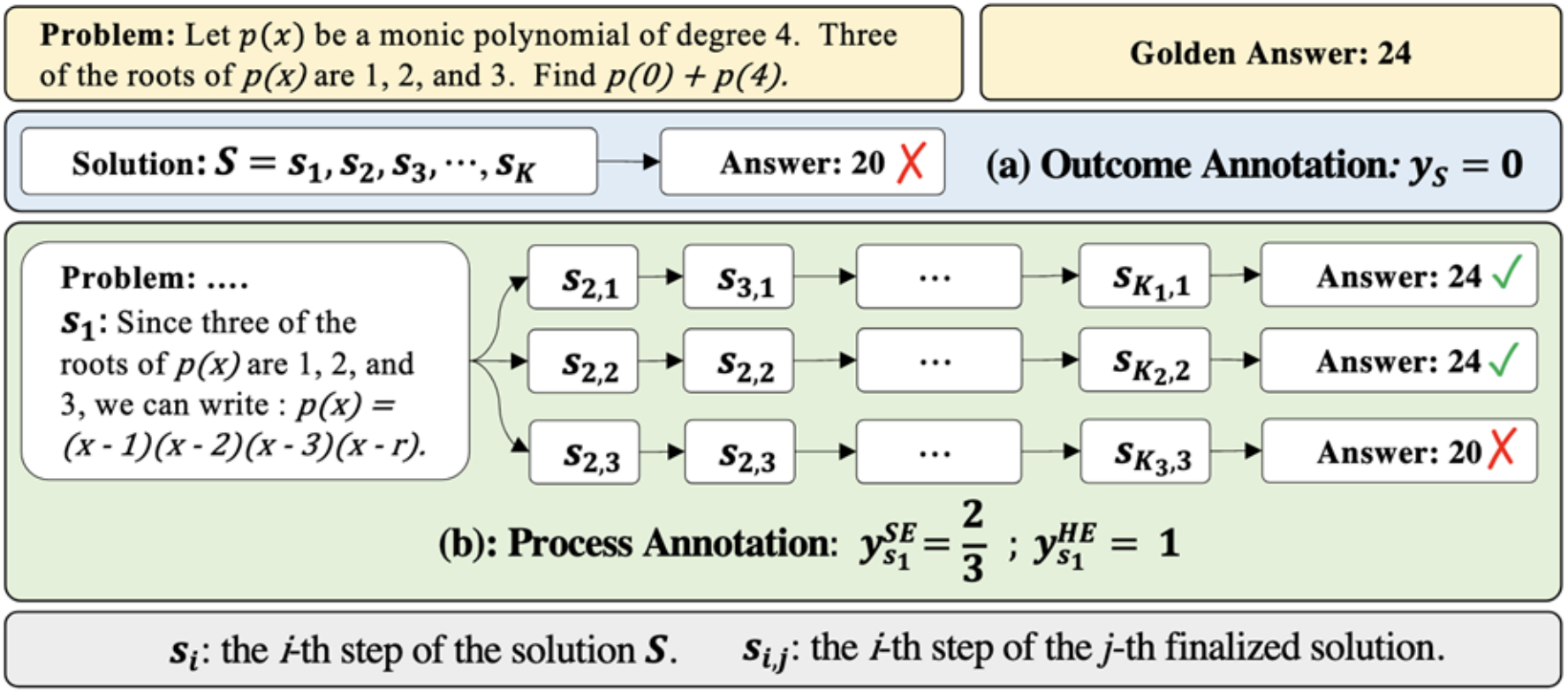
Method

- Automatic Process Annotation
  - Def: The quality of a reasoning step is its potential to deduce the correct answer.
  - Completion (completer)
  - Estimation
    - hard estimation (HE)
    - soft estimation (SE)

$$y_{s_i}^{HE} = \begin{cases} 1 & \exists a_j \in A, a_j = a^* \\ 0 & \text{Otherwise} \end{cases}$$

$$y_{s_i}^{SE} = \frac{\sum_{j=1}^{N} \mathbb{I}(a_j = a^*)}{N}.$$

Method


Problem: Let $p(x)$ be a monic polynomial of degree 4. Three of the roots of $p(x)$ are 1, 2, and 3. Find $p(0) + p(4)$.
Golden Answer: 24
Solution: $S = s_1, s_2, s_3, \cdots, s_K$
Answer: 20
(a) Outcome Annotation: $y_S = 0$
Problem: ....
s_1: Since three of the roots of $p(x)$ are 1, 2, and 3, we can write : $p(x) = (x - 1)(x - 2)(x - 3)(x - r)$.
$s_{2,1}$ $s_{3,1}$ ⋯ $s_{K_1,1}$ Answer: 24
$s_{2,2}$ $s_{2,2}$ ⋯ $s_{K_2,2}$ Answer: 24
$s_{2,3}$ $s_{2,3}$ ⋯ $s_{K_3,3}$ Answer: 20
(b): Process Annotation: $y_{s_1}^{SE} = \frac{2}{3}$; $y_{s_1}^{HE} = 1$
s_i: the i-th step of the solution S. $s_{i,j}$: the i-th step of the j-th finalized solution.

Method

- Automatic Process Annotation
  - The quality of a reasoning step is its potential to deduce the correct answer.
  - Completion (completer)
  - Estimation
    - hard estimation (HE)
    - soft estimation (SE)

$$y_{s_i}^{HE} = \begin{cases} 1 & \exists a_j \in A, a_j = a^* \\ 0 & \text{Otherwise} \end{cases}$$

$$y_{s_i}^{SE} = \frac{\sum_{j=1}^{N} \mathbb{I}(a_j = a^*)}{N}.$$

- Ranking for Verification

$$a_{sc+rm} = \arg\max_{a} \sum_{i=1}^{N} \mathbb{I}(a_i = a) \cdot RM(p, S_i).$$

- Reinforcement Learning with Process Supervision

Workflow

Stage 1: Prepare math LLMs

- Generators & Completers
- LLaMA27B/13B/70B, LLemma-7B/34B,
- Mistral-7B, DeepSeek-67B
- 3 epochs on MetaMATH

Stage 2: Build PRM training data

- Generators (7B/13B) produce 15 solutions
- Completer (LLemma-7B) takes each step and "rolls out" N=8 completions.
- Labeling via Hard Estimation

Stage 3: Train PRM / MATH-SHEPHERD

- Question + step-wise solution + automatic step labels
- train PRM / MATH-SHEPHERD

Stage 4: Use PRM / MATH-SHEPHERD

- Verification: (LLaMA2-70B and LLemma-34B)
  - generate 256 solutions
  - MATH-SHEPHERD scores them
  - select best answer
- Reinforcement learning: (Mistral-7B)
  - generate solution (LLama2-7B and Mistral-7B)
  - MATH-SHEPHERD (Mistral-7B)
  - gives step-level rewards
  - updates the generator’s wights

Experiment

Goal: evaluate whether Math-Shepherd works as both **an inference-time verifier** and **a training-time reward model**.

Evaluation Scenarios

Verification
Reinforcement Learning

Benchmarks

GSM8K
MATH

Math-Shepherd as Verifier

Test whether Math-Shepherd can rerank and select the best solution from generated candidates.

Models	Verifiers	GSM8K	MATH500
LLaMA2-70B: MetaMATH	Self-Consistency	88.0	39.4
	ORM	91.8	40.4
	Self-Consistency + ORM	92.0	42.0
	MATH-SHEPHERD (Ours)	93.2	44.5
	Self-Consistency + MATH-SHEPHERD (Ours)	92.4	45.2
LLemma-34B: MetaMATH	Self-Consistency	82.6	44.2
	ORM	90.0	43.7
	Self-Consistency + ORM	89.6	45.4
	MATH-SHEPHERD (Ours)	90.9	46.0
	Self-Consistency + MATH-SHEPHERD (Ours)	89.7	47.3
DeepSeek-67B: MetaMATH	Self-Consistency	88.2	45.4
	ORM	92.6	45.3
	Self-Consistency + ORM	92.4	47.0
	MATH-SHEPHERD (Ours)	**93.3**	47.0
	Self-Consistency + MATH-SHEPHERD (Ours)	92.5	**48.1**

Math-Shepherd as Reward Model for Reinforcement Learning

Test whether step-level rewards from Math-Shepherd can improve the generator through PPO.

Models	GSM8K	MATH
LLaMA2-7B: MetaMATH	66.6	19.2
+ RFT	68.5	19.9
+ ORM-PPO	70.8	20.8
+ MATH-SHEPHERD-step-by-step-PPO (Ours)	73.2	21.6
Mistral-7B: MetaMATH	77.9	28.6
+ RFT	79.0	29.9
+ ORM-PPO	81.8	31.3
+ MATH-SHEPHERD-step-by-step-PPO (Ours)	**84.1**	**33.0**

Math-Shepherd as both Reward Model and Verifier

Test whether using Math-Shepherd for both training and inference gives further improvement.

Models	Verifiers	GSM8K	MATH500
Mistral-7B: MetaMATH	Self-Consistency	83.9	35.1
	ORM	86.2	36.4
	Self-Consistency + ORM	86.6	38.0
	MATH-SHEPHERD (Ours)	87.1	37.3
	Self-Consistency + MATH-SHEPHERD (Ours)	86.3	38.3
Mistral-7B: MetaMATH +step-by-step PPO (Ours)	Self-Consistency	87.4	42.3
	ORM	87.6	41.3
	Self-Consistency + ORM	89.0	43.1
	MATH-SHEPHERD (Ours)	88.4	41.1
	Self-Consistency + MATH-SHEPHERD (Ours)	**89.1**	**43.5**

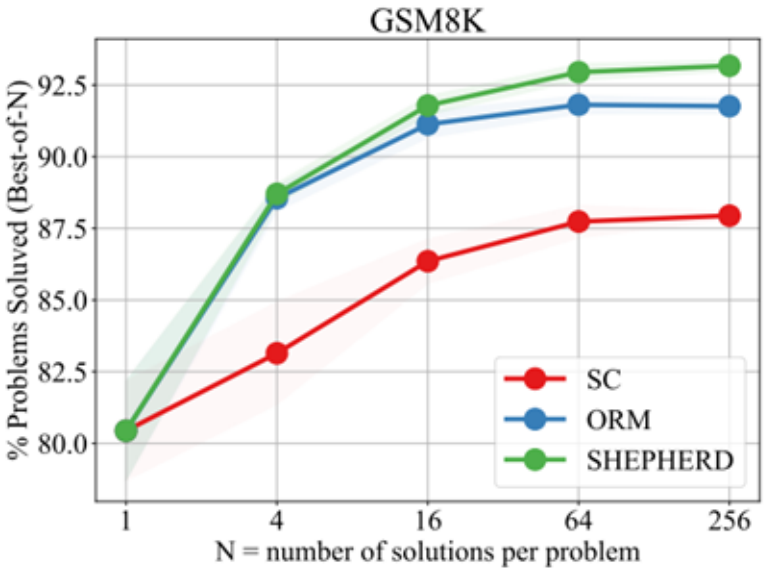
Analysis

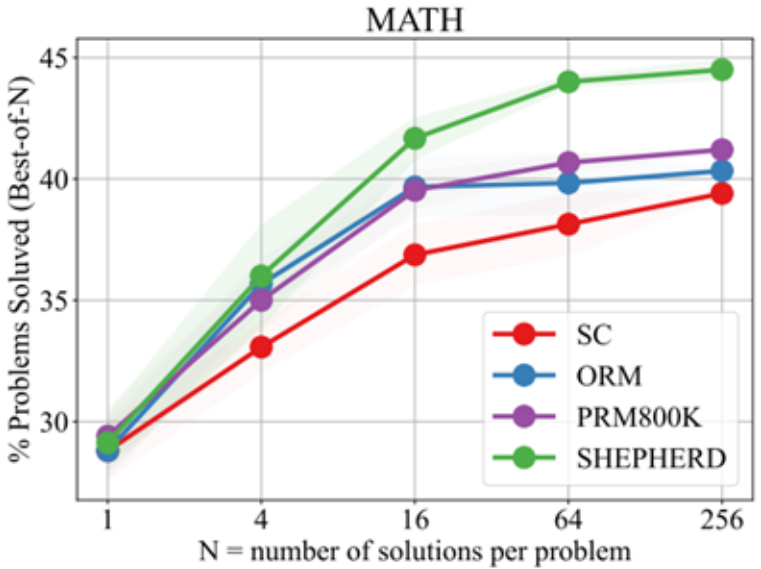
Goal: understand what factors affect PRM performance.

- candidate solutions
- automatic annotation quality
- base model
- training data size
- out-of-distribution generalization

Numbers of Solution Candidates

Study how verification performance changes as the number of candidate solutions increases.





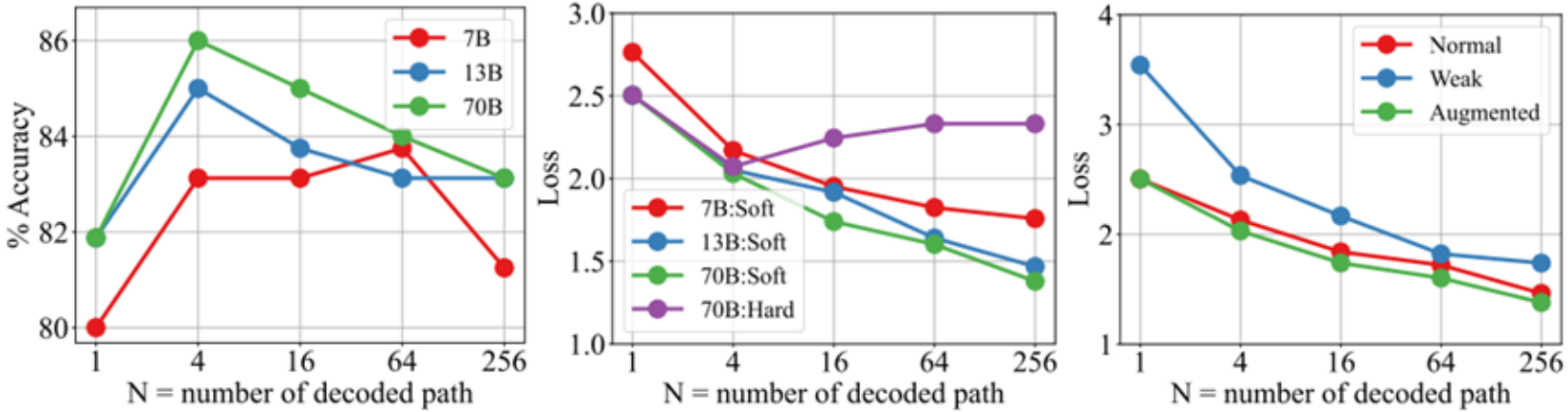
Quality of the Automatic Process Annotation

Evaluate whether the automatic step labels are reliable enough to train a PRM.

Methods	Models	Accuracy (%)	Loss
DIVERSE-NLI (Li et al., 2023b)	DeBERTa (He et al., 2020)	61.3	5.43
DIVERSE-NLI (Li et al., 2023b)	LLaMA2-13B	75.6	3.27
DIVERSE-Rule (Li et al., 2023b)	-	75.0	3.43
MATH-SHEPHERD	LLaMA2-13B (N = 4)	85.0	2.05

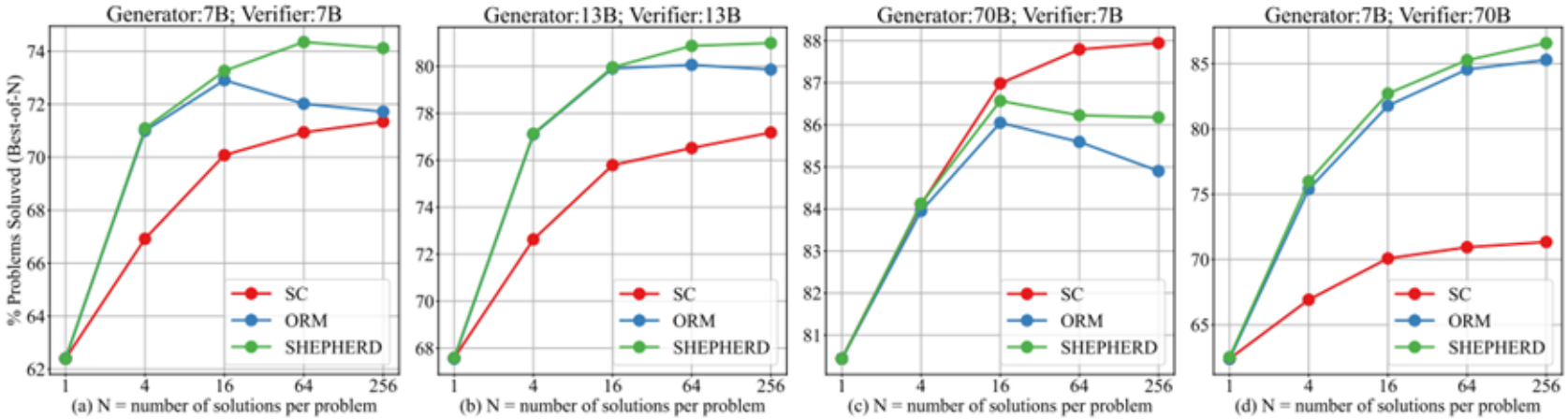
Quality of the Automatic Process Annotation

Completer size: larger models produce better labels
Estimation method: soft labels are more stable than hard labels
Data quality: weak data hurts PRM training



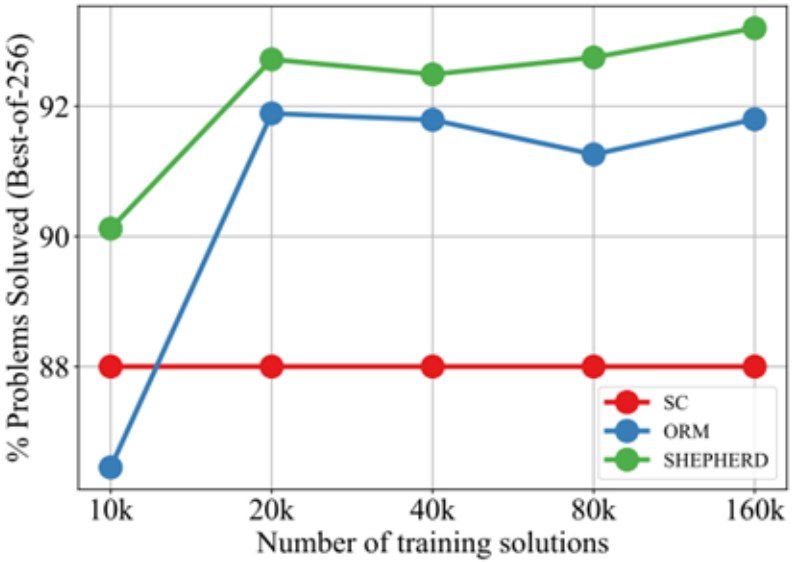
Influence of the Pre-trained Base Models

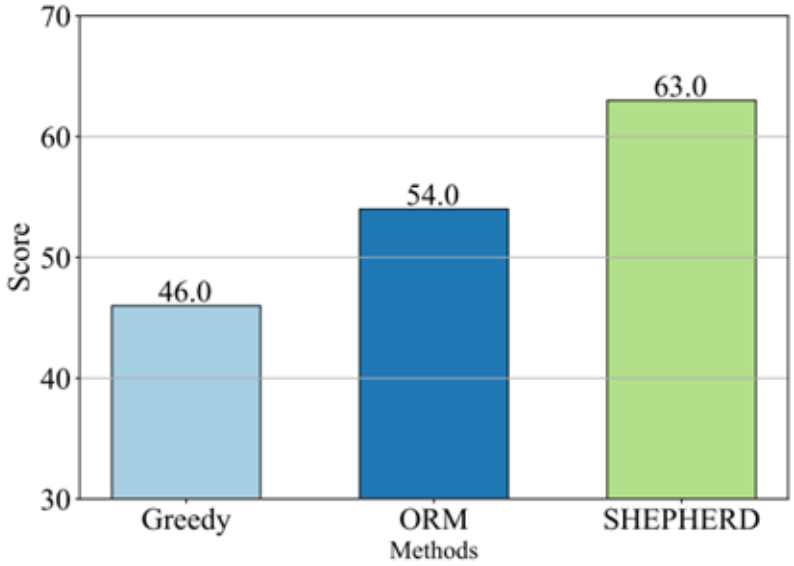
Compare different backbone models to see how model choices affects PRM performance.



Influence of the Number of Data

Study how much training data is needed to train an effective Math-Shepherd model.





Out-of-distribution Performance

Test whether Math-Shepherd can generalize to math problems outside the original benchmarks.

Question	*The first term of an arithmetic sequence is 18. The sum of the first six terms of the sequence is equal to the sum of the first seven terms. Show that the sum of the first 13 terms of the sequence is 0 and calculate the 13th term of the sequence.*
MATH-SHEPHERD	Step 1: Let the common difference of the arithmetic sequence be d. MATH-SHEPHERD: 0.99. Step 2: The sum of the first six terms of the sequence is $18 + (18 + d) + (18 + 2d) + (18 + 3d) + (18 + 4d) + (18 + 5d) = 108 + 30d$. MATH-SHEPHERD: 0.85. ..... Step 6: Therefore, $d = -3$. MATH-SHEPHERD: 0.85. ..... Step 8: The 13th term of the sequence is $18 - 36 = -18$. MATH-SHEPHERD: 0.93. ORM Score: 0.73.
ORM	Step 1: Let the common difference of the arithmetic sequence be d. MATH-SHEPHERD: 0.99. Step 2: The sum of the first six terms is $18 + (18 + d) + (18 + 2d) + (18 + 3d) + (18 + 4d) + (18 + 5d) = 90 + 15d$. MATH-SHEPHERD: 0.53. ..... Step 6: Dividing by -6, we find that $d = -2$. MATH-SHEPHERD: 0.38. ..... Step 9: The 13th term of the sequence is $18 - 26 = -8$. MATH-SHEPHERD: 0.38. ORM Score: 0.84.

Overall Results

- Math-Shepherd improves both verification and reinforcement learning.
- Process reward model is stronger than self-consistency and outcome reward models.

- Performance improves with **more candidates**, **better annotations**, **stronger matched models**, and **more data**.
- Math-Shepherd shows promising out-of-distribution generalization.

Limitations

- The **computational cost** of the completion process.
- The automatic process annotation consists of **noise**.

Takeaways

- **Math-Shepherd** shows that process supervision can be built without human step annotations.

- As a **verifier**, it selects better solutions at inference time.
- As a **reward model**, it improves LLM training through step-level rewards.
- Combined RL + verification gives stronger results.