Empowering Math Problem Generation and Reasoning for Large Language Model via Synthetic Data based Continual Learning Framework

Qian Wan[1,2], Wangzi Shi[1,2], Jintian Feng[1,2], Shengyingjie Liu[1,2], Luona Wei[3], Zhicheng Dai[1,4], Jianwen Sun[1,2*]

[1]Faculty of Artificial Intelligence in Education, Central China Normal University
[2]National Engineering Research Center of Educational Big Data, Central China Normal University
[3]College of Electronics and Information Engineering, South-Central Minzu University
[4]National Engineering Research Center for E-learning, Central China Normal University
{wanq8228, dzc, sunjw}@ccnu.edu.cn, {wangzi, fjt2018, lsyj}@mails.ccnu.edu.cn
wlnelysion@scuec.edu.cn

Abstract

The large language models (LLMs) learning framework for math problem generation (MPG) mostly performs homogeneous training in different epochs on small-scale manually annotated data. This pattern struggles to provide large-scale new quality data to support continual improvement, and fails to stimulate the mutual promotion reaction between generation and reasoning ability of math problem, resulting in the lack of reliable solving process. This paper proposes a synthetic data based continual learning framework to improve LLMs ability for MPG and math reasoning. The framework cycles through three stages, "supervised fine-tuning, data synthesis, direct preference optimization", continuously and steadily improve performance. We propose a synthetic data method with dual mechanism of model self-play and multi-agent cooperation is proposed, which ensures the consistency and validity of synthetic data through sample filtering and rewriting strategies, and overcomes the dependence of continual learning on manually annotated data. A data replay strategy that assesses sample importance via loss differentials is designed to mitigate catastrophic forgetting. Experimental analysis on abundant authoritative math datasets demonstrates the superiority and effectiveness of our framework.

1 Introduction

Math problems are crucial means of knowledge propagation and learning assessment in teaching, but the high cost of manual compilation can no longer meet the growing demand for personalized learning (Koncel-Kedziorski et al., 2016). The preparation of high-quality math problems requires that teachers have professional knowledge and teaching experience, since they not only need to consider multiple dimensions, including difficulty gradient, question type and knowledge point, but also should reasons detailed solving process and standard answers (Liu et al., 2024).

In view of its unique educational value and potential application, researchers have carried out in-depth studies on math problem generation (MPG) and proposed a series of rule-based or deep neural network methods (Deane and Sheehan, 2003; Polozov et al., 2015; Koncel-Kedziorski et al., 2016; Nandhini and Balasundaram, 2011; Williams, 2011; Scarlatos and Lan, 2023; Wu et al., 2022; Zhou and Huang, 2019; Wang et al., 2021; Cao et al., 2022, 2021). However, these methods depend on pre-prepared reference problems or equation templates, thus the generation process is more regarded as reconstructions and transformation of training data. The lack of promotability leads to uncontrollable and homogenization of generated problems.

Large language models (LLMs) can respond to diverse queries due to the ability of in-context learning and instruction following, providing a new way to realize the controllable generation of personalized math problems. Previous studies have shown that LLMs can be guided to generate math problems that meet the requirements based on few-shot prompting(Drori et al., 2022; Zong and Krishnamachari, 2023). This method is simple and efficient, but it relies on the quality of context, which makes it difficult to ensure generation quality and content consistency in the face of complex math problems. Other researchers embraced the mind of data-centric artificial intelligence, which injects LLMs with new expertise or capabilities through fine-tuning based on solidified model architecture. Relying on high-quality human-annotated datasets, the generation quality and controllability of LLMs have been significantly improved (Christ et al.,

[*]indicates corresponding author.

2024; Liu et al., 2024).

Despite the effectiveness of naive supervised fine-tuning, current LLM learning frameworks for MPG still exhibit notable limitations. First, the interactive relationship between MPG and math reasoning (MR) ability remains insufficiently explored. Current learning framework separates the closely related ability of MPG and MR, fails to effectively use common knowledge to improve training efficiency, and leads to the loss of completeness of the generated problems due to the lack of reliable reasoning. Second, high-cost manually annotated data struggles to support the continual learning (CL) of LLMs. Synthetic data is able to broaden the connotation boundaries of the original corpus, but there is no CL framework to consider the corresponding synthetic data algorithm. Finally, the catastrophic forgetting in CL process causes LMMs to lose previously acquired knowledge. Mitigating catastrophic forgetting is a key challenge in ensuring steady improvement of LLMs capabilities.

To address these challenges, we propose a synthetic data based CL framework to improve the MPG and MR of LLM. The framework includes three stages in each iteration: supervised fine-tuning (SFT), data synthesis, and direct preference optimization (DPO). In SFT phase, we use high-quality synthetic data (one sample includes "query-problem-solution") to fine-tune the target LLM, giving it the generation ability of math problems and corresponding solving. In data synthesis stage, we construct MPG query based on random sampling and rely on the fine-tuned LLM in this iteration to generate a large amount of initial synthetic data. We establish two mechanisms of model self-play and multi-agent cooperation to ensure the consistency and validity of synthetic data. The former significantly filters out the synthetic data with inconsistent content, while the latter directly evaluates data quality and performs secondary synthesis. According to the above two mechanisms, the synthetic data in this iteration will be divided into three groups: support the DPO in this iteration, support the SFT in the next iteration, or directly discarded. In the DPO stage, the LLM learns high-quality problem modes and standardized solving steps from positive examples, and avoids common vulnerabilities and logic errors from negative examples.

Notably, we design a data replay strategy to evaluate sample importance based on the loss differentials, and selectively incorporate important historical samples into the newly synthesized data to ensure that LLM continues to learn new knowledge while maintaining stable mastery of acquired capabilities. Extensive qualitative and quantitative analyses on authoritative datasets demonstrate that the LLM trained with our framework outperforms both general or math-specific LLMs in MPG and MR.

The main contributions of this paper are summarized as follows:

- We propose a synthetic data based CL framework for LLMs, deeply explore the mutual promotion relationship between MPG and MR, and realize the collaborative improvement of the two abilities on a single LLM.
- We propose a synthetic data method with model self-play and multi-agent cooperation mechanism. On the premise of sample validity, the dependence of LLM CL on manually labeled data is overcome.
- A data replay strategy based on loss differentials to assess sample importance is designed to mitigate catastrophic forgetting of large models accompanied by CL.
- Sufficient qualitative and quantitative conclusions on multiple authoritative datasets demonstrate the advancement and effectiveness of the proposed framework.

2 Methodology

This section presents our continual learning framework and data synthesis method using model self-play and multi-agent collaboration. The approach starts with a base language model and undergoes continual learning through iterative stages of SFT and DPO, until performance converges or reaches maximum iterations, as shown in Figure 1.

2.1 Initialization

Our approach initially assumes access to a large language model M_0, a small amount of high-quality manually annotated math problem generation data $\mathcal{D}_0^{SFT}$(data structure includes problem statement, knowledge points, difficulty level, question type, solution, and answer, see Section 3.1), and three LLM-driven agents A_1, A_2, and A_3. M_0 undergoes training and updates during the continual learning process, producing models M_1, M_2, ..., M_T, where T represents the total number of iterations.

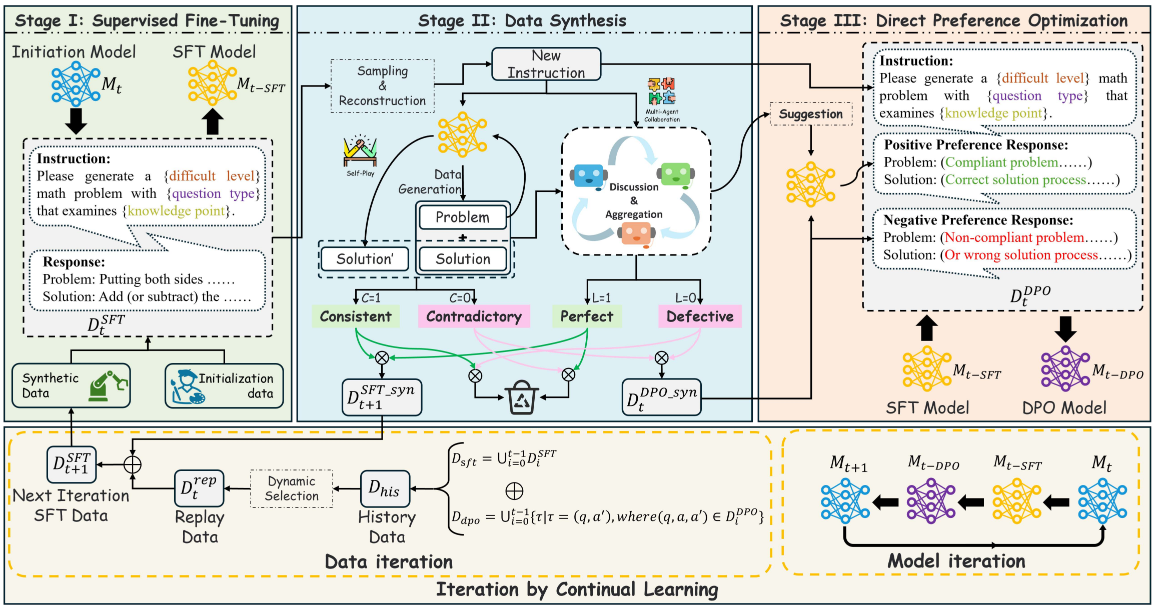

Stage I: Supervised Fine-Tuning
Initiation Model
SFT Model
M_t
M_{t-SFT}
Instruction:
Please generate a {difficult level} math problem with {question type} that examines {knowledge point}.
Response:
Problem: Putting both sides ……
Solution: Add (or subtract) the ……
D_t^{SFT}
Synthetic Data
Initialization data
Stage II: Data Synthesis
Sampling & Reconstruction
New Instruction
Self-Play
Data Generation
Multi-Agent Collaboration
Problem
Solution'
Solution
Discussion & Aggregation
C=1
C=0
L=1
L=0
Consistent
Contradictory
Perfect
Defective
$D_{t+1}^{SFT\_syn}$
$D_t^{DPO\_syn}$
Stage III: Direct Preference Optimization
Suggestion
Instruction:
Please generate a {difficult level} math problem with {question type} that examines {knowledge point}.
Positive Preference Response:
Problem: (Compliant problem……)
Solution: (Correct solution process……)
Negative Preference Response:
Problem: (Non-compliant problem……)
Solution: (Or wrong solution process……)
D_t^{DPO}
M_{t-SFT}
M_{t-DPO}
SFT Model
DPO Model
D_{t+1}^{SFT}
Next Iteration SFT Data
D_t^{rep}
Replay Data
Dynamic Selection
D_{his}
History Data
$D_{sft} = \cup_{i=0}^{t-1} D_i^{SFT}$
$\oplus$
$D_{dpo} = \cup_{i=0}^{t-1} \{\tau \mid \tau = (q, a'), where (q, a, a') \in D_i^{DPO}\}$
Data iteration
M_{t+1}
M_{t-DPO}
M_{t-SFT}
M_t
Model iteration
Iteration by Continual Learning

Figure 1: Overview of the framework

2.2 SFT Step

This stage performs supervised learning on $\mathcal{D}_t^{SFT}$ to meet task requirements. The model is fine-tuned using cross-entropy loss between predictions and labels. For dataset $\mathcal{D}_t^{SFT}$ with prompts q and responses a, the log-likelihood loss is:

$$\underset{(q,a)\sim\mathcal{D}_t^{\mathrm{SFT}}}{\mathbb{E}}\left[\sum_{\mathrm{i=1}}^{\mathrm{N}} \log \mathrm{M_t}\left(a_{\mathrm{i}} \mid a_{1:\mathrm{i}-1}, \mathrm{q}\right)\right] \qquad (1)$$

Here, M_t is the current model and M_{t-SFT} the trained model, where $\mathrm{M_t}(a \mid \mathrm{q}) = \prod_{i=1}^{N} \mathrm{M_t}\left(a_{\mathrm{i}} \mid a_{<\mathrm{i}}, \mathrm{q}\right)$ is the conditional probability of response a given q. N is the sequence length, i the timestep, and $\mathcal{D}_t^{SFT}$ evolves during training.

Initial supervised learning (i.e., t=0, T=1) establishes fundamental problem generation and reasoning capabilities, enabling subsequent synthetic data construction and DPO alignment.

2.3 Data Synthesis: Self-Play and Multi-Agent Collaboration

This section details how to construct synthetic data with high quality density by means of large language model self-play mechanisms and multi-agent collaboration. Our proposed approach ensures the quality and reliability of the generated data through two complementary validation mechanisms, which evaluate and optimize the generated math problems from different perspectives.

2.3.1 Generating New Problem Requests

Based on the distribution of annotated data $\mathcal{D}_0^{SFT}$, we construct a task space with 6 basic question types, 1200+ math knowledge points, and three difficulty levels. Using random sampling, we generate 50K unique problem instructions I per training round by combining 1-3 knowledge points, question type, and difficulty level (detailed analysis in Appendix B). This method ensures high diversity and validity of generated instructions while avoiding invalid or duplicate instructions through the introduction of randomness.

2.3.2 Building Preference Data and SFT Data

Generating Responses. First, for each instruction q in I, we utilize the supervised fine-tuned model M_{t-SFT} from each round of continual learning (see Section 2.2) to generate a complete problem sample a, which includes both the problem statement Q and detailed solution steps R_1. Subsequently, we evaluate and validate each generated math problem a through the following two mechanisms from different perspectives and approaches, continuing until all instructions in I have been utilized.

Self-Play Mechanism. The self-play mechanism (SPM) ensures data quality through adversarial generation and validation by the model itself, which acts as both question setter and solver. As a setter, the model generates problems with detailed solutions; as a solver, it independently solves the previously generated problem and compare its solution process with the original solution to verify the consistency and solvability of the generated problem.

The specific process is as follows: (1) Creating an identical but independent model copy M_{t-SFT}^{Shadow} of the current SFT-trained model. This setup simulates the role of an independent solver; (2) For the currently generated problem sample a, decomposing it into a question statement Q and solution R_1; (3) Q is provided as input to M_{t-SFT}^{Shadow}, which independently generates solution steps R_2; (4) The two sets of solutions are compared to generate a binary consistency signal $C = SPM(Q, R_1, R_2)$ (See Appendix C for details). When the two solutions are consistent, C=1; otherwise, C=0.

Multi-Agent Collaboration Mechanism. Different models can bring novel ways of thinking about knowledge reasoning, enabling consideration from multiple perspectives and capturing issues that might be difficult to identify from a single viewpoint. The multi-agent collaboration mechanism (MCM) ensures problem quality through multi-perspective evaluation by introducing three external agent models with distinct expertise, guided by different role prompts (see Appendix M).

The specific design is as follows: (1) Expert evaluation: models A_1 and A_2 independently evaluate problem a within their domains, maintaining response independence; (2) Comprehensive decision-making: based on these responses of A_1 and A_2, A_3 either synthesizes consensual perspectives into a hierarchical viewpoint system, or analyzes conflicts by identifying specific points of contention and finding reasonable compromises through detailed evaluation. Finally, A_3 generates a binary quality assessment signal L and feedback suggestions E, denoted as $L, E = MCM(q, a)$, where L=1 and E=Null for flawless problems, otherwise L=0 with E containing error descriptions and modification suggestions.

Data Optimization and Categorization. Based on the dual validation results from self-play and multi-agent collaboration, we establish an adaptive data classification framework that categorizes data into three classes (see Appendix D for data generation results from the first two rounds):

1. SFT Data: Data with consistent self-play signals and no quality assessment defects. After screening for difficulty level and removing data with incorrect formats, these samples serve as candidate SFT data for the next stage of continual learning, denoted as $\mathcal{D}_{t+1}^{SFT\_syn}$ (complete dataset also includes replay data, see Section 2.5.1): $\mathcal{D}_{t+1}^{SFT\_syn} = \{(q, a) \mid q \sim I, a \sim M_{t-SFT}(\cdot \mid q), C = 1 \text{ and } L = 1\}$.
2. Preference Data: Based on findings that LLMs can improve outputs through self-assessment or external feedback (Madaan et al., 2024; Zhang et al., 2024), we enhance data with inconsistent self-play signals and quality assessment defects by: using the solution R_2 of replica model and feedback E of expert model as contextual prompts for the current generation model M_{t-SFT} to self-improve the initial problem a, resulting in improved problem a'. After filtering format errors, these constitute the preference dataset $\mathcal{D}_t^{DPO}$ for the current round: $\mathcal{D}_t^{DPO} = \{(q, a, a') \mid q \sim I, a \sim M_{t-SFT}(\cdot \mid q), a' \sim M_{t-SFT}(\cdot \mid q, a, R_2, E), C = 0 \text{ and } L = 0\}$.
3. Abandoned data: When signals from self-play and multi-agent collaboration are inconsistent (i.e., C+L=1), we adopt a conservative strategy by directly discarding such data to ensure data quality.

Without requiring manual annotation, we ensure the high quality and reliability of data used for subsequent training. Experimental results demonstrate that this dual validation framework significantly enhances the quality of generated data, thereby improving the overall performance of model.

2.4 DPO Step

This stage enhances problem quality evaluation and reasoning through preference learning. We construct $\mathcal{D}_t^{DPO}$ with positive and negative samples (Section 2.3.2) and optimize using direct preference optimization (Rafailov et al., 2024). DPO fine-tunes the model directly on preference data, maximizing:

$$\mathop{\mathbb{E}}_{(q,a,a')\sim\mathcal{D}_t^{DPO}} \log\sigma\left(\beta\log\frac{M_{t-DPO}(a'|q)}{M_{t-SFT}(a'|q)} - \beta\log\frac{M_{t-DPO}(a|q)}{M_{t-SFT}(a|q)}\right) \tag{2}$$

where M_{t-DPO} and M_{t-SFT} are target and reference model distributions respectively, and β controls their divergence. The trained model initializes the next iteration: $M_{t+1} = M_{t-DPO}$.

2.5 Continual Learning with Synthetic Data

2.5.1 Replay Mechanism

In continual learning, models face catastrophic forgetting and efficiency challenges. Our proposed experience replay-based strategy addresses these through dynamic sampling and knowledge accumulation, ensuring stability while facilitating efficient knowledge transfer to new tasks.

History Data. The historical dataset $\mathcal{D}_{hist}$ consists of the following two sources:

1. SFT Data: Training data from SFT steps in previous training rounds: $\mathcal{D}_{sft} = \cup_{i=0}^{t-1} \mathcal{D}_i^{SFT}$.
2. DPO Data: Positive samples from preference data pairs in previous DPO training rounds: $\mathcal{D}_{dpo} = \cup_{i=0}^{t-1} \{\tau | \tau = (q, a') \,\text{where}(q, a, a') \in \mathcal{D}_i^{DPO}\}$, where τ represents the dataset constructed by removing negative samples from all preference data pairs in the D_i^{DPO} dataset. The final historical dataset is defined as $\mathcal{D}_{hist} = \mathcal{D}_{sft} \oplus \mathcal{D}_{dpo}$.

Dynamic Selection. We measure impact of new knowledge on existing knowledge through loss differentials to identify forgetting-prone historical data. For $(q, a) \in \mathcal{D}_{hist}$, the forgetting score is:

$$\mathcal{J}(q, a) = \mathcal{L}_{M_{No-Replay}}(q, a) - \mathcal{L}_{M_{t-DPO}}(q, a) \tag{3}$$

where $\mathcal{L}_{M_{t-DPO}}$ represents the loss value of model M_{t-DPO} obtained after the previous training round on this historical data; $\mathcal{L}_{M_{No-Replay}}$ represents the loss value on this data from model $M_{No-Replay}$, which is obtained by continuing to the next training round without replay after the previous round ends. If the forgetting score of this historical data exceeds the average loss μ of all historical data on model M_{t-DPO}, it is considered forgotten.

Additionally, we replay high-difficulty data where $\mathcal{L}_{M_{t-DPO}}$ exceeds μ, indicating insufficient learning in the previous round. The replay dataset combines both types: $\mathcal{D}_t^{rep} = \{(q, a) | \mathcal{J}(q, a) > \mu\} \cup \{(q, a) | \mathcal{L}_{M_{t-DPO}}(q, a) > \mu\}$. This $\mathcal{D}_t^{rep}$ and $\mathcal{D}_{t+1}^{SFT\_syn}$ form SFT dataset $\mathcal{D}_{t+1}^{SFT}$ for next iteration.

2.5.2 Continual Learning

From base model M_0, we train models M_1, M_2, M_3 over T=3 iterations (M_2 selected as experimental model, see Section 4.4). Each M_{t+1} is trained through SFT and DPO on D_t^{SFT} and D_t^{DPO} respectively (synthesis process in Section 2.3).

We define the models and their training data as follows, at iteration T (t=T-1):

$M_{t+1} = DPO(SFT(M_t, \mathcal{D}_t^{SFT}), \mathcal{D}_t^{DPO})$

where the training data $\mathcal{D}_t^{SFT} = \mathcal{D}_{t-1}^{rep} \cup \mathcal{D}_t^{SFT\_syn}$ consists of replay data and synthetic data, and $\mathcal{D}_t^{DPO}$ comprises preference pairs obtained from optimized synthetic data.

Note that the continual learning process is optional, as it may lead to performance degradation when data quality is poor or when the model experiences overfitting.

3 Experiments

3.1 Experimental Setup

Base Model. We use Qwen2-7B-Instruct and DeepSeekMath-7B-Instruct as our initial base models M_0, to validate the effectiveness of our method on both general-purpose and specialized math models.

Agent Model. We select Qwen1.5-72B-Chat, Qwen2-72B-Instruct and Qwen2.5-72B-Instruct as the external agent models A_1, A_2 and A_3 of our method, respectively (For details, see Appendix E).

SFT Data Initialization. We collect math examination and classroom test problems with their detailed solution processes from elementary to junior high school levels. Using manual annotation methods, we label the collected problems along three dimensions: question type, knowledge points, and difficulty level. Through reverse reasoning, we generate problem generation prompt for each problems based on these three annotations. Finally, we construct query-answer pairs (where query is problem generation prompt and answer is the corresponding problem with solution steps), forming the first round of SFT training data D_0^{SFT}, totaling 18K samples.

3.2 Evaluation and Metrics

Math Problem Generating. We evaluate the effectiveness of our proposed method by randomly constructing 500 unseen problem generation prompts (For details on the construction method, see Section 2.3.1). This approach is similar to Christ et al. (2024), though our evaluation sample size is twice as large. Following Liu et al. (2024), we use GPT-4o to evaluate generated problems across eight dimensions: language fluency (LF), logical correctness (LC), content completeness (CC), knowledge relevance (KR), difficulty appropriateness (DA), type adaptability (TA), analysis completeness (AC), and answer accuracy (AA), with scores from 1-10 and supporting rationales.

Math Problem Reasoning. We evaluate our method on nine English and Chinese math benchmarks (see Appendix F for detailed dataset descriptions), including widely used benchmarks like GSM8K (Cobbe et al., 2021), Math (Hendrycks et al., 2021), and various Chinese math datasets. Using evaluation scripts from Gou et al. (2023), we assess model accuracy through chain-of-thought reasoning under 0-shot settings, except for multiple-choice problems in GaoKao and CN Middle School 24 which use 5-shot settings.

3.3 Baselines

General Language Models. We select Qwen2-1.5/7B-Instruct (Yang et al.), Llama-3.1-8B-Instruct (AI@Meta, 2024), Ministral-8B-Instruct-2410 (Mistral-AI, 2024b), ChatGLM3-6B, GLM4-9B-Chat (GLM et al., 2024), Baichuan2-7/13B-Chat (Yang et al., 2023), Yi-1.5-9/34B-Chat (Young et al., 2024), and Gemma-2-9b-it (Team et al., 2024) as general model baselines.

Specialized Models Trained On Math. Additionally, we select several representative specialized models in the math domain as baselines. These include DeepSeekMath-7B-RL (Shao et al., 2024), DeepSeekMath-7B-Instruct (Shao et al., 2024), DeepSeek-Coder-V2-Lite-Instruct (Zhu et al., 2024), DeepSeek-R1-Distill-Llama-8B/Qwen-7B (Guo et al., 2025), Internlm2-math-plus-7/20b (Ying et al., 2024), Mathstral-7B-v0.1 (Mistral-AI, 2024a), and NuminaMath-7B-CoT (Beeching et al., 2024).

4 Result and Discussion

4.1 Main Results

Math Problem Generating Ability. As shown in Table 1, our method achieves excellent performance on the general model Qwen2-7B-Instruct, with an average score of 9.401 across all eight dimensions, consistently outperforming other baseline models. The model optimized by our method shows particularly outstanding performance in knowledge relevance (KR, 9.786), answer accuracy (AA, 9.982), type adaptability (TA, 9.760), and analysis completeness (AC, 9.270), significantly surpassing domain-specific baselines such as DeepSeek-Coder-V2-Lite-Instruct (KR: 9.476, AA: 9.07, TA: 9.108, AC: 8.556). The optimized model also achieves high scores in content completeness (CC, 9.11) and logical correctness (LC, 9.882), demonstrating the ability to generate well-structured math problems with clear logical progression.

Importantly, our method shows remarkable effectiveness on math-specialized models. When applied to DeepSeekMath-7B-Instruct, our approach improves its average generation score from 7.256 to 9.025, achieving a substantial improvement of 1.769 points. Notably, both general models (Qwen2-7B-Instruct) and specialized models (DeepSeekMath-7B-Instruct) trained with our method outperform recent advanced distillation models like DeepSeek-R1-Distill-Qwen-7B (9.401, 9.025 vs. 9.106) and DeepSeek-R1-Distill-Llama-8B (9.401, 9.025 vs. 7.703) across most metrics.

Model	Dimension								Avg.
	LF	LC	CC	AC	AA	KR	DA	TA	
DeepSeekMath-7B-RL	8.024	8.404	7.394	7.388	8.612	8.682	7.362	7.968	7.979
DeepSeek-Coder-V2-Lite-Instruct	8.484	8.916	8.494	8.556	9.07	9.476	8.158	9.108	8.783
DeepSeekMath-7B-Instruct	7.826	7.384	6.704	6.434	7.416	8.158	6.71	7.414	7.256
DeepSeek-R1-Distill-Llama-8B	7.572	7.497	7.333	7.265	7.717	8.773	7.473	7.998	7.703
DeepSeek-R1-Distill-Qwen-7B	8.88	9.441	8.939	8.812	9.391	9.598	8.315	9.469	9.106
Internlm2-math-plus-7b	7.314	7.348	5.996	5.884	7.708	7.904	6.466	6.096	6.839
Internlm2-math-plus-20b	7.056	7.38	6.026	5.934	7.65	7.764	6.492	6.362	6.833
Mathstral-7B-v0.1	8.336	8.634	7.54	7.84	8.814	8.622	7.378	7.656	8.103
Ministral-8B-Instruct-2410	8.448	8.486	8.11	8.232	8.432	8.83	7.682	8.628	8.356
NuminaMath-7B-CoT	7.512	6.948	6.68	6.552	6.886	7.514	6.616	7.132	6.98
Meta-Llama-3.1-8B-Instruct	7.26	6.082	6.498	6.138	5.808	7.848	6.768	7.604	6.751
Qwen2-1.5B-Instruct	7.134	5.926	5.934	5.606	5.65	7.106	6.174	6.743	6.284
Qwen2-7B-Instruct	8.526	8.54	8.536	8.376	8.494	9.392	8.134	9.112	8.639
ChatGLM3-6B	7.18	5.576	5.806	5.412	5.212	7.066	6.136	6.608	6.125
GLM4-9B-Chat	8.482	8.396	8.356	8.202	8.286	9.214	7.964	9.004	8.488
Baichuan2-7B-Chat	7.322	5.576	5.912	5.296	5.056	7.418	6.214	6.884	6.209
Baichuan2-13B-Chat	6.814	5.238	5.494	5.03	4.792	7.034	5.98	6.35	5.842
Yi-1.5-9B-Chat	8.214	7.66	7.628	7.432	7.502	8.722	7.512	8.336	7.876
Yi-1.5-34B-Chat	8.322	8.176	8.136	7.916	7.972	9.092	7.872	8.892	8.297
Gemma-2-9b-it	8.58	8.816	8.468	8.45	8.642	9.298	8.25	9.288	8.724
Our-DeepSeekMath-7B-Instruct	8.666	9.454	8.656	8.772	9.512	9.48	8.288	9.374	9.025
Our-Qwen2-7B-Instruct	**8.88**	**9.882**	**9.11**	**9.27**	**9.982**	**9.786**	**8.54**	**9.76**	**9.401**

Table 1: Math Problem Generation Results (Bold: best, Underline: second best).

Math Problem Reasoning Ability. As shown in Table 2, our method demonstrates exceptional performance across various math reasoning benchmarks. For English benchmarks, Qwen2-7B-Instruct optimized by our method achieves the best accuracy of 90.1% on GSM8K, and on the MATH, our method enables Qwen2-7B-Instruct to reach an impressive 60.7%, showing outstanding performance among 7B-parameter models. On advanced benchmarks such as Minerva Math, GaoKao 2023 En, and CollegeMath, our optimized models achieve 27.9%, 54.3%, and 39.5% respectively, approaching the performance of the best-performing models.

For Chinese benchmarks, the advantages of our method become more pronounced. On CMATH, our optimized Qwen2-7B-Instruct achieves the best result of 91.0%, surpassing DeepSeek-R1-Distill-Qwen-7B (90.0%). On CN Middle School 24, our model leads significantly with 74.3% accuracy, far exceeding the strongest baseline DeepSeekMath-7B-RL (67.3%).

Our method demonstrates broad applicability in math reasoning tasks, bringing significant improvements to both general-purpose model like Qwen2-7B-Instruct and math-specialized model like DeepSeekMath-7B-Instruct. Qwen2-7B-Instruct shows improvements of 4.4%, 15.0%, 7.5%, and 19.8% on GSM8K, College Math, CMATH, and CN Middle School 24 respectively, while DeepSeekMath-7B-Instruct also gains 3.9%, 7.7%, 4.1%, and 14.8% on these benchmarks.

These results demonstrate that training the model to generate problems with complete solutions enhances both generation and reasoning capabilities through a "teaching to learn" effect. The experiments validate our training strategy and the effectiveness of our proposed data synthesis and replay mechanisms.

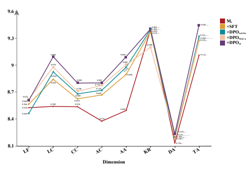
4.2 Ablation Study

As shown in Figure 2(a), our ablation experiments across eight evaluation dimensions demonstrate significant improvements after SFT training, with key metrics LC, AA, and AC increasing by 0.302, 0.402, and 0.292 points respectively. In the DPO phase, the self-play mechanism shows improvements across most dimensions except LF, while the multi-agent collaboration mechanism increases LC by 0.136 points compared to SFT. The combination of both mechanisms achieves optimal performance, with scores of 9.094 (LC), 9.402 (KR), and 9.450 (TA), validating the effectiveness of our dual-mechanism approach in ensuring data quality and diversity.

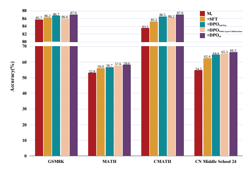
Furthermore, as shown in Figure 2(b), ablation experiments on GSM8K, MATH, CMATH, and CN Middle School 24 demonstrate that SFT improves performance by 0.5%, 3.1%, 1.7%, and 7.9% respectively. In the DPO phase, the self-play mechanism further increases accuracy by 0.5%, 0.7%, 1.3%, and 2.0%, while the multi-agent collaboration shows notable improvements particularly on MATH (1.6%) and CN Middle School 24 (2.9%). The combined mechanisms achieve optimal performance with accuracy rates of 87.0%, 58.0%, 87.0%, and 66.3% respectively, demonstrating their complementary nature.

4.3 Effect of Multi-agent Collaboration

To validate the effectiveness of our proposed multi-agent collaboration method, we conducte ablation experiments comparing three alternative approaches: (1) Single-Model Role-playing: a single large model (A_3) with different prompts; (2) Multi-Model Voting: majority voting from three models



(a)

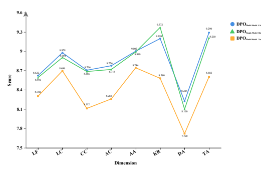


(b)

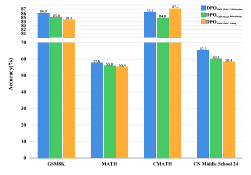
Figure 2: (a) Ablation studies in math problem generation during first round of continual learning. M_0 denotes base model, +SFT indicates SFT training on M_0, $+DPO_{Self-Play}$ represents DPO training using only Self-Play generated synthetic data, $+DPO_{Multi-agent\,Collaboration}$ indicates exclusive use of Multi-agent Collaboration, and $+DPO_{all}$ represents simultaneous use of both mechanisms; (b) Ablation studies in math problem reasoning with same setup as generation task.

(A_1, A_2 and A_3); (3) Multi-Model Collaboration: our proposed multi-agent collaboration method.

As shown in Figure 3(a), we evaluate these three methods on eight dimensions of math problem generation capability using the Qwen2-7B-Instruct model during the first round of iteration training. Additionally, we assess these methods on four math reasoning benchmarks, with results shown in Figure 3(b).



(a)



(b)

Figure 3: Performance comparison of collaboration methods on (a) math problem generation and (b) math reasoning.

The experimental results demonstrate that in the math problem generation task, our multi-agent collaboration method excels in most dimensions, achieving the highest scores in LF, LC, CC, AC, AA, DA, and TA. Only in KR did the single-model role-playing approach marginally outperform our method. In the math reasoning task, our proposed multi-agent collaboration method performs either better than or comparable to the other two methods across all four math test sets. Notably, on the CN Middle School 24 benchmark, it shows an improvement of 5.2 percentage points compared to the single-model method and 6.9 percentage points

Model	EN Benchmark						ZH Benchmark		
	GSM8K	MATH	Minerva Math	GaoKao 2023 En	Olympiad Bench	CollegeMath	GaoKao	CMATH	CN Middle School 24
DeepSeekMath-7B-RL	88.2	52.4	20.6	43.6	19.0	37.5	33.6	86.7	67.3
DeepSeek-Coder-V2-Lite-Instruct	87.6	61.0	29.4	56.1	**26.4**	**39.8**	**51.1**	89.8	66.3
DeepSeekMath-7B-Instruct	82.9	46.8	21.3	41.3	14.7	30.2	31.3	84.6	54.5
DeepSeek-R1-Distill-Llama-8B	87.0	63.3	26.5	52.2	22.7	32.2	41.2	82.8	52.5
DeepSeek-R1-Distill-Qwen-7B	89.5	**69.8**	**32.4**	**58.2**	26.4	36.4	42.4	90.0	49.5
Internlm2-math-plus-7b	84.0	54.4	17.3	50.1	18.8	36.2	34.5	82.7	32.7
Internlm2-math-plus-20b	87.9	56.5	20.2	51.9	23.1	37.5	36.1	81.3	33.7
Mathstral-7B-v0.1	84.9	56.6	16.2	46.0	21.5	33.7	31.6	76.7	42.6
Ministral-8B-Instruct-2410	87.8	55.2	23.9	47.0	20.7	32.9	44.7	82.3	61.4
NuminaMath-7B-CoT	75.4	55.2	19.1	47.5	19.9	36.9	36.4	78.2	60.4
Meta-Llama-3.1-8B-Instruct	76.6	47.2	21.7	38.4	15.4	33.8	30.4	64.8	43.6
Qwen2-1.5B-Instruct	64.1	25.1	5.5	19.7	4.1	10.4	17.0	65.5	31.7
Qwen2-7B-Instruct	85.7	52.9	19.5	36.4	21.3	24.5	35.1	83.5	54.5
ChatGLM3-6B	52.1	24.3	3.3	17.9	4.4	12.8	18.6	44.0	40.6
GLM4-9B-Chat	85.4	51.3	20.2	43.6	18.7	30.6	41.2	86.0	66.3
Baichuan2-7B-Chat	30.2	8.0	2.2	11.2	2.4	4.7	15.5	57.7	29.7
Baichuan2-13B-Chat	45.3	9.3	4.4	15.3	2.5	7.5	15.9	54.3	30.7
Yi-1.5-9B-Chat	83.6	51.8	22.4	46.0	17.0	31.7	32.3	74.5	63.4
Yi-1.5-34B-Chat	88.3	55.6	24.6	44.7	19.0	30.8	23.9	80.2	48.5
Gemma-2-9b-it	88.8	49.9	27.9	45.7	15.4	32.4	33.1	85.3	54.5
Our-DeepSeekMath-7B-Instruct	86.8	51.4	30.1	47.8	18.7	37.9	35.8	88.7	69.3
Our-Qwen2-7B-Instruct	**90.1**	60.7	27.9	54.3	19.3	39.5	41.6	**91.0**	**74.3**

Table 2: Math Problem Reasoning Results (Bold: best, Underline: second best).

compared to the majority voting method.

These results validate the effectiveness of our proposed multi-agent collaboration mechanism, demonstrating that collaborative interaction among multiple agents with distinct roles provides more comprehensive and higher-quality outputs than either single models with different prompts or simple majority voting approaches.

4.4 Effect of Continual Learning

In both math problem generation and reasoning tasks, our continual learning strategy demonstrates significant performance improvements. As shown in Table 3, for the math problem generation task, the base model M_0 achieves an average score of 8.639 across eight evaluation dimensions. After the first iteration (M_1) and second iteration (M_2), the average scores increase to 8.936 and 9.401 respectively, validating the effectiveness of continual learning in generation tasks.

Iterations	Dimension								Avg.
	LF	LC	CC	AC	AA	KR	DA	TA	
iter 0	8.526	8.54	8.536	8.376	8.494	9.392	8.134	9.112	8.639
iter 1	8.608	9.094	8.8	8.802	9.094	9.402	8.234	9.45	8.936
iter 2	8.88	9.882	9.11	9.27	9.982	9.786	8.54	9.76	9.401
iter 3	8.77	9.564	8.846	8.998	9.668	9.586	8.382	9.476	9.161

Table 3: Model performance across dimensions varies with iteration count (iter 0: base model, iter 1-n: subsequent training iterations).

As shown in Table 4, for math reasoning tasks, the base model M_0 achieves an average accuracy of 45.933% across nine benchmark tests. After the first iteration, model performance improves significantly to 52.389% (+6.456%), with MATH and CN-Middle-School-24 increasing by 5.1% and 11.8% respectively. The second iteration further enhances model capabilities, reaching an average accuracy of 55.411%, achieving improvements ranging from 2% to 8% on most benchmark tests.

We find that the third round of continual learning shows the smallest benefit, so in our main experiment we report the results of two rounds of continual learning (Detailed analysis in Appendix G).

4.5 Effect of Data Replay

As shown in Table 5, our loss-based dynamic replay strategy (Loss Replay) outperforms baseline strategies (No Replay without historical data replay and All Replay with complete historical data replay) across math reasoning benchmarks. Loss Replay achieves higher accuracy on GSM8K (89.9% vs. 86.3%, 87.3%), MATH (59.1% vs. 57.5%, 58.9%), and Minerva Math (27.9% vs. 24.3%, 25.7%). It also performs optimally on Chinese tasks CMATH (89.3%) and CN-Middle School 24 (72.3%). These results demonstrate our strategy effectively balances performance preservation and training efficiency.

4.6 Analysis of Synthetic Data Quality

We conduct quality analyses on the synthetic preference and SFT data generated by our framework (See Appendix H and I for details). For preference data, evaluation using InternLM2-7B-Reward (model details in Appendix K) shows that 86% and 84% of the improved responses surpass their initial versions in iterations 1 and 2, validating our preference data synthesis approach. For SFT data, comparison with human-written data using PPL and BERTScore shows comparable or better

Iterations	EN Benchmark						ZH Benchmark			Avg.
	GSM8K	MATH	Minerva Math	GaoKao 2023 En	Olympiad Bench	CollegeMath	GaoKao	CMATH	CN Middle School 24	
iter 0	85.7	52.9	19.5	36.4	21.3	24.5	35.1	83.5	54.5	45.933
iter 1	87.0	58.0	25.0	50.4	23.0	35.1	39.7	87.0	66.3	52.389
iter 2	90.1	60.7	27.9	54.3	19.3	39.5	41.6	91.0	74.3	55.411
iter 3	89.9	60.9	28.7	54.5	19.9	39.6	38.3	91.3	72.3	55.044

Table 4: Model performance in math problem reasoning across iterations (iter 0: base model, iter 1-n: subsequent iterations).

Method	EN Benchmark						ZH Benchmark		
	GSM8K	MATH	Minerva Math	GaoKao 2023 En	Olympiad Bench	CollegeMath	GaoKao	CMATH	CN Middle School 24
M_1	87.0	58.0	25.0	50.4	23.0	35.1	39.7	87.0	66.3
+ No Replay	86.3	57.5	24.3	47.8	21.2	37.3	38.2	87.3	61.4
+ All Replay	87.3	58.9	25.7	50.4	23.9	36.6	33.7	87.7	67.3
+ Loss Replay	89.9	59.1	27.9	52.7	20.0	39.2	40.9	89.3	72.3

Table 5: Model performance in math problem reasoning under different replay strategies (second training round).

quality (PPL: 3.01/2.37 vs 3.21, BERTScore F1: 66.22/65.42 vs 66.11 for iterations 1/2 vs human data). The embedding space visualization demonstrates overlapping distributions between synthetic and human-written data, confirming similar semantic features and high quality.

4.7 Analysis of Time Complexity and Resource Consumption

The time complexity of our framework remains asymptotically linear, expressed as $O(f_{SFT}(n) + f_{DPO}(n) + M \cdot n)$, where n is the data size and M denotes inference complexity.

In the second iteration, training required 17.2 GPU hours ($14.83), with data synthesis accounting for about 81% of resources. Overall, the process remains cost-effective relative to the performance gains (see Appendix L).

5 Related Work

This study builds upon three key research areas: math problem generation and reasoning with LLMs, synthetic data generation, and continual learning strategies.

In the field of math problem generation, research has evolved from template-based methods to neural network models, and more recently, toward LLMs. While LLMs have demonstrated promising potential in this task, challenges remain regarding consistency of quality and educational value. For math problem reasoning, researchers have employed techniques such as prompt engineering, supervised fine-tuning, and multi-agent collaboration to continually enhance the math reasoning abilities of models.

Synthetic data generation has been shown to effectively improve LLM performance, with existing approaches primarily categorized into three types: problem-driven methods, knowledge-driven methods, and model-based autonomous generation. Meanwhile, the development of continual learning strategies aims to help models acquire new knowledge without losing previously learned capabilities. To address catastrophic forgetting, researchers have proposed techniques such as parameter regularization, structural adaptation, and experience replay. However, the application of these strategies to LLMs remains an open area for further exploration.

For a more detailed review of these research areas, please refer to Appendix A.

6 Conclusion

In this paper, we propose a CL framework based on synthetic data, and deeply explore the mutual promotion relationship of LLMs on MPG and MR. Aiming at the limitation of relying on high-cost and large-scale manually annotated corpus in CL process, we design a data synthesis method by combining the model self-play and multi-agent collaboration mechanism, and realize the quality control of the synthetic data such as filtering and correction. To mitigate the catastrophic forgetting accompanied by CL process, a data replay strategy is proposed and incorporated into the data synthesis process. The replay strategy evaluates the importance of historical samples in terms of loss differentials. Through collaborative training of SFT and DPO, along with the data synthesis and replay strategy the model achieves significant performance improvements in both core tasks of MPG and MR. This work verifies the validity on multiple authoritative datasets, and provides valuable insights for developing more powerful and reliable math LLMs.

Limitations

Although our method has shown promising results and significant progress in various aspects, it is crucial to explore the potential limitations. Firstly, due to limited computational resources and time constraints, we were unable to systematically evaluate our method on larger language models, which limits our understanding of the generalizability of the method to models of different sizes. Secondly, the proposed method involves the collaboration of multiple large-scale models (including agent models of scale 72B), which places high demands on computing resources. For resource-constrained scenarios, the practicality of this method may be limited.

Acknowledgements

This work was financially supported by the National Key Research and Development Program of China (2024YFC3308200), National Natural Science Foundation of China (62437002, 62307015, 62277026, 62293554), Youth Al Talents Fund of the Chinese Association of Automation under Major Program (HBRC-JKYZD-2024-310), Hubei Provincial Natural Science Foundation of China (2024AFB169, 2023AFA020), Natural Science Foundation of Wuhan (2025040601020160), China Postdoctoral Science Foundation (2024M771108, 2023M74130), and Fundamental Research Funds for the Central Universities of South-Central Minzu University (CZQ24006).

References

AI@Meta. 2024. Llama 3 model card.

Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. 2017. Expert gate: Lifelong learning with a network of experts. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3366–3375.

Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. 2021. Rainbow memory: Continual learning with a memory of diverse samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8218–8227.

Edward Beeching, Shengyi Costa Huang, Albert Jiang, Jia Li, Benjamin Lipkin, Zihan Qina, Kashif Rasul, Ziju Shen, Roman Soletskyi, and Lewis Tunstall. 2024. Numinamath 7b cot. `https://huggingface.co/AI-MO/NuminaMath-7B-CoT`.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, Xiaoyi Dong, Haodong Duan, Qi Fan, Zhaoye Fei, Yang Gao, Jiaye Ge, Chenya Gu, Yuzhe Gu, Tao Gui, Aijia Guo, Qipeng Guo, Conghui He, Yingfan Hu, Ting Huang, Tao Jiang, Penglong Jiao, Zhenjiang Jin, Zhikai Lei, Jiaxing Li, Jingwen Li, Linyang Li, Shuaibin Li, Wei Li, Yining Li, Hongwei Liu, Jiangning Liu, Jiawei Hong, Kaiwen Liu, Kuikun Liu, Xiaoran Liu, Chengqi Lv, Haijun Lv, Kai Lv, Li Ma, Runyuan Ma, Zerun Ma, Wenchang Ning, Linke Ouyang, Jiantao Qiu, Yuan Qu, Fukai Shang, Yunfan Shao, Demin Song, Zifan Song, Zhihao Sui, Peng Sun, Yu Sun, Huanze Tang, Bin Wang, Guoteng Wang, Jiaqi Wang, Jiayu Wang, Rui Wang, Yudong Wang, Ziyi Wang, Xingjian Wei, Qizhen Weng, Fan Wu, Yingtong Xiong, Chao Xu, Ruiliang Xu, Hang Yan, Yirong Yan, Xiaogui Yang, Haochen Ye, Huaiyuan Ying, Jia Yu, Jing Yu, Yuhang Zang, Chuyu Zhang, Li Zhang, Pan Zhang, Peng Zhang, Ruijie Zhang, Shuo Zhang, Songyang Zhang, Wenjian Zhang, Wenwei Zhang, Xingcheng Zhang, Xinyue Zhang, Hui Zhao, Qian Zhao, Xiaomeng Zhao, Fengzhe Zhou, Zaida Zhou, Jingming Zhuo, Yicheng Zou, Xipeng Qiu, Yu Qiao, and Dahua Lin. 2024. Internlm2 technical report. *Preprint*, arXiv:2403.17297.

Tianyang Cao, Shuang Zeng, Xiaodan Xu, Mairgup Mansur, and Baobao Chang. 2022. Disk: Domain-constrained instance sketch for math word problem generation. *arXiv preprint arXiv:2204.04686*.

Tianyang Cao, Shuang Zeng, Songge Zhao, Mairgup Mansur, and Baobao Chang. 2021. Generating math word problems from equations with topic consistency maintaining and commonsense enforcement. In *Artificial Neural Networks and Machine Learning–ICANN 2021: 30th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 14–17, 2021, Proceedings, Part III 30*, pages 66–79. Springer.

Bryan R Christ, Jonathan Kropko, and Thomas Hartvigsen. 2024. Mathwell: Generating educational math word problems at scale. *arXiv preprint arXiv:2402.15861*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Paul Deane and Kathleen Sheehan. 2003. Automatic item generation via frame semantics: Natural language generation of math word problems.

Yuyang Ding, Xinyu Shi, Xiaobo Liang, Juntao Li, Qiaoming Zhu, and Min Zhang. 2024. Unleashing reasoning capability of llms via scalable question synthesis from scratch. *arXiv preprint arXiv:2410.18693*.

Iddo Drori, Sarah Zhang, Reece Shuttleworth, Leonard Tang, Albert Lu, Elizabeth Ke, Kevin Liu, Linda Chen, Sunny Tran, Newman Cheng, et al. 2022. A neural network solves, explains, and generates university math problems by program synthesis and few-shot learning at human level. *Proceedings of the National Academy of Sciences*, 119(32):e2123433119.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, et al. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Tora: A tool-integrated reasoning agent for mathematical problem solving. *arXiv preprint arXiv:2309.17452*.

Xiaotao Gu, Liyuan Liu, Hongkun Yu, Jing Li, Chen Chen, and Jiawei Han. 2020. On the transformer growth for progressive bert training. *arXiv preprint arXiv:2010.12562*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

G Hinton and L Van Der Maaten. 2008. Visualizing data using t-sne journal of machine learning research. *Journal of Machine Learning Research*, 9:2579–2605.

Ying Jiao, Kumar Shridhar, Peng Cui, Wangchunshu Zhou, and Mrinmaya Sachan. 2023. Automatic educational question generation with difficulty level controls. In *International Conference on Artificial Intelligence in Education*, pages 476–488. Springer.

Zixuan Ke, Bing Liu, Nianzu Ma, Hu Xu, and Lei Shu. 2021. Achieving forgetting prevention and knowledge transfer in continual learning. *Advances in Neural Information Processing Systems*, 34:22443–22456.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2. Minneapolis, Minnesota.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Rik Koncel-Kedziorski, Ioannis Konstas, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2016. A theme-rewriting approach for generating algebra word problems. *arXiv preprint arXiv:1610.06210*.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. 2024. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. 2022. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857.

Chengpeng Li, Zheng Yuan, Hongyi Yuan, Guanting Dong, Keming Lu, Jiancan Wu, Chuanqi Tan, Xiang Wang, and Chang Zhou. 2023. Mugglemath: Assessing the impact of query and response augmentation on math reasoning. *arXiv preprint arXiv:2310.05506*.

Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947.

Minpeng Liao, Wei Luo, Chengxi Li, Jing Wu, and Kai Fan. 2024. Mario: Math reasoning with code interpreter output–a reproducible pipeline. *arXiv preprint arXiv:2401.08190*.

Sannyuya Liu, Jintian Feng, Zongkai Yang, Yawei Luo, Qian Wan, Xiaoxuan Shen, and Jianwen Sun. 2024. Comet:“cone of experience” enhanced large multimodal model for mathematical problem generation. *Science China Information Sciences*, 67(12):1–2.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *ArXiv*, abs/1711.05101.

Zimu Lu, Aojun Zhou, Houxing Ren, Ke Wang, Weikang Shi, Junting Pan, Mingjie Zhan, and Hongsheng Li. 2024. Mathgenie: Generating synthetic data with question back-translation for enhancing mathematical reasoning of llms. *arXiv preprint arXiv:2402.16352*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*.

Trung Quoc Luong, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. 2024. Reft: Reasoning with reinforced fine-tuning. *arXiv preprint arXiv:2401.08967*.

Qianli Ma, Haotian Zhou, Tingkai Liu, Jianbo Yuan, Pengfei Liu, Yang You, and Hongxia Yang. 2023. Let's reward step by step: Step-level reward model as the navigators for reasoning. *arXiv preprint arXiv:2310.10080*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2024. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36.

Mistral-AI. 2024a. Mathstral model card.

Mistral-AI. 2024b. Ministral model card.

Natawut Monaikul, Giuseppe Castellucci, Simone Filice, and Oleg Rokhlenko. 2021. Continual learning for named entity recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13570–13577.

Kumaresh Nandhini and Sadhu Ramakrishnan Balasundaram. 2011. Math word question generation for training the students with learning difficulties. In *Proceedings of the International Conference & Workshop on Emerging Trends in Technology*, pages 206–211.

Oleksandr Polozov, Eleanor O'Rourke, Adam M Smith, Luke Zettlemoyer, Sumit Gulwani, and Zoran Popović. 2015. Personalized mathematical word problem generation. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Archiki Prasad, Weizhe Yuan, Richard Yuanzhe Pang, Jing Xu, Maryam Fazel-Zarandi, Mohit Bansal, Sainbayar Sukhbaatar, Jason Weston, and Jane Yu. 2024. Self-consistency preference optimization. *arXiv preprint arXiv:2411.04109*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive neural networks. *arXiv preprint arXiv:1606.04671*.

Alexander Scarlatos and Andrew Lan. 2023. Tree-based representation and generation of natural and mathematical language. *arXiv preprint arXiv:2302.07974*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Qiushi Sun, Zhangyue Yin, Xiang Li, Zhiyong Wu, Xipeng Qiu, and Lingpeng Kong. 2023. Corex: Pushing the boundaries of complex reasoning through multi-model collaboration. *arXiv preprint arXiv:2310.00280*.

Zhengyang Tang, Xingxing Zhang, Benyou Wang, and Furu Wei. 2024. Mathscale: Scaling instruction tuning for mathematical reasoning. *arXiv preprint arXiv:2403.02884*.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. 2024a. Mixture-of-agents enhances large language model capabilities. *arXiv preprint arXiv:2406.04692*.

Tianduo Wang, Shichen Li, and Wei Lu. 2024b. Self-training with direct preference optimization improves chain-of-thought reasoning. *arXiv preprint arXiv:2407.18248*.

Yifan Wang, Yafei Liu, Chufan Shi, Haoling Li, Chen Chen, Haonan Lu, and Yujiu Yang. 2024c. Inscl: A data-efficient continual learning paradigm for fine-tuning large language models with instructions. *arXiv preprint arXiv:2403.11435*.

Zichao Wang, Andrew S Lan, and Richard G Baraniuk. 2021. Math word problem generation with mathematical consistency and problem context constraints. *arXiv preprint arXiv:2109.04546*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Tianwen Wei, Jian Luan, Wei Liu, Shuang Dong, and Bin Wang. 2023. Cmath: Can your language model pass chinese elementary school math test? *arXiv preprint arXiv:2306.16636*.

Sandra Williams. 2011. Generating mathematical word problems. In *2011 AAAI Fall symposium series*.

Qinzhuo Wu, Qi Zhang, and Xuanjing Huang. 2022. Automatic math word problem generation with topic-expression co-attention mechanism and reinforcement learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:1061–1072.

Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Ce Bian, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, et al. 2023. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2 technical report. corr, abs/2407.10671, 2024. doi: 10.48550. *arXiv preprint ARXIV.2407.10671*.

Huaiyuan Ying, Shuo Zhang, Linyang Li, Zhejian Zhou, Yunfan Shao, Zhaoye Fei, Yichuan Ma, Jiawei Hong, Kuikun Liu, Ziyi Wang, et al. 2024. Internlm-math: Open math large language models toward verifiable reasoning. *arXiv preprint arXiv:2402.06332*.

Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, et al. 2024. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*.

Di Zhang, Jianbo Wu, Jingdi Lei, Tong Che, Jiatong Li, Tong Xie, Xiaoshui Huang, Shufei Zhang, Marco Pavone, Yuqiang Li, et al. 2024. Llama-berry: Pairwise optimization for o1-like olympiad-level mathematical reasoning. *arXiv preprint arXiv:2410.02884*.

Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. 2020. Class-incremental learning via deep model consolidation. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1131–1140.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Yanzhe Zhang, Xuezhi Wang, and Diyi Yang. 2022. Continual sequence generation with adaptive compositional modules. *arXiv preprint arXiv:2203.10652*.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2023. Agieval: A human-centric benchmark for evaluating foundation models. *arXiv preprint arXiv:2304.06364*.

Qingyu Zhou and Danqing Huang. 2019. Towards generating math word problems from equations and topics. In *Proceedings of the 12th international conference on natural language generation*, pages 494–503.

Zihao Zhou, Maizhen Ning, Qiufeng Wang, Jie Yao, Wei Wang, Xiaowei Huang, and Kaizhu Huang. 2023. Learning by analogy: Diverse questions generation in math word problem. *arXiv preprint arXiv:2306.09064*.

Qihao Zhu, Daya Guo, Zhihong Shao, Dejian Yang, Peiyi Wang, Runxin Xu, Y Wu, Yukun Li, Huazuo Gao, Shirong Ma, et al. 2024. Deepseek-coder-v2: Breaking the barrier of closed-source models in code intelligence. *arXiv preprint arXiv:2406.11931*.

Mingyu Zong and Bhaskar Krishnamachari. 2023. Solving math word problems concerning systems of equations with gpt-3. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 15972–15979.

A Detailed Related Work

A.1 LLM for Math Problem

In recent years, large language models have made significant progress in math problem generation and reasoning.

Math problem generation has evolved from template-based methods to neural networks, and finally to large language models. Early template methods generate through extraction or rewriting (Deane and Sheehan, 2003; Polozov et al., 2015; Koncel-Kedziorski et al., 2016; Nandhini and Balasundaram, 2011; Williams, 2011), while neural networks (Cao et al., 2021; Scarlatos and Lan, 2023; Wang et al., 2021; Wu et al., 2022; Zhou and Huang, 2019) improve generation diversity but face semantic challenges. LLM-based methods include few-shot prompting (Drori et al., 2022; Zong and Krishnamachari, 2023), which struggles with quality consistency, and instruction fine-tuning (Christ et al., 2024), which improves quality but relies on manual annotation. Despite advances, the controllability and educational value of generated problems need enhancement.

In math reasoning, researchers have developed various methods to enhance LLM capabilities, which can be broadly categorized into four approaches: (1) Prompting: activates inherent reasoning abilities, with chain-of-thought (Wei et al., 2022; Kojima et al., 2022) becoming fundamental. (2) Fine-tuning: improves problem reasoning through supervised learning and preference optimization (Ma et al., 2023; Luong et al., 2024; Shao et al., 2024; Lai et al., 2024). (3) Agent: constructs multi-agent systems (Wang et al., 2024a; Sun et al., 2023) for collaborative problem reasoning. (4) Self-optimization: leverages the self-reflection capabilities of model (Prasad et al., 2024; Wang et al., 2024b; Yuan et al., 2024; Madaan et al., 2024; Zhang et al., 2024) for iterative optimization. Additionally, more comprehensive approaches like InternLM-Math (Ying et al., 2024) integrate multiple capabilities by unifying chain-of-thought reasoning, reward modeling, formal reasoning, and code interpretation in a seq2seq format, enabling both problem-reasoning and verification functionalities in a single framework.

A.2 Synthetic Data

Research shows that high-quality synthetic data can enhance LLM on task-specific performance. In math data synthesis, researchers have developed three main approaches: problem-driven, knowledge-driven, and model-autonomous generation. Problem-driven methods build on existing data, using few-shot prompting for data transformation, such as increasing reasoning complexity through additional constraints (Luo et al., 2023) or enriching problem formulations via semantic paraphrasing (Yu et al., 2023; Lu et al., 2024; Li et al., 2023). While technically straightforward, these methods often produce data structurally similar to original samples, limiting diversity. Knowledge-driven methods extract core concepts and problem-solving strategies from seed data to construct new reasoning scenarios (Tang et al., 2024), significantly enhancing synthetic data diversity. Model-autonomous generation methods optimize inherent generation capabilities of model without seed data dependency (Ding et al., 2024), similar to our approach.

A.3 Continual Learning

Continual learning develops systems that update and acquire knowledge while preserving learned information, similar to human learning (Kenton and Toutanova, 2019). Various continual learning methods have been developed to improve language models: (1) Constraint-based methods protect key parameters, using techniques like EWC (Kirkpatrick et al., 2017) with fisher information matrix and model distillation (Li and Hoiem, 2017; Zhang et al., 2020; Monaikul et al., 2021). (2) Structural adaptation methods add task-specific parameters (Rusu et al., 2016; Gu et al., 2020; Aljundi et al., 2017), minimizing impact on old tasks but increasing model size linearly. (3) Replay methods preserve past training samples (Bang et al., 2021; Zhang et al., 2022; Wang et al., 2024c) or generate pseudo-samples (Ke et al., 2021).

While these methods show progress, they are typically based on smaller models. For LLMs, constraint-based and structural methods incur additional costs, while replay methods are widely adopted in fine-tuning due to lower overhead.

B Analysis of Problem Requests

B.1 Task Space

The task space is constructed based on three fundamental dimensions:

1. Question Types: True/False, Proof, Problem-Sets, Multiple-Choice, Problem-Solving, Fill-in-the-Blank.

2. Knowledge Points: Addition of Rational Numbers, Two-digit Number Addition, Basic Decimal Addition and Subtraction, Method of Undetermined Coefficients for Quadratic Function Expression, Multiplication of Powers with Same Base, Diagonals of a Parallelogram Bisect Each Other, Angle Measurement, Geometric Translation Transformation, Circumcenter, Triangle Inequality Theorem, Practical Applications of Percentages, Solutions of Quadratic Equations, Addition and Subtraction of Fractions with Like Denominators, Complex Word Problems Involving Fractions and Percentages, Trigonometric Values of Special Angles, Expression of Inverse Proportion Functions, Multi-digit Numbers Multiplied by Single Digits, Systems of Linear Inequalities in One Variable, Solving Proportions, Coefficient Reduction to Unity, Properties and Graphs of Linear Functions, Applications of Similar Triangles, Polygons, Construction of Cartesian Coordinate System, Real Numbers, Formula Method, Division by Single-digit Numbers, Solving Linear Inequalities in One Variable, etc.

3. Difficulty Levels: The difficulty scale (0.0-1.0) is divided into three ranges: Easy (0.1-0.3), Medium (0.4-0.6), Difficult (0.7-0.9).

B.2 Problem Requests Format

The unified instruction format is "Please generate a math problem with question type: {type}, covering knowledge points: {knowledge point combination}, and difficulty level: {difficulty}, and provide step-by-step reasoning and express the final answer enclosed in \boxed{}."

B.3 Problem Requests Scale

The initial scale of 50K instructions for synthetic data in each training round is primarily based on best practices in the fine-tuning of large language models. According to technical reports from multiple open-source models, such as Llama2 (Touvron et al., 2023), BaiChuan2 (Yang et al., 2023), and the technical presentation of OpenAI(State of GPT[1]) at the Microsoft Build 2023 conference, the ideal training dataset size for supervised fine-tuning typically ranges from 10K to 100K samples. This range has been extensively validated to achieve an optimal balance between model performance improvement and training costs.

In our methodology, the generation of synthetic data undergoes a rigorous quality control process. As shown in Figure 4, during the first round of training, approximately 54.7% of the synthetic data (around 27,350 samples) are classified as SFT data through the dual verification mechanism of self-play and multi-agent collaboration. These data require further format verification, including validation of key phrases (such as \boxed{}) and difficulty requirements. Ultimately, about 87.8% of the synthetic SFT data (approximately 24,000 samples) pass the verification process and serve as valid samples for the next round of SFT training, with subsequent rounds following a similar pattern to the first round. These statistics on data distribution and conversion rates indicate that setting the scale at 50K not only ensures sufficient high-quality SFT training samples but also prevents excessive consumption of computational resources. This choice also takes into full consideration our computational resources and associated costs. According to budget assessments, processing instruction generation and verification tasks at the 50K scale, along with subsequent model training, can be completed within an acceptable timeframe. Therefore, the scale of 50K both aligns with industry practice standards and meets our specific requirements.

[1]https://karpathy.ai/stateofgpt.pdf

C Solution Consistency Assessment in Self-Play

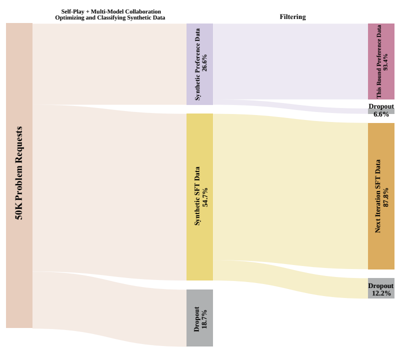
In the Self-Play mechanism, the consistency assessment of solution steps employs the same evaluation model A_3 as used in the Multi-Agent Collaboration mechanism. We design a specialized evaluation prompt (see Appendix M) that enables A_3 to conduct comprehensive comparative analysis of two solution approaches for the same problem. This evaluation process leverages expertise in math reasoning of A_3, analyzing the consistency between solution strategies, reasoning processes, and final answers, ultimately outputting a binary signal C to indicate whether there are substantial differences between the two solution steps. This assessment approach not only ensures professional and reliable evaluation but also maintains efficient processing, providing crucial criteria for subsequent data filtering.

D Data Statistics

The data synthesis process during the first two rounds of training is illustrated in Figure 4. Through self-play mechanisms and multi-agent collaboration, we categorize the 50K synthesized math problems into three groups: In the first round of training (left figure), 26.6% (approximately 13,300 samples) are further refined into synthetic preference data, 54.7% (approximately 27,350 samples) are classified as synthetic SFT data, and 18.7% (approximately 9,350 samples) are filtered out as discarded data. Subsequently, we perform format verification on the positive responses within the synthetic preference data pairs. If key phrases such as \boxed{} are missing, we filter these instances to ensure data format integrity, resulting in 93.4% of the synthetic preference data being converted into the preference dataset for this round. For the synthetic SFT data, we filter out samples with difficulty level below 0.3 and those lacking key phrases such as \boxed{}, resulting in 87.8% of the synthetic SFT data being converted into the SFT dataset for the next iteration. Similarly, the data generation process in the second round of training (right figure) follows a comparable pattern.

E Detailed of Agent Model

To establish a differentiated expert evaluation system, we employ three large-scale models with distinct characteristics as external agents: Qwen1.5-72B-Chat and Qwen2-72B-Instruct serve as A_1 and



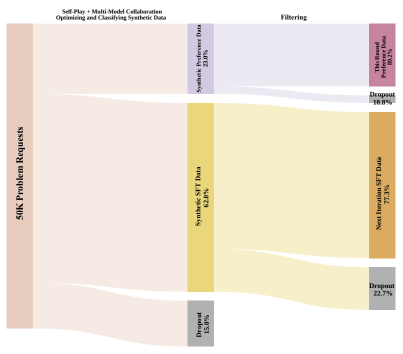


Figure 4: The synthetic data generation process during the first two rounds of training.

A_2 respectively, both excelling in open-ended dialogue and multi-turn interactions, capable of independently evaluating problem quality; Qwen2.5-72B-Instruct, as the latest version with strong comprehensive capabilities, serves as A_3, responsible for integrating the evaluations from the previous two models and providing final improvement suggestions.

F Dataset Descriptions

The benchmarks use in our experiments include:
English Benchmarks:

- GSM8K (Cobbe et al., 2021), consists of 8.5K high quality elementary school math problems created by human writers, which are divided into 7.5K training problems and 1K test problems.
- Math (Hendrycks et al., 2021), contains 12,500 challenging math competition problems, each with a detailed step-by-step solution, with the dataset divided into a training set (7,500 problems) and a test set (5,000 problems).
- OlympiadBench (He et al., 2024), contains 8952 math and physics problems from the International Olympiad, Chinese Olympiad, Chinese College Entrance Examination, and simulations.
- CollegeMath (Tang et al., 2024), the dataset extracts problems and answers from in nine college textbooks covering a variety of math topics, covering the key math disciplines of algebra, pre-calculus, calculus, vector calculus, probability, linear algebra, and differential equations.
- GaoKao 2023 En (Liao et al., 2024), consists of 385 math problems from the 2023 Chinese College Entrance Examination, professionally translated into English.
- MinervaMath (Lewkowycz et al., 2022), consists of more than 200 undergraduate-level science and math problems from the Massachusetts Institute of Technology OpenCourseWare (OCW).

Chinese Benchmarks:

- CMATH (Wei et al., 2023), a dataset containing 1,700 elementary school math applications, covering the math content of the six grades of elementary school.
- GaoKao Series:
  - GaoKao I/II 2024[2], consists of multiple-choice and fill-in-the-blank problems from the 2024 Chinese College Entrance Examination national I/II papers.
  - GaoKao-Math-QA (Zhong et al., 2023), comprises 351 multiple-choice problems from past Chinese College Entrance Examination papers.
  - GaoKao-Math-Cloze (Zhong et al., 2023), consists of 118 fill-in-the-blank questions from past Chinese College Entrance Examination papers.
  - GaoKao2024-Mix[3], consists of 91 multiple-choice and fill-in-the-blank problems, from the 2024 Chinese College Entrance Examination.

[2]https://github.com/llmeval/Llmeval-Gaokao2024-Math
[3]https://github.com/QwenLM/Qwen2.5-Math

- CN Middle School 24[3], 101 problems collect from 2024 Chinese Senior-High School Entrance Examination.

G Analysis of Iteration Rounds

As shown in Table 3 and Table 4, while the third iteration (M_3) still shows slight improvements in some metrics for both math problem generation and reasoning tasks, the overall gains are relatively limited and even show some decline. This phenomenon may be related to the high proportion of replay data in the SFT phase during the third round of training. As the number of training iterations increases, the rising proportion of replay data and insufficient learning of new data may limit the potential of model for further improvement. This observation indicates that balancing the ratio between historical and newly generated data has a significant impact on sustained model performance improvement in the continual learning process. Experimental results demonstrate that under the current training settings, two iterations can achieve optimal performance levels. Therefore, in our main experiments, we report the results of two rounds of continual learning.

H Synthetic Preference Data Quality Analysis

To validate the effectiveness of our synthesized preference data, we employed an external reward model, InternLM2-7B-Reward (Cai et al., 2024), to evaluate the preference pairs. This model assessed whether the improved responses were superior to their initial versions. As shown in Figure 5, among the preference data synthesized in the first iteration, 86% of the improved responses were evaluated as superior to their initial versions. The preference data generated in the second iteration maintained a high consistency rate of 84%. The consistently high agreement rates from an independent reward model provide strong empirical support for the reliability of our preference data synthesis method. This not only validates the ability of model to refine its outputs based on self-feedback and external feedback but also demonstrates the effectiveness of combining self-play and multi-agent collaboration mechanisms to generate high-quality preference pairs.

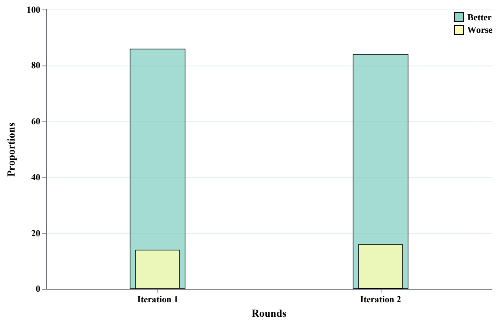


Figure 5: The effectiveness of synthetic preference pairs during the first two rounds of training.

I Synthetic SFT Data Quality Analysis

Following previous research (Christ et al., 2024; Jiao et al., 2023; Zhou et al., 2023), we employ two commonly used metrics to compare the quality of our synthesized SFT data with human-written data (5000 samples randomly selected from each): perplexity (PPL) and BERTScore. A lower PPL indicates better output, and we use Qwen2.5-72B-Instruct to calculate PPL. We utilize BERTScore (Zhang et al., 2019) to compute the semantic similarity of our synthesized SFT data and compare it with human-written data to determine whether the synthetic data achieves comparable quality to human data.

As shown in Table 6, the SFT data synthesized in the first two iterations exhibits lower PPL than human-written data, and the synthetic data performs very similarly to human-written data on BERTScore. Furthermore, similar to Ding et al. (2024), we first compute embedding vectors for both synthetic and human-written data using the text-embedding-ada-002 model, then project them into a two-dimensional space using t-SNE (Hinton and Van Der Maaten, 2008). The visualization of the embedding space, as shown in Figure 6, reveals highly overlapping cluster distributions between synthetic and human data, indicating that the synthetic data possesses similar feature distributions to human-written data in high-dimensional semantic representations.

The experimental results demonstrate that the SFT data synthesized through our method approaches or even surpasses the quality of human-written data, exhibiting high levels of language fluency, semantic consistency, and feature completeness. This validates the effectiveness and reliability of our proposed data synthesis pipeline, further sup-

Data	Metric	
	PPL	BERTScore F1
Human-labeled SFT Data	3.21	66.11
Synthetic SFT Data (Iteration 1)	3.01	66.22
Synthetic SFT Data (Iteration 2)	2.37	65.42

Table 6: Quality analysis of synthetic SFT data.PPL is perplexity, and BERTScore F1 compares the data of each dataset with the data labeled by human beings.

porting the broad applicability and practical value of synthetic data in training large language models.

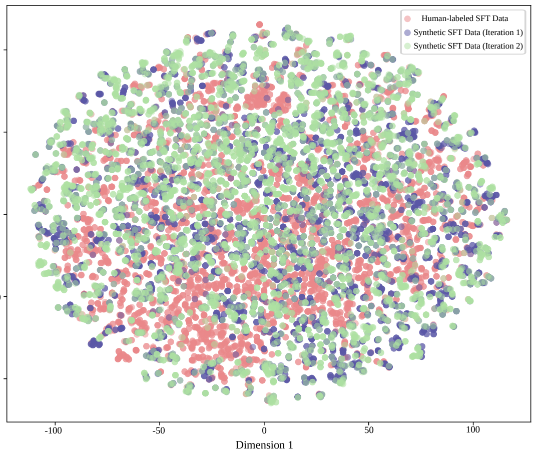


Figure 6: t-SNE visualization of synthetic SFT data from the first two rounds and human-annotated data.

J Implementation Details

In this section, we present the hyperparameters used in our experiments. We employ the AdamW optimizer (Loshchilov and Hutter, 2017) for training, with gradient clipping set to 1.0, and adopt a cosine learning rate schedule with warmup. Our final experimental model undergoes two iterations, and the hyperparameters used in different stages of each iteration are shown in Table 7.

Hyperparameters	Iteration 1		Iteration 2	
	SFT	DPO	SFT	DPO
Epochs	1.0	1.0	3.0	2.0
Learning Rate	3e-4	2e-5	9e-7	1e-5
Warm-up Ratio	0.01	0.01	0.03	0.02
Max Steps	2048	2048	2048	2048

Table 7: Training details of SFT and DPO steps.

K Details of Reward Model

InternLM2-7B-Reward is a reward model trained on InternLM2-Chat-7B-SFT. The model is trained on over 2.4 million human-annotated and AI-synthesized preference samples, covering multiple domains including dialogue, writing, poetry, summarization, coding, and math. While achieving excellent performance, it maintains a balance between practicality and safety preferences.

The InternLM2-7B-Reward model provides a comprehensive text evaluation system that can score the quality of individual texts, directly compare the relative quality of two text segments and return boolean values, and rank multiple texts by quality. Additionally, the model implements a Best-of-N sampling mechanism that can select the highest quality response from multiple candidate answers generated by language models, a feature that plays a crucial role in improving the quality of generated text.

In this study, we primarily utilize the text comparison functionality of mdoel to perform quality comparisons on each pair of preference data, determining whether the relative quality between two text segments in the data pair meets expectations based on the boolean values returned by the model. The model demonstrates stable judgment capabilities during the evaluation process, validating the effectiveness of preference data constructed through model self-improvement.

L Time Complexity and Cost

To provide a comprehensive understanding of the computational efficiency of our method, this section presents a detailed analysis of the time complexity and resource consumption of our framework.

Time Complexity Analysis The proposed iterative training framework consists of three stages: SFT (Stage I), Data Synthesis (Stage II), and DPO (Stage III). The time complexity can be formally expressed as:

1. SFT Stage: The time complexity mainly depends on the training data size n, represented as $O\left(f_{SFT}(n)\right)$, where $O\left(f_{SFT}()\right)$ characterizes the inherent time complexity function of the SFT algorithm.

2. Data Synthesis Stage: Using self-play mechanisms and a multi-agent collaborative framework:

   (a) Self-play process: Each training sample requires a single inference from the LLM, with time complexity $O(M \cdot n)$, where M represents the computational complexity of a single inference.
   (b) Multi-agent collaboration: Employs a three-agent architecture (two evaluators and one synthetic decision-maker) for data generation, with time complexity $O(3M \cdot n)$.
   (c) Overall time complexity for the synthesis stage: $O(4M \cdot n) \rightarrow O(M \cdot n)$.

3. DPO Stage: Similar to the SFT stage, the time complexity is $O\left(f_{DPO}(n)\right)$.

The overall time complexity for a single iteration is: $O(f_{SFT}(n) + f_{DPO}(n) + M \cdot n)$

For the continual learning framework with T iterations, the time complexity exhibits linear growth characteristics, introducing only a linear multiple of time overhead compared to single training. Therefore, the overall time complexity of the method is: $O(f_{SFT}(n) + f_{DPO}(n) + M \cdot n)$

Resource Consumption Analysis As shown in Table 8, we have quantified the resource consumption of our method during the second iteration of training:

These results indicate that although the data synthesis phase does require more computational resources (accounting for approximately 81% of the total GPU hours and cost), the overall resource consumption remains reasonable and cost-effective. The entire iteration requires only 17.2 GPU hours, with an estimated cost of \$14.83.

In summary, the data synthesis framework and continual learning mechanism proposed in this research demonstrate good computational efficiency characteristics. Despite introducing multi-level data reconstruction in the data synthesis stage, its time complexity maintains asymptotic linear growth. Compared to traditional single-stage data synthesis paradigms, although our method incurs some additional computational cost, experiments prove that this strategy can bring significant performance improvements, further confirming that our method achieves an effective balance between computational cost and model capability enhancement.

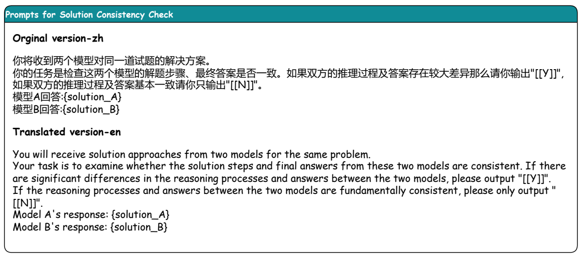
Additionally, in practice, we find that maintaining the number of iterations at T=2 can achieve good results (see Appendix G for a detailed analysis), further demonstrating that our method can efficiently improve model capabilities without requiring more time and computational resources.

Stage (Iteration 2)	Type 1	Samples	GPU hours	Cost($)
SFT	Train	40K	1.5	1.29
Data Synthesis	Infer	50K	14	12.07
DPO	Train	12K	1.7	1.47

Table 8: The resource consumption during the second iteration of training.

M Prompts

In this section, we provide the prompts that are used in this paper.



Prompts for Solution Consistency Check

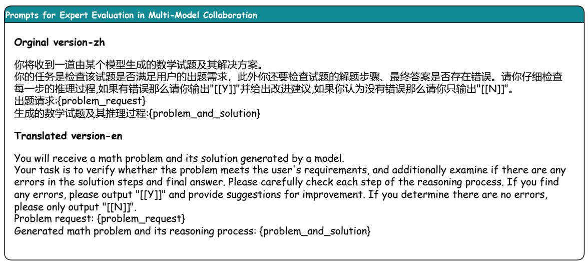
Orginal version-zh

你将收到两个模型对同一道试题的解决方案。
你的任务是检查这两个模型的解题步骤、最终答案是否一致。如果双方的推理过程及答案存在较大差异那么请你输出"[[Y]]",
如果双方的推理过程及答案基本一致请你只输出"[[N]]"。
模型A回答:{solution_A}
模型B回答:{solution_B}

Translated version-en

You will receive solution approaches from two models for the same problem.
Your task is to examine whether the solution steps and final answers from these two models are consistent. If there are significant differences in the reasoning processes and answers between the two models, please output "[[Y]]".
If the reasoning processes and answers between the two models are fundamentally consistent, please only output "[[N]]".
Model A's response: {solution_A}
Model B's response: {solution_B}

Figure 7: The prompt used in the Self-Play mechanism to determine whether two problem reasoning steps are consistent.



Prompts for Expert Evaluation in Multi-Model Collaboration

Orginal version-zh

你将收到一道由某个模型生成的数学试题及其解决方案。
你的任务是检查该试题是否满足用户的出题需求，此外你还要检查试题的解题步骤、最终答案是否存在错误。请你仔细检查每一步的推理过程,如果有错误那么请你输出"[[Y]]"并给出改进建议,如果你认为没有错误那么请你只输出"[[N]]"。
出题请求:{problem_request}
生成的数学试题及其推理过程:{problem_and_solution}

Translated version-en

You will receive a math problem and its solution generated by a model.
Your task is to verify whether the problem meets the user's requirements, and additionally examine if there are any errors in the solution steps and final answer. Please carefully check each step of the reasoning process. If you find any errors, please output "[[Y]]" and provide suggestions for improvement. If you determine there are no errors, please only output "[[N]]".
Problem request: {problem_request}
Generated math problem and its reasoning process: {problem_and_solution}

Figure 8: The prompt uses for expert evaluation in the Multi-Agent Collaboration mechanism.

Prompts for Comprehensive Decision-making in Multi-Model Collaboration

Orginal version-zh

你已获得来自各种开源模型的一组针对用户任务请求的响应。
你的任务如下:
1、通过对各个模型的回答进行比较，如果所有模型观点达成共识,那么请你将这些共识综合成一个单一的高质量响应。
2、如果各个模型间的观点存在分歧，那么请你找出这些分歧的核心原因，并且系统对分歧进行合理评判，尝试找到一个可以接受的折中方案，使回答更具包容性。
注意:在你评估过程中保留每个专家提出的独特见解，确保系统在最终答案中涵盖不同的思考维度，而不仅仅是达成共识。批判性地评估这些响应中提供的信息至关重要，认识到其中一些信息可能有偏见或不正确。您的响应不应简单地复制给定的答案，
而应提供对指令的完善、准确和全面的答复。确保您的响应结构良好、连贯，并遵守最高的准确性和可靠性标准。
来自模型的响应:
模型A:{model_A}
模型B:{model_B}
用户任务请求:{user_request}

Translated version-en

You have received a set of responses from various open-source models addressing a user's task request.
Your tasks are as follows:
1、By comparing the responses from each model, if all models reach a consensus, please synthesize these agreements into a single high-quality response.
2、If there are divergent viewpoints among the models, please identify the core reasons for these differences, systematically evaluate the disagreements, and attempt to find an acceptable compromise that makes the response more inclusive.
Note: During your evaluation, preserve the unique insights offered by each expert, ensuring the system encompasses different dimensions of thinking in the final answer, rather than merely reaching consensus. It is crucial to critically evaluate the information provided in these responses, recognizing that some information may be biased or incorrect. Your response should not simply replicate the given answers but should provide a refined, accurate, and comprehensive reply to the instructions. Ensure your response is well-structured, coherent, and adheres to the highest standards of accuracy and reliability.
Responses from models:
Model A: {model_A}
Model B: {model_B}
User task request: {user_request}

Figure 9: The prompt uses to synthesize expert decisions in the Multi-Agent Collaboration mechanism.

Prompts for Model Self-Refine

Orginal version-zh

请参考你自身的验证过程以及由专业评委给出的建议,改进或重新生成上述试题及其解题步骤,直接给出改进后的内容,不要输出其他无关内容。
自我验证过程如下:{self_verification}
专家建议如下:{expert_advice}

Translated version-en

Please refer to your own verification process and suggestions provided by professional reviewers to improve or regenerate the above problem and its solution steps. Provide only the improved content directly, without outputting any other irrelevant content.
Self-verification process: {self_verification}
Expert suggestions: {expert_advice}

Figure 10: The prompt uses to make the model self-refine the previous answer and thus construct a positive response in the preference data pair.

Prompts for Math Problem Generating Ability Evaluation

Orginal version-zh

请您充当一个公正的裁判，评估AI数学出题助手对下面显示的用户出题需求生成的试题及其解析的质量。
你的评估应考虑生成题目及其在语言流畅性（包括：数学术语与公式的使用）、逻辑正确性、内容完整性、解析完整性、答案正确性以及结合用户的需求考虑其知识点相关性、难度合理性、题型适配性，共八个方面。尽可能保持客观，请严格按照以下格式对上述八个方面进行1到10的评分：“方面-得分“，例如：语言流畅性-9。
用户请求:{user_request}
助手回答:{assistant_answer}

Translated version-en

Please act as an impartial judge to evaluate the quality of the problem and its analysis generatedby the AI Math Problem Assistant in response to the user requirements shown below.
Your evaluation should consider the generated problem and its analyses in eight aspects: language fluency (including: use of mathematical terminology and formulas), logical correctness, content completeness, analysis completeness, answer accuracy, and in relation to user requirements: knowledge point relevance, difficulty appropriateness, and question type compatibility. Maintain objectivity as much as possible, and please strictly follow the format below to score each of the eight aspects from 1 to 10: "aspect-score", for example: language fluency-9.
User request: {user_request}
Assistant's response: {assistant_answer}

Figure 11: Prompts for evaluating math problem generation capabilities of models using GPT-4o.