OPENSIR: OPEN-ENDED SELF-IMPROVING REASONER

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-play promises to advance LLM reasoning without human-annotated data, yet existing methods lack open-ended learning and yield only marginal or even negative gains on post-trained models. We introduce **Open**-Ended **S**elf-**I**mproving **R**easoner (OpenSIR), a self-play framework where a single LLM alternates teacher and student roles to generate and solve novel problems without external verifiers or annotated data. From a single seed, OpenSIR sustains open-ended exploration via diversity rewards pushing toward unfamiliar concepts and difficulty calibration keeping problems learnable. Across seven math benchmarks, OpenSIR consistently improves all models, averaging +3.6 on instruction models and +3.1 on reasoning models, outperforming recent self-play baselines. Without any annotated data, it surpasses GRPO baselines trained on over 7K examples. Despite training only on self-generated math, OpenSIR is the only self-play method that transfers to general reasoning, improving reasoning models by at least +4.4 points across three benchmarks. Our analysis shows that, with diversity rewards, generated problems progress from basic arithmetic and algebra to advanced topics such as calculus and optimisation, reaching regions of the problem space beyond human-curated datasets. Without diversity rewards, problems concentrate in a narrow region of this space and accuracy falls on both mathematical and general reasoning. Difficulty calibration depends on teacher-student co-evolution: freezing the teacher destabilises problem difficulty and lowers accuracy.

1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) drives recent advances in LLM reasoning. Recent works on DeepSeek-R1 (DeepSeek-AI et al., 2025) and OpenAI o1 (OpenAI, 2024) have shown that large-scale reinforcement learning improves reasoning capabilities. Yet, these methods require extensive human-annotated data for reward signals, which bottleneck scalability and potentially limit performance to human-level (Hughes et al., 2024b). One promising direction to address these fundamental limitations is to generate synthetic training data through self-play, which demonstrated remarkable success in various games (Silver et al., 2016; 2017; Brown & Sandholm, 2019; FAIR et al., 2022), allowing systems to exceed human-level performance by learning from unambiguous reward signals (Silver et al., 2017; FAIR et al., 2022).

Applying self-play to mathematical reasoning reveals a fundamental obstacle: unlike games with clear rules and winners, generated mathematics problems lack ground-truth answers to provide feedback signals. Existing approaches work around this by importing external verifiers, such as compilers (Pourcel et al., 2024; Zhao et al., 2025) or game rules (Liu et al., 2025), or by relying on heuristic agreement signals such as majority voting with repetition penalties, as in R-Zero (Huang et al., 2025). Absolute Zero (Zhao et al., 2025) and R-Zero (Huang et al., 2025) take this approach furthest, as both generate and solve problems within a single framework without relying on external verifiers. Yet neither was evaluated on post-trained models, the very setting where self-play is most needed to push beyond what human-curated data has already provided. When applied to post-trained models, their gains are typically marginal and can even turn negative (§3.2). This failure points to a deeper limitation: neither method supports open-ended learning (Bauer et al., 2023; Hughes et al., 2024a)—the capacity to continuously generate and pursue genuinely novel challenges without external supervision. Without it, the teacher collapses toward familiar problem templates and ceases to produce new learning signals, rendering self-play ineffective for models that have already mastered human-curated data.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

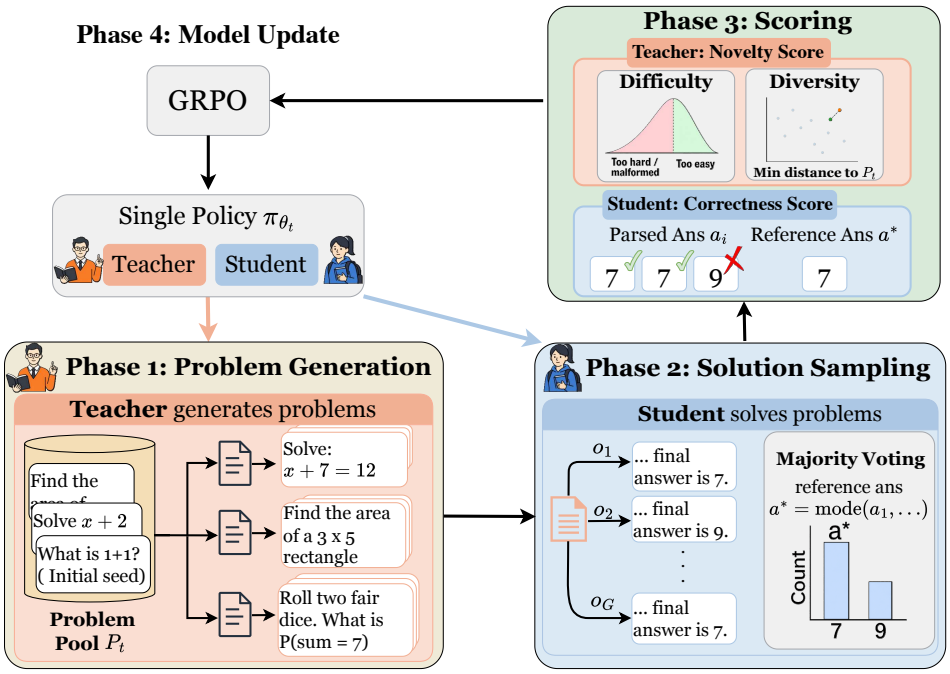

Figure 1: Overview of the OpenSIR framework. A single policy π_θ alternates between generating and solving novel problems without external supervision. Each training iteration consists of **problem generation**, **solution sampling**, **scoring**, and **model update**. Novelty is captured through both *difficulty* and *diversity*: problems must be challenging yet solvable, and they must explore new concepts. These dimensions together drive open-ended self-improvement in the LLM reasoning ability.

To address this limitation, we present **Open**-ended **S**elf-**I**mproving **R**easoner (OpenSIR), a self-play framework in which a single policy π_θ alternates between teacher and student roles to continuously generate and solve novel problems without external supervision. OpenSIR takes a concrete step toward open-ended learning (Bauer et al., 2023; Hughes et al., 2024a) by sustaining novelty-driven self-improvement from a single trivial seed and building broadly transferable reasoning skills (§3.2). OpenSIR captures novelty through two reward components. A *difficulty* reward, computed from consistency and solution length across multiple solution attempts, selects problems that are appropriately challenging, while an embedding-based *diversity* reward promotes problems that explore semantically new mathematical concepts. Together, these rewards continually expand the model's capacity, sustaining the novel learning signal that prior self-play methods fail to maintain.

Across seven math and three general reasoning benchmarks, existing self-play methods yield negligible gains on instruction models (up to $+0.87$) and actively degrade reasoning models (up to -1.93). OpenSIR consistently improves all models, averaging $+3.35$ on maths and up to $+4.79$ on general reasoning, highlighting the need for novelty to sustain learning in well post-trained models. Starting from a single trivial seed problem, these results surpass even GRPO baselines trained on over 7,000 human-annotated examples. Our analysis shows that the diversity reward is central to this behaviour: it nearly doubles concept coverage and helps the gains transfer beyond math to general reasoning. We also find that teacher and student must co-evolve to keep problem difficulty calibrated, while the solvability reward prevents the model from chasing harder but increasingly invalid problems that hurt learning.

Our contributions are as follows: (1) We show that existing self-play methods fail on well post-trained models and that problem diversity is critical for sustained improvement. (2) We propose OpenSIR, the first self-play framework to enable open-ended self-improvement by jointly optimising problem difficulty and diversity, bootstrapping from a single trivial seed without external supervision. (3) We demonstrate that OpenSIR succeeds where prior self-play methods fail, reliably improving well

post-trained models with gains that transfer to general reasoning and outperform GRPO baselines trained on over 7,000 annotated examples.

2 OPEN-ENDED SELF-IMPROVING REASONER

Figure 1 illustrates the Open-Ended Self-Improving Reasoner (OpenSIR), a self-play framework in which a policy π_θ learns to both generate and solve novel mathematical problems without external supervision. We use reinforcement learning to optimise two roles within one policy: the *teacher*, which creates new problems, and the *student*, which solves them. This open-ended approach enables the policy to bootstrap its learning and discover new and diverse challenges without annotated data. Each training iteration involves four phases (§1):

1. **Problem generation** (§2.1): The teacher proposes new problems by conditioning on reference problems from an accumulated pool of previously generated problems;
2. **Solution sampling** (§2.2): The student attempts multiple solutions per problem, with majority voting determining the reference answer and solve rate measuring reliability;
3. **Scoring** (§2.3): We compute novelty scores for the teacher's generated problems and correctness scores for the student's solutions; and
4. **Model update** (§2.4): We update the policy's parameters with role-specific rewards using the problem-solution pairs selected by the novelty scores.

In OpenSIR, we define novelty along two dimensions that together drive continuous open-ended learning. First, problems must have an appropriate level of difficulty. It should be challenging enough to promote learning but solvable enough to provide reliable training signals. Second, problems must explore diverse concepts, preventing the model from repeating learning on familiar concepts. This two-dimensional view of novelty ensures the model continuously expands both the depth and breadth of its mathematical reasoning abilities.

2.1 PROBLEM GENERATION

At each iteration t, the policy π_θ generates k groups of G problems each, denoted as $q_{1:G}$ within each group, for a total of $M = k \times G$ problems. To generate these problems, we sample k reference problems from a pool $\mathcal{P}_{t-1}$ of accumulated problems from previous iterations, where each reference problem serves as a seed for generating G new problems. Each generated problem must explicitly include the mathematical concepts required for its solution. Problems with invalid formats are filtered out, and valid problems proceed to the solution-sampling phase. We initialise the problem pool $\mathcal{P}_0$ with a single trivial problem ("What is 1+1?").

2.2 SOLUTION SAMPLING

Let a_j denote the parsed answer from solution attempt o_j. We select the most common answer across attempts as the reference answer a^*. We then compute the *solve rate* for each problem to determine the reliability of the answers. For brevity, we denote $s_{q_i} = \text{SolveRate}(q_i)$ when referring to the solve rate of problem q_i.

$$\begin{aligned} \text{SolveRate}(q_i) &= \frac{\text{count}(a^*)}{G}, \\ a^* &= \arg\max_{a \in a_{1:G}} \text{count}(a). \end{aligned} \tag{1}$$

In Equation (1), $\text{count}(a)$ denotes the number of times answer a appears. The solve rate quantifies answer reliability. High solve rates indicate reliable reference answers due to solution convergence, while low solve rates suggest inconsistent solutions that may indicate flawed problem formulations.

2.3 SCORING

We evaluate the quality of generated problems and solutions with different scoring functions. The teacher's problems are scored based on *difficulty* and *diversity*, while the student receives scores for *correctness*. Additionally, both roles incorporate format scores to ensure parseable outputs.

2.3.1 TEACHER SCORING

We capture novelty through two fundamental dimensions: difficulty and diversity. We measure difficulty using *solvability* to ensure problems remain appropriately challenging and *solution length* to encourage multi-step reasoning, as these provide complementary signals about problem difficulty. Diversity is promoted through embedding distance, which encourages exploration of varied mathematical concepts. These components form a unified novelty score that guides problem generation.

Solvability ($\text{score}_{\text{sol}}$). The solvability score identifies problems with appropriate challenge. We use solve rate as a proxy for solvability—problems with $s_{q_i} > s_{\max}$ are likely too easy, while those with $s_{q_i} < s_{\min}$ are either too difficult or malformed. We employ a triangular scoring function that peaks at the optimal solve rate and decreases linearly as problems become too easy or too hard.

We define the solve rate range as $[s_{\min}, s_{\max}]$. Easy problems ($s_{q_i} > s_{\max}$) fail to challenge the model, while problems that are too hard or malformed ($s_{q_i} < s_{\min}$) offer minimal training value.

Formally, for $s_{q_i} \in [0, 1]$, let $s_{\text{mid}} = (s_{\min} + s_{\max})/2$ be the midpoint:

$$\text{score}_{\text{sol}}(q_i) = \begin{cases} 1 - \eta |s_{q_i} - s_{\text{mid}}| & \text{if } s_{q_i} \in [s_{\min}, s_{\max}] \\ 0 & \text{otherwise,} \end{cases}$$

where $\eta = (1 - 1/G) / (s_{\text{mid}} - s_{\min})$ is the slope coefficient, with G being the number of solution attempts. The score peaks at the midpoint s_{mid} and decreases to $1/G$ at the boundaries. This creates a symmetric triangular score centred at the midpoint of the solve rate range, giving a maximum score for problems with moderate difficulty and progressively less score as the solve rate approaches either boundary.

Solution Length ($\text{score}_{\text{len}}$). Solution length complements solvability by measuring problem complexity. Problems requiring multi-step reasoning typically elicit longer solutions. We score problems using the average length of student solutions:

$$\text{score}_{\text{len}}(q_i) = \min\left(\frac{\bar{l}(q_i)}{l_{\text{base}}}, \frac{l_{\text{cap}}}{l_{\text{base}}}\right), \tag{2}$$

where $\bar{l}(q_i)$ denotes average solution length for problem q_i, l_{base} is a normalisation factor (defaults to 1000 tokens), and l_{cap} prevents outliers from dominating the scoring signal. This score complements the solvability score (see Appendix E.1).

Diversity ($\text{score}_{\text{div}}$). We compute the semantic distance between each new problem and the existing problem pool:

$$\text{score}_{\text{div}}(q_i) = \min_{q' \in \mathcal{P}_{t-1}} d(e_{q_i}, e_{q'}), \tag{3}$$

where e_{q_i} and $e_{q'}$ represent problem embeddings obtained from a pre-trained encoder, and $d(\cdot, \cdot)$ denotes cosine distance. This score maximises when a problem is semantically distant from all existing problems in the pool.

Format ($\text{score}^T_{\text{fom}}$). The format score ensures proper problem structure. Generated problems must be enclosed in <question> tags with concepts listed in <concepts> tags (maximum three concepts). We assign $\text{score}^T_{\text{fom}}(q_i) = 1$ for correct formatting and $\text{score}^T_{\text{fom}}(q_i) = 0$ otherwise.

Novelty Score. We combine these components into a novelty score capturing both difficulty and diversity:

$$\begin{aligned} \text{score}_{\text{novel}}(q_i) = \alpha\, \text{score}_{\text{sol}}(q_i) + \lambda\, \text{score}_{\text{len}}(q_i) \\ + \gamma\, \text{score}_{\text{div}}(q_i) + \delta\, \text{score}^T_{\text{fom}}(q_i), \end{aligned} \tag{4}$$

where $\alpha, \lambda, \gamma, \delta$ are hyperparameters controlling the relative importance of each component. This novelty score is used to select high-quality problem-solution pairs for training.

2.3.2 STUDENT SCORING

The student's score is based on solution correctness. For each solution attempt, we evaluate correctness by comparing the parsed answer against the reference answer from majority voting.

Format ($\text{score}^S_{\text{fom}}$). The format score ensures proper answer presentation. Solutions must present final answers in \boxed{} notation. We assign $\text{score}^S_{\text{fom}}(o_j) = 1$ for correct formatting and 0 otherwise.

Correctness Score. The student's correctness score combines accuracy with the format score:

$$\begin{aligned}&\text{score}_{\text{correct}}(o_j, a_j)\\ &\quad = \mathbf{1}[a_j = a^*] + \delta \text{score}^S_{\text{fom}}(o_j),\end{aligned} \tag{5}$$

where $\mathbf{1}[a_j = a^*]$ is an indicator function that equals 1 when parsed answer a_j from outcome o_j matches the reference answer a^*, and 0 otherwise. This correctness score evaluates both solution accuracy and proper formatting.

2.4 Model Update

After computing novelty scores, we select B high-quality samples from valid problems for reinforcement learning, allocating half to problem generation and half to solution solving. For teacher training, we choose problem groups with highest $\text{score}_{\text{novel}}$ variance to ensure diverse training signals. For student training, we select problems with the highest novelty scores to provide maximal training value.

We optimise the policy using π_θ with an objective similar to Group Relative Policy Optimisation (GRPO) (Shao et al., 2024), adapted for on-policy training to ensure stability (Chen et al., 2025):

$$\begin{aligned}\mathcal{J}(\theta) = \mathbb{E}_{\substack{q_{1:G}\sim\pi_\theta(\cdot|p_T)\\ o_{1:G}\sim\pi_\theta(\cdot|q_i,p_S)}}&\left[\sum_{r\in\{T,S\}}\frac{1}{G}\sum_{i=1}^{G}A_i^r\right]\\ &-\beta\,\mathbb{D}_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}})\,,\end{aligned}$$

where p_T and p_S are the teacher and student prompts respectively, $r \in \{T, S\}$ refers to teacher and student, $\mathbb{D}_{KL}$ denotes the KL divergence, π_{ref} refers to the initial model before training. The advantage for each role $r \in \{T, S\}$ is computed as:

$$A_i^r = \frac{R_i^r - \text{mean}\,(R_{1:G}^r)}{\text{std}\,(R_{1:G}^r)}. \tag{6}$$

We define role-specific rewards R_i^T and R_j^S using the scoring functions from Section 2.3:

$$R_i^T = \text{score}_{\text{novel}}(q_i), \quad R_j^S = \text{score}_{\text{correct}}(o_j, a_j).$$

All valid problems are then added to the problem pool $\mathcal{P}_t$ for future iterations. The full pseudocode is provided in Algorithm 1 (§C).

3 Experiments

3.1 Experimental Setup

Training Details. We experiment with two instruction-tuned models: Llama-3.2-3B-Instruct and Llama-3.1-8B-Instruct (Grattafiori et al., 2024) with GRPO (Shao et al., 2024), and two reasoning models: DeepSeek-R1-Distill-Llama-8B (DeepSeek-AI et al., 2025) and Qwen3-8B (Qwen Team, 2025). We use a learning rate of 3×10^{-7}, 20 warm-up steps, and KL divergence coefficient 10^{-4}. All methods are trained on an equivalent number of problem-solution pairs. Each experiment is run with three random seeds. More details are available in Appendices F.1 and F.2.

Datasets. We evaluate on seven mathematical benchmarks: GSM8K (Cobbe et al., 2021), MATH-500 (Hendrycks et al., 2021), Minerva (Lewkowycz et al., 2022), OlympiadBench (He et al., 2024), College Math (Tang et al., 2024), and AIME 2024 and 2025; and three general reasoning benchmarks: BBEH (Kazemi et al., 2025), MMLU-Pro (Wang et al., 2024), and SuperGPQA (Du et al., 2025). We report Pass@1, averaged over 16 independently sampled generations per problem.

Baselines. (1) **Base**: zero-shot prompting with step-by-step reasoning. (2) **GRPO$_{\text{gsm8k}}$** and **GRPO$_{\text{math}}$**: GRPO (Shao et al., 2024) trained on GSM8K (7,473 examples) (Cobbe et al., 2021) and MATH (7,500 examples) (Hendrycks et al., 2021). (3) **Absolute Zero** (Zhao et al., 2025): self-play using Python as external verifier. (4) **R-Zero** (Huang et al., 2025): verifier-free self-play with separate challenger and solver models using repetition penalties.

Method	Mathematical Reasoning								General Reasoning				Overall Avg.
	GSM8K	MATH 500	Minerva	College Math	Olymp.	AIME 24	AIME 25	Avg.	BBEH	MMLU-Pro	Super-GPQA	Avg.	
Llama-3.2-3B-Instruct													
Base	73.94	42.86	15.21	28.78	13.09	5.00	0.21	25.58	4.49	24.91	13.54	14.31	19.95
GRPO$_{\text{gsm8k}}$	**79.72**	45.30	16.27	33.33	14.56	6.82	1.88	28.27	6.12	25.38	15.63	15.71	21.99
GRPO$_{\text{math}}$	76.48	45.26	16.09	32.95	14.13	6.47	0.92	27.47	5.98	25.59	14.87	15.48	21.48
Absolute Zero	74.37	44.71	14.78	31.93	14.42	5.53	1.05	26.68	4.54	25.16	12.64	14.11	20.40
R-Zero	76.34	44.27	15.84	32.72	14.19	5.71	1.13	27.17	4.73	25.21	13.48	14.47	20.82
OpenSIR	78.28	**46.22**	**18.46**	**35.42**	**16.72**	**7.94**	**3.97**	**29.57**	**7.13**	**27.32**	**16.60**	**17.02**	**23.30**
Llama-3.1-8B-Instruct													
Base	84.50	47.89	22.75	34.10	16.26	3.12	0.42	29.86	6.48	32.85	16.57	18.63	24.25
GRPO$_{\text{gsm8k}}$	**88.70**	50.37	24.83	35.03	16.43	5.24	1.78	31.77	7.93	34.18	17.77	19.96	25.87
GRPO$_{\text{math}}$	86.23	50.82	23.98	34.93	16.54	4.39	1.92	31.26	7.79	33.27	16.65	19.24	25.25
Absolute Zero	86.89	51.38	23.21	34.39	15.96	2.78	0.65	30.75	6.76	33.08	16.66	18.83	24.79
R-Zero	86.19	50.93	24.11	32.93	15.66	2.63	1.08	30.50	6.63	32.92	16.63	18.73	24.62
OpenSIR	87.30	**52.38**	**27.29**	**36.29**	**17.81**	**7.93**	**2.94**	**33.13**	**9.94**	**36.48**	**20.16**	**22.19**	**27.66**

Table 1: Results on instruction-tuned models. OpenSIR achieves the strongest overall performance without using training data, outperforming GRPO baselines trained on large human-annotated datasets. These gains carry over to general reasoning, suggesting that OpenSIR develops transferable skills. Prior self-play methods yield limited gains on instruction-tuned models, highlighting the need for diversity incentives.

3.2 Main Results

Instruction models. Table 1 shows that Absolute Zero and R-Zero yield at most +1.59 on Llama-3.2-3B-Instruct and +0.89 on Llama-3.1-8B-Instruct in average math performance, trailing the GRPO baselines despite much larger improvements on base models in their original work (Zhao et al., 2025; Huang et al., 2025), suggesting that self-play without explicit novelty incentives provides limited additional benefit. OpenSIR achieves the largest gains on the 3B and 8B models (+3.99, +3.27), surpassing both GRPO$_{\text{gsm8k}}$ (+2.69, +1.91) and GRPO$_{\text{math}}$ (+1.89, +1.40). It does so from a single seed problem, whereas both GRPO baselines require more than 7,000 human-annotated examples. This indicates that problem diversity is critical for effective self-play on instruction-tuned models; we examine this further in our diversity ablation (§3.3).

Reasoning models. We next examine models with extensive reasoning-focused post-training, where less headroom remains for further gains. As Table 2 shows, OpenSIR improves average math performance by +3.71 on DeepSeek-R1-Distill-Llama-8B and +2.41 on Qwen3-8B. In contrast, the data-based GRPO baselines provide only small gains, with a best result of +0.88, whereas the self-play baselines deteriorate in three of four cases, demonstrating that OpenSIR can expand capabilities even after heavy reasoning post-training and further supporting the value of novel problem generation. We include training curves across all four models in Appendix B.1.

General reasoning transfer. We now ask whether these gains extend beyond mathematics. Across Tables 1 and 2, OpenSIR improves general reasoning performance by +2.71 and +3.56 on Llama-3.2-3B-Instruct and Llama-3.1-8B-Instruct respectively; gains are larger on reasoning models, reaching +4.79 and +4.41 on DeepSeek-R1-Distill-Llama-8B and Qwen3-8B. The best baseline reaches only +1.40 on instruction models and is negative or negligible on reasoning models, suggesting that training on diverse self-generated novel problems develops transferable reasoning skills, not only mathematical proficiency. OpenSIR accordingly achieves the highest overall average on all four base models (+3.35 to +4.25).

Method	Mathematical Reasoning								General Reasoning				Overall Avg.
	GSM8K	MATH 500	Minerva	College Math	Olymp.	AIME 24	AIME 25	Avg.	BBEH	MMLU-Pro	Super-GPQA	Avg.	
DeepSeek-R1-Distill-Llama-8B													
Base	70.68	80.75	32.40	48.27	55.75	41.04	28.12	51.00	7.15	40.33	17.38	21.62	36.31
$\text{GRPO}_{\text{gsm8k}}$	72.49	79.94	32.42	49.38	57.29	40.38	28.83	51.53	7.17	41.35	18.72	22.41	36.97
$\text{GRPO}_{\text{math}}$	71.38	81.89	33.12	50.29	57.84	39.41	29.24	51.88	8.32	40.42	19.24	22.66	37.27
Absolute Zero	71.93	80.83	29.49	48.71	56.42	39.72	27.94	50.72	6.04	39.37	16.40	20.60	35.66
R-Zero	73.78	80.44	31.29	49.83	55.39	40.87	28.42	51.43	7.29	40.45	15.45	21.06	36.25
OpenSIR	**78.39**	**84.68**	**35.88**	**51.42**	**57.92**	**43.38**	**31.29**	**54.71**	**11.45**	**44.18**	**23.59**	**26.41**	**40.56**
Qwen3-8B													
Base	95.93	97.02	48.44	58.45	75.18	75.21	66.25	73.78	15.35	64.57	34.06	37.99	55.89
$\text{GRPO}_{\text{gsm8k}}$	**96.84**	96.94	49.12	57.94	73.38	74.78	64.29	73.33	14.31	62.53	34.03	36.96	55.15
$\text{GRPO}_{\text{math}}$	96.25	**98.32**	48.43	59.29	74.57	77.84	65.13	74.26	13.49	65.34	35.43	38.09	56.18
Absolute Zero	93.73	94.14	48.71	57.32	72.29	72.42	63.82	71.78	15.41	62.51	33.96	37.29	54.53
R-Zero	94.83	93.83	49.89	52.38	71.48	73.15	62.23	71.11	12.53	64.68	33.19	36.80	53.96
OpenSIR	96.34	97.13	**52.28**	**61.37**	**76.77**	**79.52**	**69.91**	**76.19**	**18.30**	**69.05**	**39.84**	**42.40**	**59.29**

Table 2: Results on reasoning models. GRPO baselines largely stagnate and prior self-play methods often degrade, while OpenSIR consistently improves both mathematical and general reasoning. Its general-reasoning transfer is stronger than on instruction-tuned models, showing that diversity-driven self-play remains effective even after heavy post-training.

3.3 Analysis

We analyse each key component of OpenSIR on Llama-3.2-3B-Instruct.

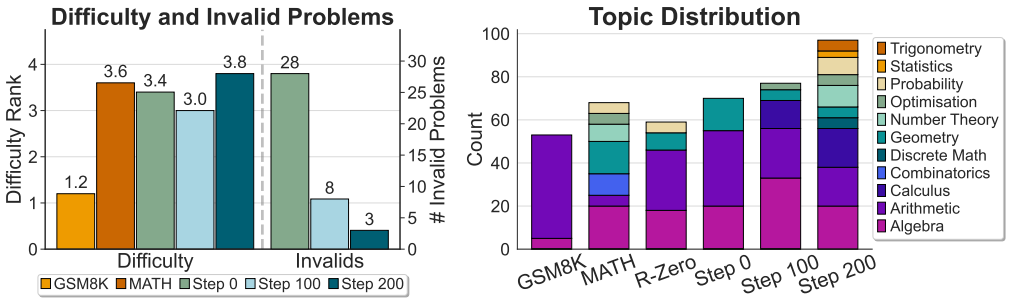

Figure 2: Evolution of problem difficulty, validity, and topic diversity during OpenSIR training (50 problems per source). **(Left)** Human-rated difficulty (1=easiest, 5=hardest) and invalid-problem counts for GSM8K, MATH, and generated problems at training steps 0, 100, 200. Difficulty follows a V-shaped trajectory, dipping then rising as solving ability improves, while validity improves sharply over training. **(Right)** Mathematical topic distribution across training stages. OpenSIR progressively broadens topic coverage, whereas R-Zero concentrates on few categories without an explicit diversity mechanism.

Evolution of Difficulty and Diversity. We track how difficulty and diversity evolve during training through human evaluation. We sample 50 problems from three OpenSIR training checkpoints (steps 0, 100, 200) and 50 each from GSM8K and MATH. Annotators evaluate mixed sets of five problems (one per source), identifying topics, assessing validity, and ranking difficulty. We additionally sample 50 problems from R-Zero for topic annotation. Figure 2 shows average difficulty rankings (1=easiest, 5=hardest); see Appendix D for full annotation instructions.

Figure 2 (left) suggests a V-shaped difficulty trend across training stages. Problems start at 3.4 average difficulty ranking, drop to 3.0 at midpoint, then rise to 3.8. This pattern reflects OpenSIR's self-calibration: the model first generates overly difficult problems, then learns appropriate difficulty, and finally increases challenge as its solving capabilities improve. The model also generates increasingly valid problems during training — validity improves from below 50% initially to 94% (47 of 50 problems) by the end.

Model	Math Avg.	Validity	Solve Rate
$\text{OpenSIR}_{0.5}$	**29.57**	70.82	89.82
$\text{OpenSIR}_{0.3}$	27.81	52.32	81.38
$\text{OpenSIR}_{0.1}$	25.97	42.31	78.31

Table 3: Lower solve-rate thresholds produce harder but less valid problems, reducing overall performance.

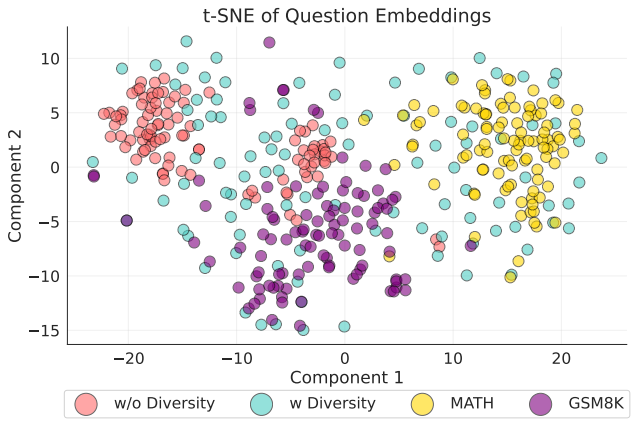

Figure 3: Diversity reward spreads generated problems across a broader embedding space, while removing it yields a tighter cluster.

Figure 2 (right) shows topic diversity expansion across training. OpenSIR progresses from basic topics (algebra, arithmetic, geometry) to advanced domains including calculus and optimisation, eventually incorporating trigonometry, statistics, and other mathematical areas. In contrast, R-Zero's problems remain concentrated in a small number of topic categories, suggesting that self-play without an explicit diversity mechanism tends to produce repetitive problem distributions. §B.2 provides detailed case studies that illustrate this evolution.

Difficulty-Validity Trade-off. We investigate the difficulty-validity trade-off by training OpenSIR variants with lower solve-rate thresholds of 0.1, 0.3, and 0.5, keeping the upper threshold at 0.9. From each variant, we sample 300 problems and assess quality with GPT-5 (OpenAI, 2025a) using 8 responses per problem. We measure validity as the agreement between GPT-5's majority answer and our reference answer, and difficulty by GPT-5's solve rate. Table 3 reveals a steep trade-off between difficulty and validity. Lowering the threshold from 0.5 to 0.1 produces moderately harder problems (GPT-5 solve rate: 89.82% $\rightarrow$ 78.31%), but validity collapses from 70.82% to 42.31%, suggesting that low-solve-rate problems are more often invalid than genuinely challenging. Downstream math accuracy confirms that this trade-off is decisively unfavourable: performance drops monotonically from 29.57 to 27.81 to 25.97 as the threshold is relaxed from 0.5 to 0.3 to 0.1, as the increased difficulty cannot compensate for training on a growing share of invalid problems. This supports our selection of 0.5 as the lower threshold for the solvability reward. While the residual $\sim$30% reference-answer disagreement at $s_{\min} = 0.5$ is non-trivial, OpenSIR still outperforms GRPO baselines trained on human-annotated examples (Table 1), indicating that this level of pseudo-label noise does not impair downstream performance.

In addition to solve-rate thresholds, we find that rewarding longer solutions provides another mechanism to promote problem complexity, encouraging complex multi-step problems (§E.1).

Impact of Diversity Rewards. We analyse the impact of diversity rewards on problem diversity through problem embeddings, n-gram similarity, and concept overlap. Figure 3 visualises problem embeddings with t-SNE, where red points represent problems without diversity reward, cyan with diversity reward, gold indicates MATH problems, and purple marks GSM8K problems. Without diversity rewards, problems cluster in narrow regions; with them, problems spread across the embedding

Model	Math Avg.	General Avg.	# Concepts
w diversity	**29.57**	17.02	5914
w/o diversity	26.84	14.52	3328

Table 4: OpenSIR gains accuracy on math and general benchmarks with diversity reward, while nearly doubling concept coverage.

space, reaching areas beyond MATH and GSM8K. N-gram similarity and concept overlap analyses confirm these patterns (§B.3). Diversity rewards yield substantial gains in both mathematical and general reasoning. Removing them reduces mathematical reasoning by 2.73 (Table 4; 29.57→26.84) and general reasoning by 2.50 (17.02→14.52), alongside a sharp drop in unique concepts explored (5914→3328). The decline in general reasoning shows that training with diversity rewards develops transferable capabilities rather than narrow mathematical proficiency. Without them, the model generates repetitive problems that limit the variety of concepts the teacher can present. We also find that joint teacher-student training is essential for difficulty calibration; fixing the teacher degrades accuracy by 3.68 points (§E.2).

4 Related Work

Self-play. Self-play achieved superhuman performance in games without human data, from AlphaGo (Silver et al., 2016; 2017), StarCraft II (Vinyals et al., 2019), Poker (Brown & Sandholm, 2019), DotA (OpenAI et al., 2019), and Diplomacy (FAIR et al., 2022). Baker et al. (2019) show that agents can discover complex strategies through self-play, suggesting that it is a promising avenue for continuous, open-ended learning. Recent works apply self-play to LLM reasoning: Absolute Zero (Zhao et al., 2025) and Spiral (Liu et al., 2025) rely on external verifiers or game rules that limit their use beyond specific domains. R-Zero (Huang et al., 2025) attempts verifier-free self-play using repetition penalties rather than explicit exploration incentives; our topic analysis suggests this leads to narrower coverage than OpenSIR, constraining open-ended learning. In contrast, OpenSIR generates and solves problems without external supervision while actively promoting diversity to enable continuous discovery of novel mathematical concepts.

Reinforcement Learning with Verifiable Feedback (RLVF). RLVF drives recent advances in LLM reasoning (OpenAI, 2024; 2025b; DeepSeek-AI et al., 2025) but requires extensive human-annotated data for verifiable reward signals (Zeng et al., 2025), creating a scalability bottleneck and potentially limiting performance to human-level. Recent works show that moderate-difficulty training samples provide optimal learning signals (Zheng et al., 2025; Sun et al., 2025), while diverse problem types enhance mathematical reasoning (Akter et al., 2025; Chen et al., 2025). These insights directly motivate OpenSIR to optimise for appropriate difficulty calibration and diversity-driven exploration, enabling models to learn math reasoning open-endedly without human supervision.

5 Conclusions

We present OpenSIR, a self-play framework that enables LLMs to autonomously learn to generate and solve novel problems without external supervision. Starting from only a single trivial math problem, our framework outperforms GRPO-trained models that utilise thousands of human annotations across diverse model families. This approach demonstrates that models can effectively bootstrap mathematical reasoning through recursive self-improvement, eliminating dependence on extensive curated datasets. We show that OpenSIR succeeds by combining difficulty calibration with diversity rewards to create an adaptive curriculum in which models continuously discover and master increasingly challenging mathematical concepts. Overall, OpenSIR represents a compelling paradigm for open-ended autonomous mathematical reasoning development, enabling models to recursively expand their capabilities beyond the boundaries of human-annotated data.

Ethics Statement

OpenSIR trains language models on self-generated mathematical problems without human supervision, which raises several considerations. First, removing the human-annotated curriculum eliminates a layer of editorial oversight—any biases, gaps, or systematic errors present in the base model can be amplified through self-play, since the same model both proposes and validates the training signal. We partially mitigate this through the solvability and diversity rewards, which filter malformed problems and discourage repetitive distributions, but residual pseudo-label noise remains (§3.3) and could compound over longer training runs. Second, the framework is domain-agnostic in principle, and the same recipe applied to non-mathematical domains lacking objective verifiability could amplify hallucinated or harmful content rather than reasoning skill; we deliberately restrict our study to mathematics, where majority-voted answers provide a meaningful proxy signal. Third, self-play increases compute relative to standard GRPO ($2\times$ per gradient update; Appendix F.1), and scaling such methods further has environmental and accessibility implications that warrant weighing against the gains in data efficiency. Finally, OpenSIR relies entirely on publicly available pre-trained models and benchmarks used under their intended research licenses; no human-subjects data is collected, and the only human annotation (§D) was performed by the authors on model-generated math problems.

AI Use Statement

AI as part of the method. OpenSIR is a self-play framework, so LLMs generate the synthetic training data by design: the policy model produces the mathematical problems and their solutions used for training, as described in Sections 2.1 and 2.2. We additionally use GPT-5 to assess the validity and difficulty of sampled generated problems in our analysis (Section 3.3). All such uses are documented in the paper, and the quality of generated problems is further examined through human annotation (Appendix D).

AI as a research aid. We used AI assistants for two purposes. (i) *Writing*: to check grammar and improve the clarity of text drafted by the authors. Every suggestion was reviewed by an author before being incorporated. (ii) *Code*: to help write code and scripts for our experiments. All AI-assisted code was reviewed and verified by at least two authors before use.

What AI was not used for. AI tools were not used to formulate the research idea, design the methodology or experiments, derive mathematical claims, interpret results, or survey the literature. The authors take full responsibility for all content of this paper.

References

Syeda Nahida Akter, Shrimai Prabhumoye, Matvei Novikov, Seungju Han, Ying Lin, Evelina Bakhturina, Eric Nyberg, Yejin Choi, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Nemotron-CrossThink: Scaling Self-Learning beyond Math Reasoning, April 2025.

Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch. Emergent Tool Use From Multi-Agent Autocurricula. In *International Conference on Learning Representations*, September 2019. URL https://openreview.net/forum?id=SkxpxJBKwS.

Jakob Bauer, Kate Baumli, Feryal Behbahani, Avishkar Bhoopchand, Nathalie Bradley-Schmieg, Michael Chang, Natalie Clay, Adrian Collister, Vibhavari Dasagi, Lucy Gonzalez, Karol Gregor, Edward Hughes, Sheleem Kashem, Maria Loks-Thompson, Hannah Openshaw, Jack Parker-Holder, Shreya Pathak, Nicolas Perez-Nieves, Nemanja Rakicevic, Tim Rocktäschel, Yannick Schroecker, Satinder Singh, Jakub Sygnowski, Karl Tuyls, Sarah York, Alexander Zacherl, and Lei M. Zhang. Human-Timescale Adaptation in an Open-Ended Task Space. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 1887–1935. PMLR, July 2023. URL https://proceedings.mlr.press/v202/bauer23a.html.

Noam Brown and Tuomas Sandholm. Superhuman AI for multiplayer poker. *Science*, 365(6456): 885–890, August 2019. doi: 10.1126/science.aay2400.

Yang Chen, Zhuolin Yang, Zihan Liu, Chankyu Lee, Peng Xu, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. AceReason-Nemotron: Advancing Math and Code Reasoning through Reinforcement Learning, May 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training Verifiers to Solve Math Word Problems, November 2021.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning, January 2025.

Xinrun Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, et al. SuperGPQA: Scaling LLM evaluation across 285 graduate disciplines. *arXiv preprint arXiv:2502.14739*, 2025.

FAIR, Anton Bakhtin, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, Jonathan Gray, Hengyuan Hu, Athul Paul Jacob, Mojtaba Komeili, Karthik Konath, Minae Kwon, Adam Lerer, Mike Lewis, Alexander H. Miller, Sasha Mitts, Adithya Renduchintala, Stephen Roller, Dirk Rowe, Weiyan Shi, Joe Spisak, Alexander Wei, David Wu, Hugh Zhang, and Markus Zijlstra. Human-level play in the game of Diplomacy by combining language models with strategic reasoning. *Science*, 0(0):eade9097, November 2022. doi: 10.1126/science.ade9097.

Joseph L Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76 (5):378, 1971.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv e-prints*, pp. arXiv–2407, 2024.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A Challenging Benchmark for Promoting AGI with Olympiad-Level Bilingual Multimodal Scientific Problems. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.211.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring Mathematical Problem Solving With the MATH Dataset. *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 1, December 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. R-Zero: Self-Evolving Reasoning LLM from Zero Data, August 2025.

Edward Hughes, Michael Dennis, Jack Parker-Holder, Feryal Behbahani, Aditi Mavalankar, Yuge Shi, Tom Schaul, and Tim Rocktaschel. Open-Endedness is Essential for Artificial Superhuman Intelligence, June 2024a.

Edward Hughes, Michael D. Dennis, Jack Parker-Holder, Feryal M. P. Behbahani, Aditi Mavalankar, Yuge Shi, Tom Schaul, and Tim Rocktäschel. Position: Open-endedness is essential for artificial superhuman intelligence. In *ICML*. OpenReview.net, 2024b.

Mehran Kazemi, Bahare Fatemi, Hritik Bansal, John Palowitch, Chrysovalantis Anastasiou, Sanket Vaibhav Mehta, Lalit K Jain, Virginia Aglietti, Disha Jindal, Peter Chen, et al. Big-bench extra hard. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 26473–26501, 2025.

Maurice G Kendall and B Babington Smith. The problem of m rankings. *The annals of mathematical statistics*, 10(3):275–287, 1939.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving Quantitative Reasoning Problems with Language Models. *Advances in Neural Information Processing Systems*, 35:3843–3857, December 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/18abbeef8cfe9203fdf9053c9c4fe191-Abstract-Conference.html.

Bo Liu, Leon Guertler, Simon Yu, Zichen Liu, Penghui Qi, Daniel Balcells, Mickel Liu, Cheston Tan, Weiyan Shi, Min Lin, Wee Sun Lee, and Natasha Jaques. SPIRAL: Self-Play on Zero-Sum Games Incentivizes Reasoning via Multi-Agent Multi-Turn Reinforcement Learning, June 2025.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*, September 2018. URL https://openreview.net/forum?id=Bkg6RiCqY7.

OpenAI. Learning to reason with LLMs, 2024. URL https://openai.com/index/learning-to-reason-with-llms/.

OpenAI. GPT-5 System Card, August 2025a. URL https://openai.com/index/gpt-5-system-card/.

OpenAI. Introducing OpenAI o3 and o4-mini, 2025b. URL https://openai.com/index/introducing-o3-and-o4-mini/.

OpenAI, Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique P. d O. Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with Large Scale Deep Reinforcement Learning, December 2019.

Julien Pourcel, Cédric Colas, Gaia Molinaro, Pierre-Yves Oudeyer, and Laetitia Teodorescu. ACES: Generating a Diversity of Challenging Programming Puzzles with Autotelic Generative Models. *Advances in Neural Information Processing Systems*, 37:67627–67662, December 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/7d0c6ff18f16797b92e77d7cc95b3c53-Abstract-Conference.html.

Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models, April 2024.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, January 2016. ISSN 0028-0836, 1476-4687. doi: 10.1038/nature16961.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, October 2017. ISSN 1476-4687. doi: 10.1038/nature24270.

Yifan Sun, Jingyan Shen, Yibin Wang, Tianyu Chen, Zhendong Wang, Mingyuan Zhou, and Huan Zhang. Improving Data Efficiency for LLM Reinforcement Fine-tuning Through Difficulty-targeted Online Data Selection and Rollout Replay, June 2025.

Zhengyang Tang, Xingxing Zhang, Benyou Wang, and Furu Wei. MathScale: Scaling Instruction Tuning for Mathematical Reasoning. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 47885–47900. PMLR, July 2024. URL https://proceedings.mlr.press/v235/tang24k.html.

Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782): 350–354, November 2019. ISSN 1476-4687. doi: 10.1038/s41586-019-1724-z.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. TRL: Transformers reinforcement learning. https://github.com/huggingface/trl, 2020.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. MMLU-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37, 2024.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does Reinforcement Learning Really Incentivize Reasoning Capacity in LLMs Beyond the Base Model?, April 2025.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. SimpleRL-Zoo: Investigating and Taming Zero Reinforcement Learning for Open Base Models in the Wild, March 2025.

Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Yang Yue, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute Zero: Reinforced Self-play Reasoning with Zero Data, May 2025.

Haizhong Zheng, Yang Zhou, Brian R. Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. Act Only When It Pays: Efficient Reinforcement Learning for LLM Reasoning via Selective Rollouts, June 2025.

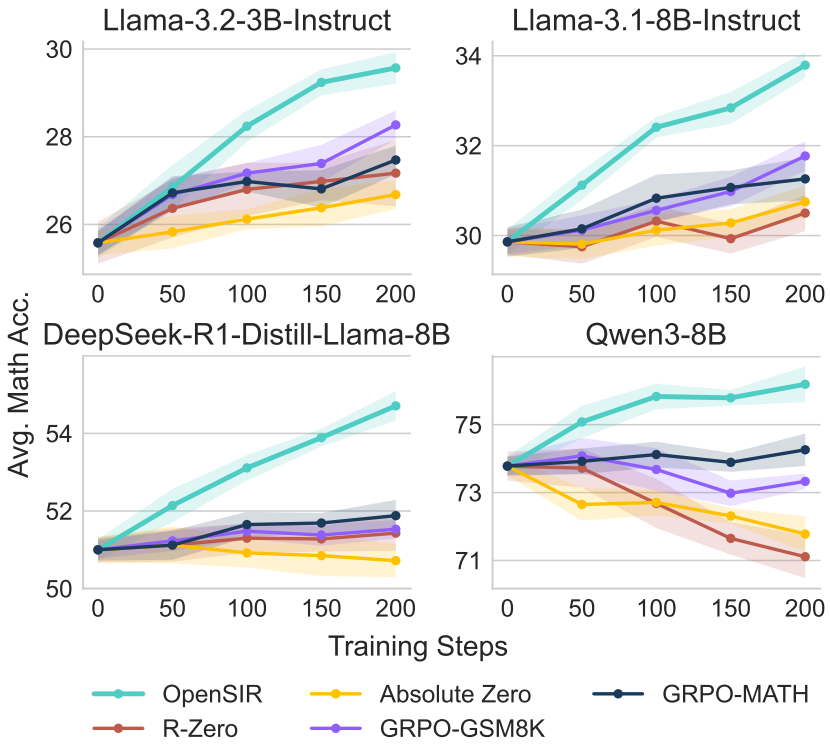

Figure 4: Average math accuracy over training steps for all four models.

A Limitations

Performance eventually plateaus after extended training, suggesting that sustaining novelty within one domain may require stronger or complementary exploration mechanisms. Like prior self-play work, OpenSIR explores within a single domain (mathematics with verifiable answers); extending novelty-driven self-improvement across domains is a natural direction toward more open-ended learning. The diversity reward might rely on the quality of the embedding model used for scoring; we do not ablate alternative embedding models, though the consistent improvements across all four base models suggest robustness of the overall approach.

B Extended Results and Analysis

B.1 Training Curves

Figure 4 shows the average math accuracy over training steps for all four models. OpenSIR consistently improves throughout training, whereas baselines plateau or degrade.

B.2 Case Study

This section provides further analysis of question-solution pairs during training.

As discussed in Section 3.3, the model generates predominantly invalid problems early in training. The majority of these problems, primarily involving simple mathematical concepts like arithmetic, fail due to missing information (Figures 5 and 6). When attempting complex topics like optimisation, which are rare in the beginning, the model produces problems with missing information and fundamental formulation errors (Figure 7). This reveals the model has a limited understanding of underlying mathematical concepts. Invalid problems tend to exhibit low solve rates (≤ 0.25) and correspondingly

> A concert venue charges an admission price per seat and also offers a VIP ticket package that includes food, drinks, and other perks. If the food and other perks are included in the VIP ticket price and are worth $10 per person, and a group of friends want to buy the same number of VIP tickets as they would with regular tickets, what is the minimum admission price per regular ticket to make the total cost of the VIP tickets equal to or less than the total cost of the regular tickets?

Figure 5: An invalid arithmetic question generated in step 0 with solve rate of 0.25. This question is invalid since the VIP tick price is not provided, and therefore, it's impossible to calculate the minimum regular ticket price.

> Find the percentage difference in the cumulative growth of two continuously compounded interest functions after 5 years: $A = P * e^{(rt)}$, where A is the amount of money accumulated after n years, including interest, P is the principal amount, r is the annual interest rate, and t is the time the money is invested for.

Figure 6: An invalid arithmetic question generated in step 0 with solve rate of 0.125. This question is invalid since the two interest rates and principal amounts are not provided. Hence, it's impossible to calculate the percentage difference with just the general formula provided.

receive lower rewards, helping the model learn to generate valid problems. Consequently, invalid problems decrease rapidly across training (§3.3).

However, not all problems with low solve rates are invalid (§3.3). We find that some problems involving certain topics that are challenging for the model, such as geometric series, persistently exhibit low solve rates (Figure 8). The model struggles with exponentiation calculations, resulting in poor performance on geometric series problems. This reveals a fundamental trade-off in OpenSIR: while higher solve rate thresholds effectively filter out invalid problems, they inevitably discourage exploration of genuinely difficult topics. Since these problems have low solvability scores, they are likely to not receive sufficient encouragement to further explore these topics.

In later training stages, we observe OpenSIR gradually expanding into advanced mathematical domains. After 100 training steps, the model starts to generate problems involving concepts like optimisation (Figure 9), calculus (Figure 10), trigonometry-based physics (Figure 11), probability (Figure 12), among others. While these advanced problems yield lower solve rates, which indicate the model has a limited understanding of these domains, they achieve high novelty scores with large semantic distances and longer solutions. This progression validates how novelty rewards in OpenSIR drive exploration of diverse mathematical concepts, enabling open-ended learning.

B.3 Further Analysis on Questions Diversity

Figure 13 presents n-gram similarity and concept analysis. We compute ROUGE-L scores between problem texts and extract mathematical concepts using GPT-5 from problems at steps 0, 100, and 200, as well as from the MATH and GSM8K training sets. With diversity rewards (top row), problems maintain low ROUGE-L scores and minimal concept overlap both across training stages and with MATH/GSM8K. Without diversity rewards (bottom row), both textual similarity and concept overlap increase, confirming limited exploration of new problem types.

B.4 Further Analysis on Question Difficulty Progression

Figure 14 compares solve rates between the evolving OpenSIR policy and the fixed instruction model (Base) on problems generated during training. While OpenSIR's solve rate remains stable around 0.7 due to solvability-based problem selection (§2.3.1), this approximately stable rate does not imply constant problem difficulty. As OpenSIR improves during training, maintaining the same solve rate requires generating progressively harder problems. To verify this difficulty progression objectively, we measure how the initial instruction model (Base) performs on the same problems. The base model's solve rate first rises (0.48→0.58 at step 80) then declines (0.58→0.32 at step 200),

Consider two positive integers m and n (m $\geq$ n). Given a 2x2 matrix of numbers where each element is a non-negative integer, find the maximum value of the following expression: $ax^2 + by^2 + cxy$, where a, b, and c are constants, subject to the constraint that the sum of any two elements in each row and column of the matrix are equal. What is the maximum possible value of $ax^2 + by^2 + cxy$?

Figure 7: An invalid optimisation question generated in step 0 with solve rate of 0.125. This question is invalid because there are missing information about the constants a, b, and c. There are also ambiguities in the question, such as the role of m and n in the problem. It also did not explain what the elements of the matrix represent. Lastly, it contains problem formulation errors, specifically failing to specify constraints that ensure bounded solutions, demonstrating insufficient understanding of optimization problem structure.

Model	GSM8K	MATH 500	Minerva	College Math	Olymp.	AIME 24	AIME 25	Math Avg.
OpenSIR	78.28	46.22	18.46	35.42	16.72	7.94	3.97	29.57
$\text{OpenSIR}_{\text{MATH}}$	77.96	46.48	18.72	35.68	16.92	7.82	4.22	29.72
$\text{OpenSIR}_{\text{AIME}}$	78.00	45.90	18.20	35.10	16.40	8.20	3.86	29.38

Table 5: Performance of OpenSIR with different initial seed problem.

confirming the V-shaped difficulty pattern trend from Section 3.3: problems initially become easier as OpenSIR learns appropriate calibration, then progressively harder as it increases challenge. Crucially, this pattern shows that OpenSIR's reasoning ability improved over training.

B.5 Sensitivity to the initial seed problem

To address whether OpenSIR can robustly escape the limited starting point of a trivial arithmetic seed ("What is 1+1?"), we experiment with two substantially different initial seeds using Llama-3.2-3B-Instruct: a geometry problem from the MATH dataset, representing a different mathematical domain, and a competition-level problem from AIME 2024, which is significantly more challenging than the trivial seed.

Table 5 shows that all three variants achieve nearly identical performance (29.57, 29.72, and 29.38), with differences of less than 0.4 percentage points. These results indicate that OpenSIR is robust to the initial seed problem, successfully escaping the limited starting point regardless of whether it begins with trivial arithmetic, a different mathematical domain (geometry), or a significantly more challenging competition-level problem.

Figure 17 visualises the diversity of problems generated at the final training step across the three different initial seeds. The t-SNE embeddings reveal that all three variants produce diverse problems spanning similar regions of the semantic space, with substantial overlap in their distributions regardless of the initial seed. This confirms that OpenSIR successfully escapes its starting point by exploring a wide range of mathematical concepts, driven by the diversity and solvability rewards that encourage continuous exploration beyond the initial problem domain and difficulty level.

B.6 OpenSIR Incentivises Reasoning Capacity

To investigate whether OpenSIR elicits reasoning improvements rather than memorisation, we evaluate pass@k performance on five challenging mathematical benchmarks following (Yue et al., 2025).

Figure 18 shows that OpenSIR consistently outperforms base instruction models across all k values (8–256) on all benchmarks, with stable or increasing gaps at higher k. These results suggest that OpenSIR improves the model's mathematical reasoning capacity.

Question

A company is adding new employees to its workforce. Initially, the company has 500 employees and adds 10 new employees every quarter. If this trend continues for the next 5 years, and each quarter the company adds a fixed percentage increase in the number of new employees (starting with 10%), how many new employees can the company be expected to add during the first 2 years?

Solution (Incorrect)

To find the total number of new employees added in the first 2 years, we need to calculate the number of new employees added each quarter for the first 2 years and then sum them up.

First, let's calculate the number of quarters in 2 years: 2 years * 4 quarters/year = 8 quarters.

The company starts with 10 new employees and adds 10% more each quarter. To find the total number of new employees added each quarter, we can use the formula for compound interest:

$A = P * (1 + r)^n$

where:
A = total number of new employees after n quarters
P = initial number of new employees (10)
r = percentage increase per quarter (10% or 0.1)
n = number of quarters

However, since we want to find the total number of new employees added each quarter, we need to find the difference between A and the initial number of new employees (10). We can rewrite the formula as:

$A - P = P * (1 + r)^n$
$A - 10 = 10 * (1 + 0.1)^n$
$A = 10 * (1.1)^n$

Now, we can calculate A for each quarter:

Quarter 1 (n = 1): A = $10 * (1.1)^1 = 11$
Quarter 2 (n = 2): A = $10 * (1.1)^2 = 12.1$
Quarter 3 (n = 3): A = $10 * (1.1)^3 = 13.21$
Quarter 4 (n = 4): A = $10 * (1.1)^4 = 14.41$
Quarter 5 (n = 5): A = $10 * (1.1)^5 = 15.71$
Quarter 6 (n = 6): A = $10 * (1.1)^6 = 17.31$
Quarter 7 (n = 7): A = $10 * (1.1)^7 = 19.33$
Quarter 8 (n = 8): A = $10 * (1.1)^8 = 21.59$

Now, we can calculate the total number of new employees added in the first 2 years by summing up the number of new employees added each quarter:

Total = 11 + 12.1 + 13.21 + 14.41 + 15.71 + 17.31 + 19.33 + 21.59
Total = 119.66

Since we can't have a fraction of an employee, we can round down to the nearest whole number.

Therefore, the final answer is: $\boxed{119}$

Figure 8: A valid arithmetic problem involving geometric series with a solve rate of 0.125. The solution requires either summing quarterly employee additions or applying the geometric series formula: $a \cdot \frac{r^n-1}{r-1}$. Llama-3.2-3B-Instruct struggles with exponentiation calculations, resulting in poor performance on geometric series problems.

A farmer has 10 different types of vegetables to store in a warehouse with 1,000 square meters of floor space. There are four container sizes available:
- Small (5 sq m): Maximum 50 available
- Medium (10 sq m): Maximum 40 available
- Large (15 sq m): Maximum 30 available
- Extra-large (20 sq m): Maximum 25 available

The vegetables have different storage requirements:
- 3 bulky vegetables (pumpkins, watermelons, cabbages) require containers of at least 15 sq m
- 4 medium vegetables (tomatoes, peppers, eggplants, zucchini) require containers of at least 10 sq m
- 3 small vegetables (carrots, onions, potatoes) can fit in any container size

Each vegetable type must be stored in at least one container. What is the maximum number of containers that can be used while satisfying all constraints and not exceeding 1,000 sq m total space?

Figure 9: A valid optimisation problem with a solve rate of 0.375 generated at step 124.

Find the equation of the curve y = f(x) where the derivative is given by f′(x) = $(3x^2 - x - 2)$/2x and the curve passes through the point (2, 3).

Figure 10: A valid calculus problem with a solve rate of 0.375 generated at step 156.

A golfer hits a ball from the top of a 50-meter high cliff with an initial velocity of 30 m/s at an angle of 45 degrees above the horizontal. What is the horizontal distance traveled by the ball when it hits the ground?

Figure 11: A valid physics problem that involves trigonometry with a solve rate of 0.5 generated at step 172.

Consider a randomly ordered sequence of $n = 3q$ distinct integers $\{a_1, a_2, \ldots, a_{3q}\}$ where q is a positive integer. Define f as the number of adjacent pairs (a_i, a_{i+1}) in the sequence where both integers have the same remainder when divided by 3 (i.e., $a_i \bmod 3 = a_{i+1} \bmod 3$). If the integers 1 through $3q$ are randomly permuted to form this sequence, what is the expected value of f?

Figure 12: A valid probability problem with a solve rate of 0.25 generated at step 188.

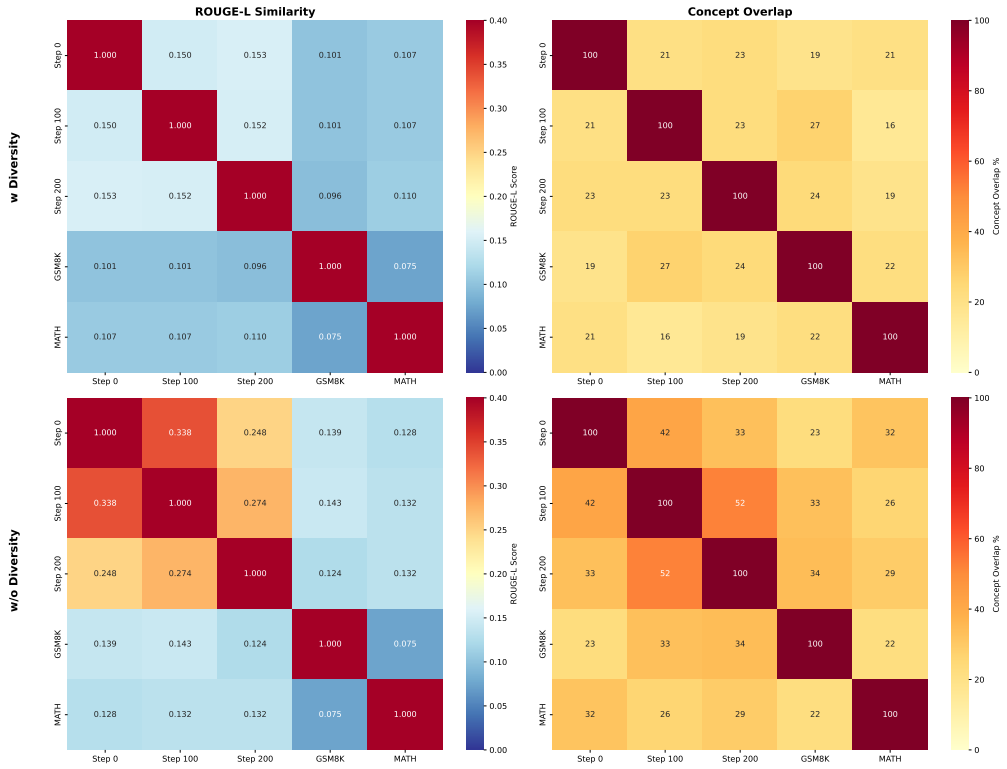

Figure 13: Heatmap visualisation of n-gram similarity (ROUGE-L scores) and concept overlap between generated problems at training steps 0, 100, 200 and reference datasets (MATH, GSM8K). Top row: with diversity reward; Bottom row: without diversity reward. With diversity reward incorporated, the generated problems exhibit low textual similarity and minimal concept overlap, demonstrating effective exploration of diverse problem types.

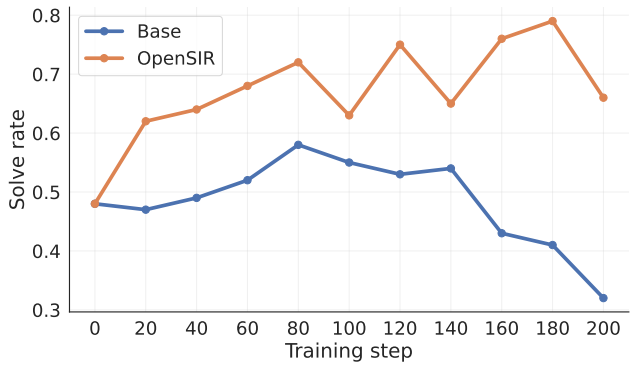

Figure 14: Progression of solve rates of OpenSIR and the initial instruction model as training goes.

B.7 Prolonged Training Analysis

To better understand the limitation of OpenSIR, we extend training of a single run of Llama-3.2-3B-Instruct to 500 steps, substantially beyond the 200 steps used in the main experiments. Figure 19 shows the evaluation performances over training. We observe consistent improvement from 25.6% to 33.5% at step 300, representing a gain of +7.9 points. Performance then plateaus after step 300 and remains around 33% through step 500, with no further noticeable gains in this single run.

B and C trisect $\overline{AD}$ and M is the midpoint of $\overline{AD}$. $MC = 8$. How many units are in the length of $\overline{AD}$?

Figure 15: A geometry problem from the MATH dataset, representing a different mathematical domain from the trivial arithmetic seed.

Every morning Aya goes for a 9-kilometer-long walk and stops at a coffee shop afterwards. When she walks at a constant speed of s kilometers per hour, the walk takes her 4 hours, including t minutes spent in the coffee shop. When she walks $s + 2$ kilometers per hour, the walk takes her 2 hours and 24 minutes, including t minutes spent in the coffee shop. Suppose Aya walks at $s + \frac{1}{2}$ kilometers per hour. Find the number of minutes the walk takes her, including the t minutes spent in the coffee shop.

Figure 16: A competition-level problem from AIME 2024, significantly more challenging than the trivial seed.

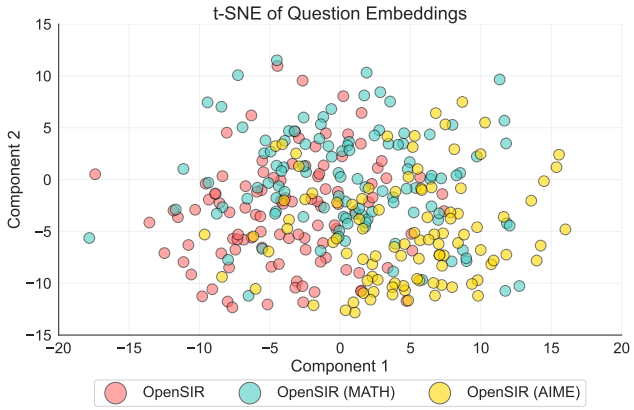

Figure 17: t-SNE visualisation of problem embeddings generated by OpenSIR from three different initial seeds. The substantial overlap demonstrates that the method converges to similar problem distributions regardless of the starting point.

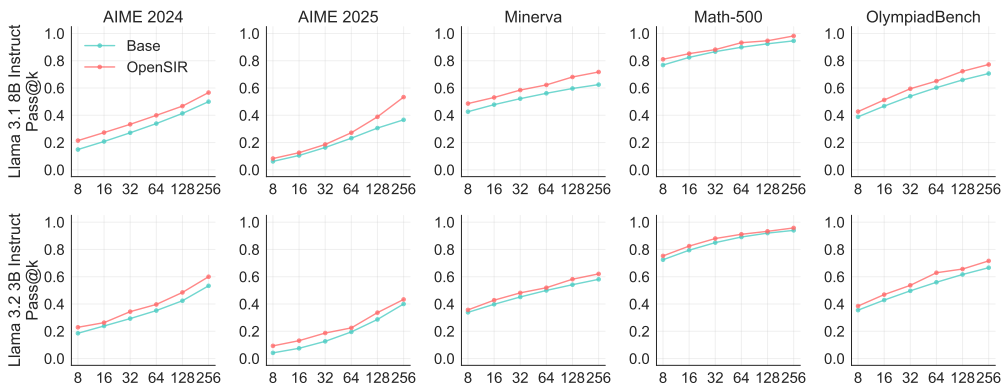

Figure 18: Pass@k curves comparing base instruction models and OpenSIR across five mathematical benchmarks. OpenSIR consistently improves performance across all k values, with stable or increasing gaps at higher k, demonstrating genuine reasoning improvements rather than memorization.

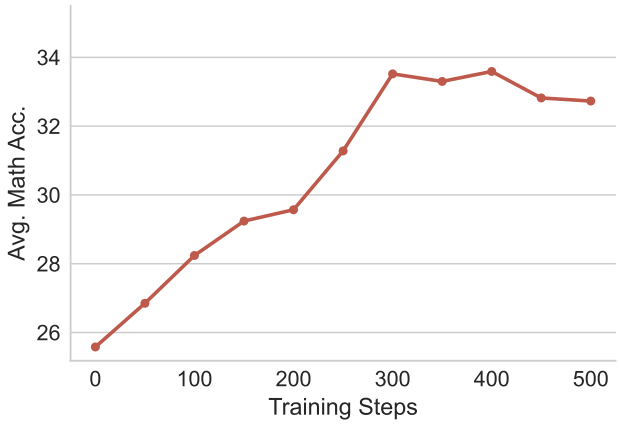

Figure 19: Performance of OpenSIR extended training using Llama-3.2-3B-Instruct.

By step 300, generated problems show narrower topical and difficulty variation than earlier checkpoints, and performance plateaus thereafter. Future work could explore richer seed curricula, larger model capacity, or complementary exploration strategies to prolong broad exploration before plateauing.

B.8 Synergy with Annotated Data

While we showed that OpenSIR achieves significant improvements in math reasoning without using annotated data, we further investigate if OpenSIR can be combined with annotated data to achieve even greater performance gains. Having demonstrated that OpenSIR achieves significant improvements without annotated data, we investigate whether combining OpenSIR with annotated data can yield further gains. We focus on Llama-3.2-3B-Instruct and use Gsm8K as the training data, as Table 1 shows that fine-tuning on GSM8K consistently outperforms using MATH.

We explore two training strategies: (1) **GSM8K → OpenSIR**: The model is first trained on GSM8K for half the training iterations, then trained with OpenSIR for the remaining half. (2) **GSM8K & OpenSIR**: Each training iteration uses half GSM8k samples and half OpenSIR samples.

Table 6 shows that both setups achieve better performance than using OpenSIR alone or only on GSM8K. One possible explanation for the sequential approach's effectiveness (GSM8K → OpenSIR) is that training on GSM8K first may improve the model's foundational reasoning abilities, which could provide a stronger starting point for OpenSIR's self-generated questions. The concurrent

Model	GSM8K	MATH 500	Minerva	College Math	Olymp.	AIME 24	AIME 25	Math Avg.
Base	73.94	42.86	15.21	28.78	13.09	5.00	0.21	25.58
$\text{GRPO}_{\text{gsm8k}}$	79.72	45.30	16.27	33.33	14.56	6.82	1.88	28.27 $^{+2.69}$
OpenSIR	78.28	46.22	18.46	35.42	16.72	7.94	3.97	29.57 $^{+3.99}$
GSM8K $\rightarrow$ OpenSIR	81.43	46.12	19.43	36.15	**18.35**	8.38	4.12	30.57 $^{+4.99}$
GSM8K & OpenSIR	**81.57**	**49.48**	**20.39**	**36.85**	18.14	**8.64**	**4.38**	**31.35** $^{+5.77}$

Table 6: The Pass@1 performance on seven mathematical benchmarks. OpenSIR obtains better results when trained together with GSM8K compared to OpenSIR or GSM8K alone.

approach (GSM8K & OpenSIR) achieves a slight additional edge, which might be attributed to the model receiving better feedback signals for question calibration from the beginning, as it can leverage both supervised and self-generated data simultaneously throughout training. The precise underlying mechanisms for these improvements require further investigation.

B.9 Computational Cost Analysis

Let B be the rollout batch size per gradient update. OpenSIR uses double the rollout batch size of standard GRPO, split evenly between teacher (problem-generation) and student (solution) rollouts after problem selection (§2.4). The per-update cost is therefore exactly $2\times$ the baseline.

Problem embeddings for diversity scoring are computed asynchronously during solution generation, adding negligible wall-clock time. Cosine distance calculations require $\mathcal{O}(B \times |\mathcal{P}_t|)$ operations, where $|\mathcal{P}_t|$ is the problem pool size, and take under 3 seconds per iteration.

C ALGORITHM

Algorithm 1 OpenSIR

Require: Problem pool $\mathcal{P}_0$, policy $\pi_\theta^{(0)}$, encoder ε, batch size B, group size G, solve rate range $[s_{\min}, s_{\max}]$, prompts p_T, p_S

for $t = 1$ **to** T **do**

// Problem Generation

Sample $k{=}B/G$ refs $\{p_1, \ldots, p_k\}$ from $\mathcal{P}_{t-1}$

for $i = 1$ **to** k **do**

Sample $q_{i,1:G} \sim \pi_\theta^{(t)}(\cdot \mid p_i, p_T)$

end for

$\mathcal{Q}_{\text{valid}} \leftarrow \{q_{i,j} \mid q_{i,j}$ has valid format$\}$

// Solution Sampling

for each $q_i \in \mathcal{Q}_{\text{valid}}$ **do**

Sample $o_{i,1:G} \sim \pi_\theta^{(t)}(\cdot \mid q_i, p_S)$

Parse answers $a_{i,1:G}$ from $o_{i,1:G}$

$a_i^* \leftarrow \arg\max_{a \in a_{i,1:G}} \text{count}(a)$

$s_{q_i} = \text{count}(a_i^*)/G;\ e_{q_i} \leftarrow \varepsilon(q_i)$

end for

// Scoring

Compute $\text{score}_{\text{novel}}(q_i)\ \forall\, q_i \in \mathcal{Q}_{\text{valid}}$

// Teacher sample selection

$\mathcal{I}_T \leftarrow \text{top}_{B/(2G)}(i : \text{Var}(\text{score}_{\text{novel}}(q_{i,1:G})))$

// Student sample selection

$\mathcal{Q}_S \leftarrow \text{top}_{B/(2G)}(q : \text{score}_{\text{novel}}(q))$

Compute $\text{score}_{\text{correct}}(o_{i,j}, a_{i,j})$ for $q_i \in \mathcal{Q}_S$

// Model Update

$\mathcal{D}_T \leftarrow \{(p_T, q_{i,j}, R_{i,j}^T) : i \in \mathcal{I}_T\}$
where $R_{i,j}^T = \text{score}_{\text{novel}}(q_{i,j})$

$\mathcal{D}_S \leftarrow \{(p_S, o_{i,j}, R_{i,j}^S) : q_i \in \mathcal{Q}_S\}$
where $R_{i,j}^S = \text{score}_{\text{correct}}(o_{i,j}, a_{i,j})$

$\pi_\theta^{(t+1)} \leftarrow \text{GRPO}(\pi_\theta^{(t)}, \mathcal{D}_T \cup \mathcal{D}_S)$

$\mathcal{P}_t \leftarrow \mathcal{P}_{t-1} \cup \mathcal{Q}_{\text{valid}}$

end for

return $\pi_\theta^{(T)}$

D ANNOTATION DETAILS

One of the authors prepared the samples for annotation, and the rest of the authors annotated the samples with the instructions provide in Figure 20.

Inter-annotator agreement. To assess reliability, annotators shared a common set of 20 overlapping samples. We report Fleiss' κ (Fleiss, 1971) for topic labels ($\kappa = 0.82$) and validity ($\kappa = 0.86$), indicating satisfactory agreement on both categorical tasks. For difficulty rankings, we compute Kendall's coefficient of concordance W (Kendall & Smith, 1939) per set and average across sets, obtaining $W = 0.67$, which reflects moderate alignment on relative problem difficulty.

E ADDITIONAL ABLATIONS

E.1 SOLUTION LENGTH REWARD INCREASES PROBLEM COMPLEXITY

We investigate the impact of the solution length reward in OpenSIR. Table 7 shows this reward improves performance from 28.09% to 29.57%. It also increases the average question length (from 150 to 207 tokens) and solution lengths (from 238 to 387 tokens). By examining the generated questions manually, we find that the policy tends to generate more sophisticated problems involving advanced concepts with this reward, such as linear programming and optimization, which naturally require longer multi-step solutions to solve. These results demonstrate that the solution length reward

You will be presented with multiple sets of 5 math problems to evaluate. For each set, please complete the following three-step annotation process.
Step 1: Identify Topics
For **each problem**, identify ALL relevant mathematical topics from the following list: Algebra, Geometry, Calculus, Probability, Statistics, Number Theory, Combinatorics, Optimization, Arithmetic, Discrete Math, Trigonometry.
Step 2: Assess Validity
For **each problem**, determine if it is **valid** or **invalid**:
- **Valid**: The problem is logically sound, clearly stated, and can be answered with the given information
- **Invalid**: The problem contains logical flaws, contradictions, insufficient information, or ambiguities that prevent a proper solution
Step 3: Rank Difficulty
Rank all 5 problems from **easiest to hardest**. Provide your ranking as a sequence of problem numbers. *Example:* [3, 1, 5, 2, 4] means problem 3 is the easiest and 4 is the hardest. *Consider these factors when assessing difficulty:* Number of steps required, Complexity of concepts involved, Level of mathematical knowledge needed, Computational complexity.
Response Format
Provide your annotations as a JSON list where each element represents one problem set. Here are some examples:

```
[{"set_id": "SET_1",
  "problems": {"1": {"topics": ["Algebra", "Calculus"], "valid": true},
               "2": {"topics": ["Geometry"], "valid": false},
               "3": {"topics": ["Probability"], "valid": true},
               "4": {"topics": ["Number Theory"], "valid": true},
               "5": {"topics": ["Arithmetic"], "valid": true}},
  "difficulty_ranking": [5, 3, 1, 2, 4]},
 {"set_id": "SET_2",
  "problems": {"1": {"topics": ["Statistics"], "valid": true},
               "2": {"topics": ["Discrete Math"], "valid": true},
               "3": {"topics": ["Optimization"], "valid": true},
               "4": {"topics": ["Algebra"], "valid": false},
               "5": {"topics": ["Geometry", "Algebra"], "valid": true}},
  "difficulty_ranking": [1, 2, 5, 3, 4]}, ...]
```

Figure 20: The instruction provided to the annotators to annotate problems.

Model	Question Length	Solution Length	Math Avg.
w/ length	207	387	**29.57**
w/o length	150	238	**28.09**

Table 7: Comparison of OpenSIR performance with and without solution length reward. Solution length reward improves OpenSIR accuracy and increases average question and solution lengths.

Trained Roles	Math Avg.	General Avg.	Avg. Solve Rate
Both	**29.57**	**17.02**	72.20 ($\pm$4.49)
Student	25.89	14.27	64.56 ($\pm$17.37)

Table 8: Effect of teacher training. Joint training achieves higher accuracy and remarkably stable problem difficulty (much lower solve rate variance), demonstrating that teacher training enables calibrated problem generation at optimal difficulty levels for effective learning.

effectively guides the policy toward generating more complex problems, which in turn leads to better performance.

E.2 Importance of Dual-Role Training

We evaluate the contribution of the joint teacher-student training by testing a variant where only the student is updated while the teacher remains fixed at its initial state. Table 8 shows that accuracy drops significantly from 29.57 to 25.89 when only the student is trained. This suggests that effective self-play requires both components to co-evolve.

Without teacher training, generated problems become harder (solve rate drops from 72.20 to 64.56) and drift from the optimal 70% target solve rate established in Section 3.3. More critically, solve rate variance increases tremendously (from $\pm$4.49 to $\pm$17.37), indicating highly inconsistent difficulty during training. This poorly calibrated curriculum explains the performance drop: the fixed teacher cannot adapt to the student's evolving capabilities, whereas joint training enables continuous difficulty calibration at the optimal challenge level.

F Implementation Details

F.1 Training Details

We implement OpenSIR based on the TRL framework (von Werra et al., 2020). Each OpenSIR batch is split between teacher (problem generator) and student (solver) rollouts, so we double OpenSIR's total rollout batch size relative to the baselines; this keeps the number of student problem-solution pairs per gradient step (and the total over training) equal across methods. Table 9 provides a summary of the training hyperparameters used in our experiments.

F.2 Evaluation Details

We use sampling temperature 0.6 and top-p 0.95 with maximum response length 4,096 tokens (38,912 for reasoning models, following Qwen Team (2025)). We report Pass@1, averaged over 16 independently sampled generations per problem. Answer extraction and comparison use the `math_verify` library.

F.3 Prompts

We detailed the prompt for generating problems in Figure 21 and solving problems in Figure 22.

Category	Hyperparameter	Value
Trainer	Learning rate	3×10^{-7}
	Optimiser	AdamW (Loshchilov & Hutter, 2018)
	Warmup steps	20
	Training steps	200
	KL loss coefficient	1×10^{-4}
	Gradient norm clipping	0.5
	Seeds	42/43/44
	GPUs	3 H100
Rollout	Batch size[†]	256 / 512[‡]
	Max prompt length	1024
	Max solution length	2048
	Number of rollouts per prompt	8
	Temperature	1.0
Teacher Rewards	Solvability weight (α)	1.0
	Solution length weight (λ)	1.0
	Diversity weight (γ)	1.0
	Format weight (δ)	0.1
	Embedding model	Linq-Embed-Mistral (7B)
Student Rewards	Accuracy weight	1.0
	Format weight (δ)	0.1

[†] The number of rollouts seen for one gradient update.
[‡] Baselines use batch size 256; OpenSIR uses 512 because each batch is split between teacher (problem generator) and student (solver) rollouts, keeping the student-side problem-solution pairs per gradient step equal across methods.

Table 9: The training configurations for the experiments.

You are given a math problem: {Problem}
Your task is to create a math problem that is conceptually different from the provided problem. The new problem must be answerable with a numerical value or mathematical expression.
First, explain how your new problem differs conceptually from the original problem inside the <think>...</think> tags. Then, present your new problem inside the <problem>...</problem> tags. Finally, identify at most three math concepts required to solve your problem. Provide these concepts in a comma separated list inside the <concepts>...</concepts> tags.

Figure 21: Prompt for generating math problems. {Problem} is a placeholder for the reference problem sampled from the problem pool.

You are a helpful AI Assistant, designed to provide well-reasoned and detailed responses. You FIRST think about the reasoning process step by step and then provide the user with the answer. The last line of your response should be 'Therefore, the final answer is: `$\boxed{ANSWER}$`' (without quotes) where ANSWER is just the final number or expression that solves the problem.
{Problem}

Figure 22: Prompt for generating solutions to math problems. {Problem} is a placeholder for the actual problem.