The Self-Improvement Paradox: Can Language Models Bootstrap Reasoning Capabilities without External Scaffolding?

Yutao Sun[1], Mingshuai Chen[1], Tiancheng Zhao[2,3], Ruochen Xu[3], Zilun Zhang[1], Jianwei Yin[1]

[1]Zhejiang University, [2]Binjiang Institute of Zhejiang University, [3]Om AI Research,

Correspondence: m.chen@zju.edu.cn, tianchez@zju-bj.com, zjuyjw@zju.edu.cn

Abstract

Self-improving large language models (LLMs) – i.e., to improve the performance of an LLM by fine-tuning it with synthetic data generated by itself – is a promising way to advance the capabilities of LLMs while avoiding extensive supervision. Existing approaches to self-improvement often rely on external supervision signals in the form of seed data and/or assistance from third-party models. This paper presents CRESCENT – a simple yet effective framework for generating high-quality synthetic question-answer data in a fully autonomous manner. CRESCENT first elicits the LLM to generate raw questions via a bait prompt, then diversifies these questions leveraging a rejection sampling-based self-deduplication, and finally feeds the questions to the LLM and collects the corresponding answers by means of majority voting. We show that CRESCENT sheds light on the potential of true self-improvement with zero external supervision signals for math reasoning; in particular, CRESCENT-generated question-answer pairs suffice to (i) improve the reasoning capabilities of an LLM while preserving its general performance (especially in the 0-shot setting); and (ii) distil LLM knowledge to weaker models more effectively than existing methods based on seed-dataset augmentation.

1 Introduction

In recent years, large language models (LLMs) such as GPT-4o (Hurst et al., 2024), Gemini (Anil et al., 2023), Llama (Touvron et al., 2023a), and DeepSeek-R1 (Guo et al., 2025) have demonstrated remarkable capabilities, revolutionizing natural language processing and various other tasks. The success of these models can be attributed to the scaling laws (Kaplan et al., 2020), which dictate the relationship between model parameters, computational resources, and training data size. For instance, the prominent performance of Llama-3.1 with 405B parameters (Dubey et al., 2024) roots in, amongst others, the massive, high-quality datasets for pre- and post-training. However, as models continue to scale, the available real-world (public) data quickly becomes exhausted; meanwhile, manually crafting high-quality data is time- and labor-intensive. Thus, data volume has become a key limiting factor for the effective scaling of new-generation models.

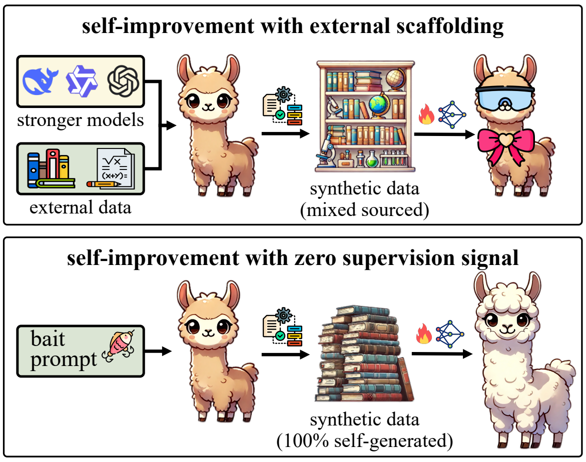


Figure 1: Different schemes of self-improvement.

In response to this challenge, synthetic data generation and data augmentation have emerged as key methods to further improve the performance of LLMs while avoiding extensive supervision. These methods leverage the ability of LLMs to mirror real-world distributions and generate high-quality, pseudo-realistic data (Zhang et al., 2023). Following this line of research, the problem of *self-improvement* naturally arises: Can we improve the performance of an LLM by fine-tuning it with synthetic data generated by itself? This problem has triggered a recent surge of research results (Wang et al., 2024). These methods, however, rely heavily on *external seed datasets* for augmentation (e.g., (Huang et al., 2023a; Wang et al., 2023b)) and/or *stronger third-party models* as classifiers or reward agents (e.g., (Le et al., 2022; Xin et al.,

Findings of the Association for Computational Linguistics: ACL 2025, pages 6501–6512
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

2024)); see Fig. 1. Such dependency on external supervision signals limits their ability to achieve true self-improvement. Orthogonally, the recently proposed method Magpie (Xu et al., 2024) suffices to generate high-quality dialogue datasets (i.e., both responses and instructions) entirely through the model itself. Nonetheless, the generated data is highly randomized and primarily dedicated to the alignment of base LLMs. Such data may improve instruction-following abilities but will degrade fundamental capabilities like math and reasoning; see (Xu et al., 2024, Sect. 6). Recent discussions (Kambhampati et al., 2024; Shumailov et al., 2024) have explicitly questioned whether genuine self-improvement is feasible, suggesting that when trained solely on self-generated data, LLMs may fail. *Can LLMs achieve true self-improvement?* remains an open question in the literature.

This paper aims to provide the infrastructure to explore the self-improvement problem of LLMs: We present CRESCENT – *a fully autonomous framework for generating high-quality synthetic question-answer (QA) data that suffice to improve the reasoning capabilities of an LLM while preserving its general performance*. CRESCENT adopts a *simple* yet *effective* workflow: (i) It uses a *bait prompt* to guide the model to generate raw questions in a specific domain, such as math word problems; (ii) It applies a *self-deduplication* mechanism based on rejection sampling (Liu and Liu, 2001) to refine and diversify the question pool; and (iii) For each question, it performs majority voting (Wang et al., 2023a) to identify the most confident answer from the model (thus *enhancing the consensus*). The so-obtained QA pairs are then used to fine-tune the original LLM via, e.g., supervised fine-tuning (SFT), to improve its math-reasoning capability.

Experiments with CRESCENT demonstrate evident self-improvement of LLMs consistently for three benchmarks on mathematical word problems in both 0-shot and 5-shot settings, without trading off their general capabilities. The improvement is especially prominent for the 0-shot case, thus improving the generalization ability of the model to real-world tasks. Ablation studies further demonstrate the superiority of CRESCENT over Magpie (Xu et al., 2024) in the generation of themed data: the latter tends to generate math-related dialogues, e.g., “Could you tell me what type of mathematics you like?” – rather than proper mathematical problems. Moreover, our experiments show that CRESCENT can serve as a highly effective and efficient distillation method, surpassing the baselines using external data and stronger models.

Contributions. Our main contributions include:

- We present a simple yet effective framework CRESCENT – utilizing the techniques of bait prompting, diversification, and consensus enhancement – to investigate the self-improvement problem of LLMs.

- We show that CRESCENT-generated QA pairs suffice to improve the reasoning capabilities of an LLM with zero supervision signals while preserving its general performance, thereby providing an affirmative answer to the self-improvement problem in the domain of mathematical reasoning (math word problems).

- Experiments demonstrate significant improvements achieved by CRESCENT compared to multiple prompting methods. As a by-product, we show CRESCENT facilitates more effective LLM knowledge distillation than existing approaches based on seed-dataset augmentation.

2 The CRESCENT Approach

This section presents CRESCENT – a framework for controlled QA self-generation via diversification and consensus enhancement. CRESCENT suffices to generate high-quality domain-specific QA pairs leveraging only the model itself, with zero external data, nor assistance from third-party models.

Fig. 2 sketches the general workflow of CRESCENT, which consists of three main steps: (I) *Bait prompting*: We use a bait prompt to instruct the original, aligned LLM to produce a set of raw questions within a specific domain; (II) *Diversification*: The raw questions may be semantically analogous to each other (as per some similarity metric), and thus we employ a rejection sampling mechanism to attain a diverse pool of representative questions through self-deduplication; (III) *Consensus enhancement*: We treat the generated questions as query prompts and feed them back to the LLM. Then, by majority vote, we obtain the final set of synthetic QA pairs. We show that such QA pairs are of high quality in the sense that they suffice to improve the domain-specific capabilities (mathematical reasoning, in our case) by fine-tuning the original LLM with these QA pairs while preserving its general capabilities.

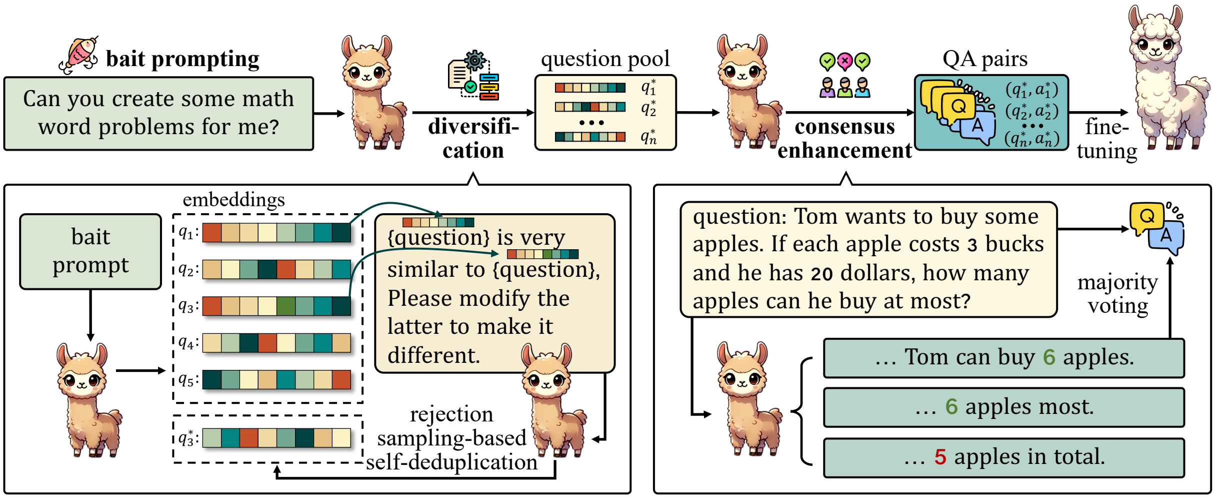

bait prompting
Can you create some math word problems for me?
diversification
question pool
q_1^*
q_2^*
q_n^*
consensus enhancement
QA pairs
(q_1^*, a_1^*)
(q_2^*, a_2^*)
(q_n^*, a_n^*)
fine-tuning
embeddings
bait prompt
q_1:
q_2:
q_3:
q_4:
q_5:
q_3^*:
{question} is very similar to {question}, Please modify the latter to make it different.
rejection sampling-based self-deduplication
question: Tom wants to buy some apples. If each apple costs 3 bucks and he has 20 dollars, how many apples can he buy at most?
majority voting
… Tom can buy 6 apples.
… 6 apples most.
… 5 apples in total.

Figure 2: The general workflow of CRESCENT in mathematical reasoning.

Below, we first present the technical details of Steps (I) to (III) and then provide the rationale behind the self-improvement achieved by these steps.

2.1 Question Generation (Steps (I) and (II))

We begin by utilizing a simple *bait prompt* to elicit the LLM to generate a bunch of domain-specific questions, such as math word problems illustrated in Fig. 2, denoted as *raw questions*. As some of them may be semantically analogous to each other, we optimize diversity of the questions in an iterative manner: Each generated question is vectorized and compared against the (embeddings of) other questions. If there exists a question that is deemed sufficiently similar (i.e., the similarity score is below a prescribed threshold), we apply the following *deduplication prompt* to modify it:

> {question} is very similar to {question}, please modify the latter to make it different.

This iterative process ensures that the question pool remains diverse and representative across the specific domain through redundancy-aware selection.

Formally, the question-generation phase can be described as follows: Let $Q = \{q_1, q_2, \ldots, q_n\}$ be the set of raw questions generated by the LLM per the bait prompt. For each question q_i, we embed it as a real-valued vector v_i and compare it against the vector representations $\{v_1, v_2, \ldots, v_{i-1}\}$ of the previously generated questions. The *similarity* between the two questions is determined by the distance between their respective vector embeddings in the inner product space, e.g., the L^2 distance. If the distance is below a given threshold θ, then q_i with $(i > j)$ is considered as a *duplicate* and thus needs to be modified via the deduplication prompt, i.e.,

$$\text{If } d\left(v_i, v_j\right) < \theta \text{ then } q_i^* = \text{Deduplicate}\left(q_i\right). \quad (\dagger)$$

Such similarity-based deduplication incorporates the *maximal marginal relevance* (MMR) criterion (Carbonell and Goldstein, 1998) to minimize repetition while preserving content relevance. Moreover, the iterative refining process falls into the paradigm of *rejection sampling* (cf. e.g., (Liu and Liu, 2001)), which ultimately yields a diversified question pool featuring relevance and representativeness w.r.t. the target domain with negligible redundancy; see Section 2.3.

2.2 Answer Generation (Step (III))

Let $Q^* = \{q_1^*, q_2^*, \ldots, q_n^*\}$ be the deduplicated set of questions generated through the previous step. The phase of answer generation aims to synthesize the corresponding high-quality answers w.r.t. each $q_i^* \in Q^*$. We achieve this by means of *consensus enhancement*, namely, we feed each question q_i^* back to the LLM and collect m *independently* produced answers, denoted by the set $A_i = \{a_1, a_2, \ldots, a_m\}$, where each a_j contains integrated chain-of-thought (CoT) processes (Wei et al., 2022) generated for question q_i^*. We then select the final answer a_i^* for question q_i^* using *majority voting* (Wang et al., 2023a). That is, we first identify the set $\bar{A}_i$ of *most frequent answers*:

$$\bar{A}_i \triangleq \left\{ a_j \in A_i \;\middle|\; f\left(a_j\right) = \max_{a_k \in A_i} f\left(a_k\right) \right\},$$

where $f(a_j)$ denotes the *frequency* (i.e., the number of occurrences) of answer a_j in A_i. Then, we

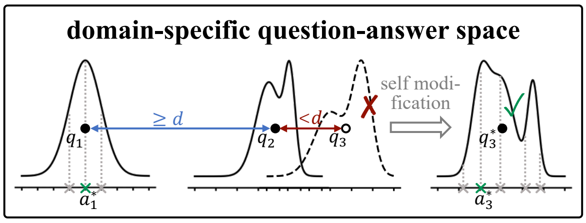


Figure 3: The intuition of CRESCENT. Let the black dots be question embeddings and distribution curve be conditional answer distribution. (1) Our diversification step modifies question samples violating the minimal distance criterion per (†) (the middle plot). (2) the consensus enhancement step selects the majority mode answer. (the green X in the left and right plots.)

uniformly sample an answer from $\bar{A}_i$ as the final answer a_i^* paired with question q_i^*. By repeating the majority voting procedure for every question, we obtain the final set of synthetic QA pairs:

$$(Q^*, A^*) = \{(q_1^*, a_1^*), (q_2^*, a_2^*), \ldots, (q_n^*, a_n^*)\} .$$

2.3 Rationale for Self-Improvement

Next, we provide the intuition on why self-generated QA pairs using the CRESCENT framework can be used to improve the capabilities of the underlying LLM. This observation will be further justified by extensive experiments in Section 3.

The intuition is three-fold (see Fig. 3):

(i) *Relevance by bait prompting*: The initial bait prompt restricts the considered space of questions and answers to a specific domain and hence all the generated QA pairs within the CRESCENT scope are pertinent to this domain.

(ii) *Diversity by rejection sampling-based deduplication*: Our diversification step explores the question space while maintaining a minimal pair-wise distance to alleviate redundancy. This is achieved by a rejection sampling loop where question samples violating the distance criterion per (†) are modified and, therefore, the generated questions exhibit a scattered distribution stretching over the space.

(iii) *Accuracy by majority voting*: Based on the observation that a complex reasoning problem typically admits multiple distinct ways of thinking yielding its unique correct answer (Wang et al., 2023a), our consensus enhancement step selects, for each question, the most frequent answer that may coincide with the correct one with high likelihood.

As a consequence, fine-tuning the original LLM with the so-obtained QA pairs will strengthen its domain-specific capabilities by *enforcing a reduction in the variance of answer generation for a diverse set of domain-relevant questions*.

3 Experiments

3.1 Experimental Setups

Benchmarks. We adopt three benchmarks on math word problems (MWPs): (i) **GSM8K** (Cobbe et al., 2021): 8.5K grade school math problems with step-by-step solutions; (ii) **ASDiv** (Miao et al., 2020): 2,305 diverse MWPs covering multiple difficulty levels; and (iii) **GSM-Plus** (Li et al., 2024): an enhanced version of GSM8K with 12K problems incorporating robustness checks. In order to accelerate the evaluation, we use **GSM-Plus-mini** – a subset of GSM-Plus containing 2,400 questions. It should be noted that the GSM-Plus-mini and GSM8K datasets do not overlap.

Baseline Models. We conduct self-improvement experiments with four different LLM models: (i) Llama2-7B-Chat (Touvron et al., 2023b): a instruction-tuned version of Llama2-7B; (ii) Llama2-13B-Chat, the 13B instruction-tuned counterpart from the same LLaMA2 series; (iii) Llama3.2-3B-Instruct, the 3B instruction-tuned model from the updated LLaMA3.2 series; and (iv) Llama3-8B-Instruct: the instruction-tuned version of Llama3-8B (Dubey et al., 2024)

Generation Configurations. For each model, we generate MWP QA pairs following these settings:

Question Generation: Bait prompt: *"Generate a diverse math word problem requiring multi-step reasoning"*. We generate 50K candidate questions for Llama2 models and 75k for Llama3/Llama3.2 models, with temperature $T = 0.95$. Diversification: We use sentence embeddings generated by the `all-MiniLM-L6-v2` model from the Sentence-BERT (Reimers and Gurevych, 2019) family; we eliminate semantically similar questions using the L^2 distance with threshold $\theta = 0.25$. We employ FAISS (Douze et al., 2024) to accelerate vector computation and comparisons.

Answer Generation: For each question, sample 5 answers with temperature $T = 0.95$, then select the most frequent answer as the final answer. We use the same answer generation settings for both models. We use the vLLM (Kwon et al., 2023) inference framework for both generation stages.

Model	Training	0-shot			5-shot		
		GSM8K	ASDiv	GSM+	GSM8K	ASDiv	GSM+
Llama2-7B-Chat	Original	18.8	41.7	11.3	23.0	**45.9**	13.5
	CRESCENT	**23.2**	**46.0**	**13.0**	**25.1**	45.2	**14.8**
Llama2-13B-Chat	Original	27.9	49.1	17.0	35.7	**49.1**	20.3
	CRESCENT	**30.9**	49.1	**20.1**	**36.3**	49.0	**21.9**
Llama3.2-3B-Inst.	Original	27.8	58.1	44.1	64.7	**61.3**	47.8
	CRESCENT	**52.2**	**60.1**	**47.1**	**66.1**	60.2	**48.2**
Llama3-8B-Inst.	Original	34.5	43.6	23.1	75.8	62.3	51.2
	CRESCENT	**63.3**	**65.9**	**48.6**	**77.6**	**63.8**	**52.8**

Table 1: Main results comparing original models vs. CRESCENT versions. Best results in **bold** (accuracy %).

GPU hours: It took 30.0 GPU hours to generate 75k QA pairs with Llama3-8B-Instruct and 42.9 GPU hours for the 50k pairs with Llama2-7B-Chat.

SFT Implementation. Our SFT procedure uses single-epoch training with max sequence length of 2,048 tokens. Optimization is performed using AdamW (Loshchilov and Hutter, 2019) ($\beta_1 = 0.9, \beta_2 = 0.95$) under a linear learning rate schedule (initial LR = 1e-5, 3% warm-up), and the batch size is set to 128 through 8-way parallelization on NVIDIA A100-80GB GPUs with 16-step gradient accumulation. We use DeepSpeed Stage3 (Rasley et al., 2020) and `bfloat16` for mitigating memory constraints, and FlashAttention-2 (Dao, 2024) for efficient attention computation.

Evaluation Protocol. We use LM-Evaluation-Harness (Gao et al., 2024) library; all datasets are evaluated under **0-shot** and **5-shot** settings. Few-shot examples are randomly selected from training sets, excluding test samples. We use two *answer extractors*: one identifies the number appearing after "####" and the other extracts the last number in the output. An answer is considered correct if either of the extractors retrieves the correct answer.

3.2 Main Results

The experimental results shown in Table 1 validate our core hypothesis: *self-generated reasoning QA pairs – boosted through diversification and consensus enhancement – enable model improvement without external supervision signals.* For GSM8K, Llama2-7B-Chat shows improvements of +4.4%↑ (0-shot) and +2.1%↑ (5-shot), while Llama3-8B-Instruct achieves noticeable gains of +28.8%↑ (0-shot) and +1.8%↑ (5-shot). Similar observations apply consistently to ASDiv and GSM-Plus-mini featuring different QA distributions, and similar trends are also observed on models of different scales, including both the 3B and 13B variants.

It is noteworthy that CRESCENT leads to *substantial improvements in the 0-shot* setting across all three datasets, with performance on certain datasets surpassing even the 5-shot counterparts for the original models. This observation highlights the potential of 0-shot learning in reducing dependency on task-specific examples, thus indicating better generalization to real-world unseen problem types.

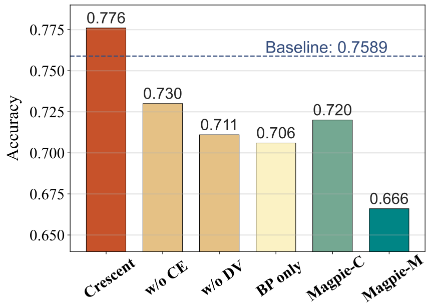


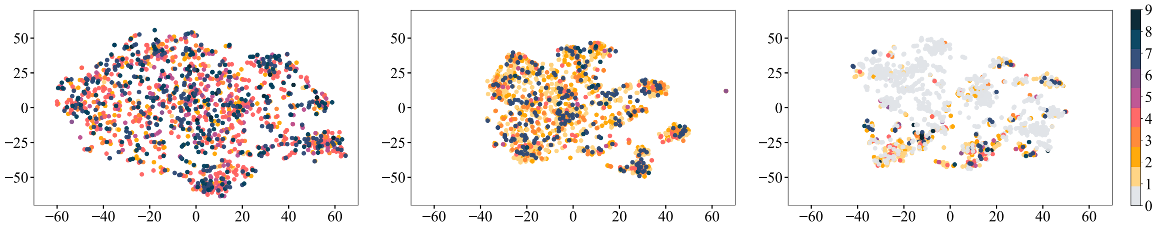
Figure 4: Accuracies w.r.t. the ablation study.

3.3 Ablation Study

To justify the pivotality of CRESCENT's core components, we conduct comprehensive ablation experiments over Llama3-8B-Instruct under 5-shot GSM8K evaluation. As depicted in Fig. 4, (i) full method of CRESCENT achieves accuracy of 77.6%, outperforming all ablated variants and the baseline; (ii) removing consensus enhancement (w/o CE) reduces performance to 73.0% (-4.6%); (iii) excluding diversification (w/o DV) yields a more severe drop to 71.1% (-6.53%); (iv) using only bait prompting (BP only) results in 70.6% (-7.0%). The results demonstrate the significance of both diversification and consensus enhancement.

Notably, CRESCENT surpasses the Magpie variants by substantial margins: (i) +5.6% over Magpie–Common (Magpie-C) (72.0%); (ii) +11.0% over Magpie-Math (Magpie-M) (66.6%).

To investigate the discrepancy between CRESCENT and Magpie-Math, we conduct a sampling analysis on the mathematical questions generated by CRESCENT, CRESCENT w/o DV, and Magpie-Math: For each method, we randomly sample 1,500 questions; Each question is then classified by difficulty using GPT-4o (Hurst et al., 2024), vectorized with the `all-MiniLM-L6-v2` embedding model, and projected into a two-dimensional plane using t-SNE (Van der Maaten and Hinton, 2008). The visualization in Fig. 5 suggests that, even without diversification, CRESCENT can still generate high-quality mathematical questions, albeit with reduced diversity and difficulty (Fig. 5b). In contrast, the vectors for Magpie-Math problems (Section 3.3)



(a) CRESCENT (b) CRESCENT w/o DV (c) Magpie-Math

Figure 5: T-SNE visualization of synthetic math questions. Points colored from 1 to 9 represent mathematical questions with increasing difficulty; Gray marks math-related questions (rather than actual mathematical problems).

Benchmark	#shots	before	after	Δ
ARC-C	0	52.9	52.3	0.6↓
MMLU	5	65.6	65.9	0.3↑
IFEval	-	50.9	52.5	1.6↑
HellaSwag	5	77.9	77.2	0.7↓
GPQA	0	31.2	31.5	0.3↑

Table 2: General capability before/after CRESCENT (%).

feature (i) a more agglomerate form exhibiting significantly low coverage than CRESCENT; and (ii) numerous gray points signifying non-mathematical problems; they are merely instructions related to the mathematics topic, e.g., *"Could you tell me what type of mathematics you like?"*. The latter aligns with the observation in (Xu et al., 2024, Sect. 6) stating that Magpie-generated dialogues may degrade math and reasoning capabilities.

4 Detailed Analysis of CRESCENT

4.1 General-Capability Preservation

Will CRESCENT *incur catastrophic forgetting of general capabilities?* We address this problem by evaluating Llama3-8B-Instruct before and after CRESCENT on five non-mathematical benchmarks covering *commonsense reasoning* (ARC-C (Clark et al., 2018), HellaSwag (Zellers et al., 2019)), *general knowledge preserving* (MMLU (Hendrycks et al., 2021)), *instruction following* (IFEval (Zhou et al., 2023)), and *graduate-level question answering* (GPQA (Rein et al., 2023)). We use the CRESCENT checkpoint directly from Section 3.2.

Table 2 shows that the CRESCENT-enhanced model exhibits performance comparable to that of the original model in all five tasks. This observation reveals that domain-specific self-enhancement through CRESCENT does not compromise general capabilities, a critical advantage over fine-tuning approaches using external data, which often exhibit significant capability trade-offs (Luo et al., 2023).

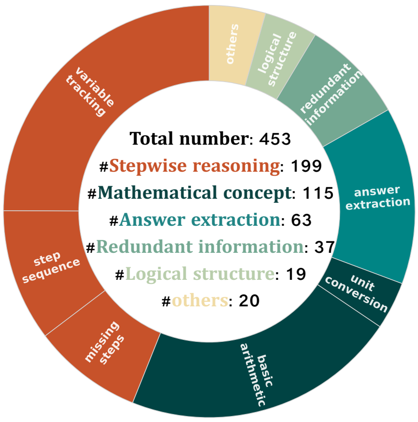


Figure 6: Breakdown of the corrected questions after applying CRESCENT in the 0-shot setting.

4.2 Analysis of Corrected Questions

Our results show significant improvements in the 0-shot setting. However, does this improvement reflect better generalization, or is it due to the lack of formatting constraints in GSM8K's 0-shot evaluation, which can lead to incorrect answer extraction? To investigate, we analyze Llama3-8B-Instruct's 0-shot results before and after applying CRESCENT, focusing on questions that were incorrect before but correct after **(corrected questions)**. We use GPT-4o to classify and analyze these errors.

Fig. 6 shows the total number of corrected questions is 453. 390 (86%) of them are due to genuine improvement in mathematical reasoning ability. These corrected questions can be further broken down into the following: (i) **Stepwise reasoning:** 199 questions (44%) had errors in stepwise reasoning due to variable tracking (113), step sequence issues (47), and missing steps (39); (ii) **Mathematical concept:** 115 questions (25%) involved fundamental math errors, with 98 attributed to calculation mistakes and 17 to unit conversion failures; (iii) **Redundant information:** 37 questions (8%) were impacted by irrelevant information in the problem statement; (iv) **Logical structure:** 19

Method	0-shot	5-shot
Standard prompt	34.5	75.8
Standard prompt + SC	37.8	75.6
Random rephrased	36.9	75.8
CoT prompt	43.6	76.0
Optimized prompt	45.1	75.7
CRESCENT + standard	**63.3**	**77.6**
CRESCENT + optimized	**69.8**	**77.1**

Table 3: Comparison with prompting methods (%).

questions (4%) involved errors in logical reasoning, such as issues with propositions or set operations; (v) **Other errors:** 20 questions (4%) were due to other miscellaneous error types.

Meanwhile, there are 63 (14%) corrected questions due to a better output format. After fine-tuning with CRESCENT-generated QA pairs, these questions are correctly answered without generating redundant content, indicating that CRESCENT's high-quality QA data also improves the model's instruction-following capability.

4.3 Comparison with Prompt Engineering

Can prompt techniques achieve a similar performance with CRESCENT*?* We address this question by comparing CRESCENT-trained Llama3-8B-Instruct against five prompting methods: (i) **Standard prompt** from Llama3 official repository;[1] (ii) **Standard prompt with self-consistency** (SC, aka majority voting) following the settings in (Wang et al., 2023a); (iii) **Random rephrased** utilizes GPT-4o to randomly rephrase the standard prompt five times (where we select the best evaluation result). Considering the answer-extractor failures discussed in Section 4.2, we carefully craft *each instruction* to control the output format, such as requesting the answer to *be placed after "####"* or *at the end of the output*, ensuring that the prompt includes relevant formatting information compatible with our answer extractor when rephrased by GPT-4o; (iv) **CoT prompt** following the settings in (Wei et al., 2022); (v) **Optimized prompt** by integrating CoT, the best candidate from random rephrased, and the SC process.

The comparison results are reported in Table 3. Overall, 0-shot outcomes demonstrate higher sensitivity to prompt variations compared to 5-shot configurations. For the original model, the optimized prompt achieves optimal performance, improving 0-shot accuracy by 10.6% over standard prompts while exhibiting comparable 5-shot results. However, this result remains *substantially inferior* (-18.2%) to CRESCENT using only standard prompts. Notably, when employing the same optimized prompts, the CRESCENT-enhanced model further improves 0-shot performance by 6.5%.

Method	Random rephrased trials					Mean	Std σ
	T1	T2	T3	T4	T5		
Original	29.9	19.9	28.6	36.9	24.4	27.9	5.69
CRESCENT	64.9	63.3	64.6	67.8	66.1	65.3	1.52

Table 4: 0-shot robustness w.r.t. rephrased prompts (%).

The observed performance gap substantiates that the improvements achieved by CRESCENT *cannot be replicated* through prompting techniques. Moreover, in random rephrased experiments (cf. Table 4), CRESCENT demonstrates *superior robustness* across five different prompts, exhibiting consistent performance with 37.4% higher accuracy and much lower standard deviation. This result indicates that CRESCENT not only enhances *domain-specific proficiency*, but also establishes *prompt-agnostic generalization* in 0-shot scenarios.

4.4 Sensitivity Analysis of Bait Prompts

To assess the sensitivity of CRESCENT to the choice of bait prompt, we conducted a controlled experiment using five prompt variants to regenerate synthetic training data. The original prompt from our main experiment was rephrased twice using GPT-4o, yielding three semantically equivalent prompts: (i) **Prompt 1 (original):** *"Generate a diverse math word problem requiring multi-step reasoning"*; (ii) **Prompt 2:** *"Create a varied math word problem that involves multiple steps of reasoning"*; (iii) **Prompt 3:** *"Design a multi-step reasoning math problem with diverse content"*.

Additionally, we tested two contrastive variants: (i) **Simple Prompt:** *"Write a math word problem"*; (ii) **Long Prompt:** *"Please generate a collection of math word problems that are not only diverse but also involve multiple steps, real-world context, and clear solution paths. Avoid duplication, ambiguity, or overly simplistic examples. "*.

For each prompt, we generated 75k synthetic samples and fine-tuned Llama3-8B-Instruct on the resulting dataset, holding all other settings constant. Evaluation was conducted on GSM8K and GSM+ in both 0-shot and 5-shot settings.

As the results shown in Table 5, the three semantically similar prompts yielded nearly identi-

[1] https://github.com/meta-llama/llama-cookbook

Model	Training	0-shot		5-shot	
		GSM8K	GSM+	GSM8K	GSM+
Llama3-8B-Inst.	Original	34.5	23.1	75.8	51.2
	Prompt 1	63.3↑	**48.6**↑	**77.6**↑	52.8↑
	Prompt 2	**69.7**↑	**48.6**↑	76.3↑	52.5↑
	Prompt 3	64.5↑	47.0↑	76.7↑	**52.9**↑
	Simple	43.6↑	39.0↑	74.5↓	46.3↓
	Long	60.6↑	43.1↑	75.6↓	52.0↑

Table 5: Results of different prompting styles (%).

	Training	MathQA	Ceval-Mid	Ceval-High	Ceval-Prog
0-shot	Original	40.1	36.8	11.1	62.1
	CRESCENT	**42.2**	**47.4**	11.1	**64.9**
5-shot	Original	42.2	36.8	11.1	64.8
	CRESCENT	**43.1**	**47.0**	11.1	64.8

Table 6: Evaluation on extended benchmarks (%).

cal results, indicating that CRESCENT is robust to stylistic variation as long as the prompt specifies the need for diverse, multi-step reasoning. In contrast, both under-specified (Simple Prompt) and overly elaborate (Long Prompt) formulations led to performance degradation. These findings confirm that CRESCENT does not rely on a narrowly optimized prompt; instead, it benefits from reasonably informative prompts aligned with the model's generative capacity.

4.5 Robustness Evaluations of CRESCENT

To further assess the robustness and generalizability of CRESCENT, we extended our evaluation to a broader range of math-related tasks. Specifically, we tested CRESCENT-trained Llama3-8B-Inst. on MathQA (Amini et al., 2019) and Ceval (Huang et al., 2023b) (including middle school math, high school math, and college-level programming).

As shown in Table 6, CRESCENT consistently improved performance on tasks aligned with its training distribution, such as MathQA and Ceval-Middle School Math. For more challenging benchmarks like Ceval-High School Math—which emphasize formal proofs and advanced topics such as geometry and number theory—performance remained stable, indicating no negative transfer. Notably, the gains on Ceval-Programming and the fact that Ceval is a Chinese-language benchmark suggest that CRESCENT's benefits extend across both task domains and languages, despite being trained exclusively on English data.

We also applied CRESCENT to Qwen2.5-Math-7B-Inst. (Yang et al., 2024b), a strong math-specialized model. Using this model to generate 50k new math QA pairs, we fine-tuned it on its own outputs using the same setup as in Section 3. Result is shown in Table 7.

Model	Training	0-shot		5-shot	
		GSM8K	GSM+	GSM8K	GSM+
Qwen2.5-Math-7B-Inst.	Original	41.4	17.0	89.2	20.3
	CRESCENT	**70.8**	**20.1**	**89.9**	**21.9**

Table 7: Result of a math-specialized model (%).

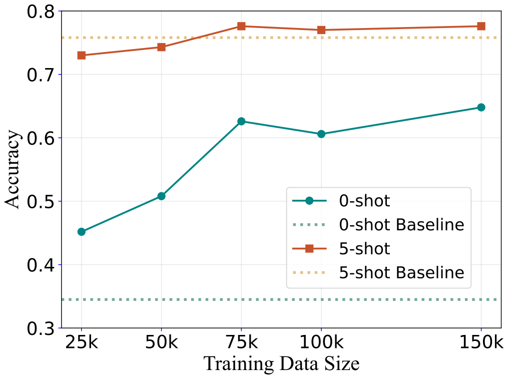


Figure 7: Accuracy in terms of synthetic data volume.

Despite being a specialized model already optimized for mathematical reasoning, Qwen2.5-Math. still benefited substantially from CRESCENT-based fine-tuning. These results reinforce the conclusion that CRESCENT enables robust self-improvement, even for strong expert models, and generalizes across task difficulty, domains, and languages.

4.6 Data Efficiency and Training Dynamics

Next, we investigate *the effect of self-improvement in terms of the volume of synthetic data and the number of training epochs.*

Data Volume: We perform one epoch of SFT using Llama3-8B-Instruct on CRESCENT data with data volumes of 25k, 50k, 75k, 100k, and 150k; we use the standard prompt for evaluation. As shown in Fig. 7, the model's performance improves consistently from 25k to 75k, but stabilizes between 75k and 150k, suggesting an upper limit to the improvement gained from increasing data volume.

Training Epochs: We perform SFT with Llama3-8B-Instruct on 50k CRESCENT data for 4 epochs. The evaluation is conducted using the standard prompt. Table 8 shows that, in both settings of 0-shot and 5-shot, the model exhibits a steady performance as the number of epochs increases.

4.7 CRESCENT for Model Distillation

Next, we explore the potential of using the CRESCENT-generated data to distil the knowledge of an

#epochs	1	2	3	4
0-shot	50.8	60.4	61.1	62.6
5-shot	74.3	75.7	75.3	75.9

Table 8: Accuracy in terms of number of epochs (%).

LLM into a weaker model. Specifically, we use 50k data generated by Llama3-8B-Instruct through CRESCENT to perform SFT on Llama2-7B-Chat, with settings inherited from Section 3.2. We compare this approach with the following distillation methods: (i) Directly using the **GSM8K training set** without external model enhancement, which contains only 7k samples; (ii) **MetaMath** (Yu et al., 2024): a method bootstraps existing math datasets by rewriting questions from multiple perspectives, generating a new dataset called MetaMathQA. For comparability, we use Llama3-8B-Instruct to generate 50k new QA pairs from GSM8K training set; (iii) **ScaleQuest** (Ding et al., 2024): a hybrid method combining multiple models, including Qwen2-Math-7B (Yang et al., 2024a), DeepSeek-Math7B-RL (Shao et al., 2024), GPT-4o, and InternLM2-7B-Reward (Cai et al., 2024), along with datasets from GSM8K and MATH. We randomly sample 50k QA pairs from their open-source dataset;[2] (iv) **MMIQC** (Liu et al., 2024): a method leverages GPT-4o to enhance existing GSM8K, MATH and MetaMathQA datasets. We similarly sample 50k QA pairs from their open-source data[3].

The results shown in Table 9 demonstrate that CRESCENT outperforms all other approaches that rely on external data or stronger models. This highlights that CRESCENT is an efficient and effective distillation approach, requiring no external datasets, let alone complex interactions with them. Furthermore, this result also suggests that excessive reliance on external data during distillation may limit the quality of the distilled data, in other words, the model inherently features the ability to produce data of higher quality than the seed dataset, but is constrained to merely modifying or enhancing the seed data; CRESCENT, in contrast, unleashes such ability to achieve self-improvement.

5 Related Work

Synthetic Data from Scratch: Recent efforts to reduce reliance on external seed data have led to the exploration of generating data from scratch for fine-tuning LLMs. UltraChat (Ding et al., 2023) shows how to generate diverse, high-quality multi-turn conversations without human queries. Magpie (Xu et al., 2024) introduces a self-synthesis method to generate large-scale alignment data by utilizing only pre-defined chat templates. GenQA (Chen et al., 2024a) aims to generate large instruction datasets with minimal human oversight by prompting LLMs to create diverse instruction examples. Note note that these methods primarily focus on *creating alignment data to train the instruction-following capabilities of base models*.

Method	Teacher data	#Data	Teacher model	Acc (5-shot)	Acc (0-shot)
-	GSM8K	7k	-	38.4	38.4
MetaMath	GSM8K	50k	Llama3-8B-I.	41.7	22.0
ScaleQuest	GSM8K&MATH	50k	Mix	38.9	22.8
MMIQC	Mix	50k	GPT-4	33.7	28.3
CRESCENT	-	50k	Llama3-8B-I.	**44.8**	**30.8**

Table 9: Comparison of distillation approaches (%).

LLM Self-Improvement: Recent methods exploring self-improvement demonstrate the potential of enhancing LLMs' capabilities through self-generated feedback. (Huang et al., 2023a) demonstrates that LLMs can improve by sampling high-confidence answers from existing high-quality question sets. Similarly, CodeRL (Le et al., 2022) introduces reinforcement learning to program synthesis, where the model receives feedback from unit tests and critic scores from other models, aiming to optimize performance on unseen coding tasks. StaR (Zelikman et al., 2022) leverages small amounts of rationale examples and iteratively refines the reasoning ability through self-generated rationales. SPIN (Chen et al., 2024b) proposes a self-play fine-tuning method, where a model generates its training data from previous iterations.

6 Conclusion

We presented CRESCENT as a simple yet effective framework – leveraging techniques of bait prompting, diversification, and consensus enhancement – for exploring the self-improvement problem of LLMs. We show that CRESCENT suffices to improve the mathematical reasoning capabilities of an LLM with zero supervision signals while preserving its general performance. Moreover, it facilitates more effective and efficient LLM knowledge distillation than existing approaches based on seed-dataset augmentation.

[2] https://huggingface.co/datasets/dyyyyyyyy/ScaleQuest-Math

[3] https://huggingface.co/datasets/Vivacem/MMIQC

Limitations

We observe the following limitations of this work:

Domain scalability. Although CRESCENT can generate a variety of domain-specific datasets, the experiments in this paper are confined to evaluating its effectiveness in improving math reasoning capabilities. Further extensions to other domains are subject to future work.

Aligned model restriction. CRESCENT is designed for aligned chat models. In this paper, we did not investigate whether the same approach can be used to generate high-quality, domain-specific data for base models without instruction tuning.

Experiment setups. Due to computational resource constraints, the model parameters used in this paper are limited to 8B. Whether CRESCENT is equally applicable to models with larger parameters remains to be verified in future work.

Acknowledgments

This work has been partially supported by the ZJNSF Major Program (No. LD24F020013), by the National Key R&D Program of China (No. 2022YFF0902600), by the Fundamental Research Funds for the Central Universities of China (No. 226-2024-00140), and by the ZJU Education Foundation's Qizhen Talent program.

References

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*.

Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Slav Petrov, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy P. Lillicrap, Angeliki Lazaridou, Orhan Firat, James Molloy, Michael Isard, Paul Ronald Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, George Tucker, Enrique Piqueras, Maxim Krikun, Iain Barr, Nikolay Savinov, Ivo Danihelka, Becca Roelofs, Anaïs White, Anders Andreassen, Tamara von Glehn, Lakshman Yagati, Mehran Kazemi, Lucas Gonzalez, Misha Khalman, Jakub Sygnowski, and et al. 2023. Gemini: A family of highly capable multimodal models. *CoRR*, abs/2312.11805.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, et al. 2024. Internlm2 technical report. *CoRR*, abs/2403.17297.

Jaime G. Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, pages 335–336. ACM.

Jiuhai Chen, Rifaa Qadri, Yuxin Wen, Neel Jain, John Kirchenbauer, Tianyi Zhou, and Tom Goldstein. 2024a. Genqa: Generating millions of instructions from a handful of prompts. *CoRR*, abs/2406.10323.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. 2024b. Self-play fine-tuning converts weak language models to strong language models. In *ICML*. OpenReview.net.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *CoRR*, abs/1803.05457.

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. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Tri Dao. 2024. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *EMNLP*, pages 3029–3051. Association for Computational Linguistics.

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. *CoRR*, abs/2410.18693.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien

Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. 2024. The Llama 3 herd of models. *CoRR*, abs/2407.21783.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. A framework for few-shot language model evaluation.

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*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *ICLR*. OpenReview.net.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023a. Large language models can self-improve. In *EMNLP*, pages 1051–1068. Association for Computational Linguistics.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Yao Fu, et al. 2023b. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. *Advances in Neural Information Processing Systems*, 36:62991–63010.

Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, et al. 2024. Gpt-4o system card. *CoRR*, abs/2410.21276.

Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Mudit Verma, Kaya Stechly, Siddhant Bhambri, Lucas Saldyt, and Anil Murthy. 2024. Position: LLMs can't plan, but can help planning in LLM-modulo frameworks. In *ICML*. OpenReview.net.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *CoRR*, abs/2001.08361.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Chu-Hong Hoi. 2022. Coderl: Mastering code generation through pretrained models and deep reinforcement learning. In *NeurIPS*.

Qintong Li, Leyang Cui, Xueliang Zhao, Lingpeng Kong, and Wei Bi. 2024. Gsm-plus: A comprehensive benchmark for evaluating the robustness of LLMs as mathematical problem solvers. In *ACL (1)*, pages 2961–2984. Association for Computational Linguistics.

Haoxiong Liu, Yifan Zhang, Yifan Luo, and Andrew Chi-Chih Yao. 2024. Augmenting math word problems via iterative question composing. *CoRR*, abs/2401.09003.

Jun S Liu and Jun S Liu. 2001. *Monte Carlo strategies in scientific computing*, volume 10. Springer.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR (Poster)*. OpenReview.net.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2023. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *CoRR*, abs/2308.08747.

Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing english math word problem solvers. In *ACL*, pages 975–984. Association for Computational Linguistics.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *KDD*, pages 3505–3506. ACM.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP/IJCNLP (1)*, pages 3980–3990. Association for Computational Linguistics.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. GPQA: A graduate-level google-proof q&a benchmark. *CoRR*, abs/2311.12022.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross J. Anderson, and Yarin Gal. 2024. AI models collapse when trained on recursively generated data. *Nat.*, 631(8022):755–759.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. LLaMA: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Ke Wang, Jiahui Zhu, Minjie Ren, Zeming Liu, Shiwei Li, Zongye Zhang, Chenkai Zhang, Xiaoyu Wu, Qiqi Zhan, Qingjie Liu, and Yunhong Wang. 2024. A survey on data synthesis and augmentation for large language models. *CoRR*, abs/2410.12896.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023a. Self-consistency improves chain of thought reasoning in language models. In *ICLR*. OpenReview.net.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023b. Self-instruct: Aligning language models with self-generated instructions. In *ACL (1)*, pages 13484–13508. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*.

Huajian Xin, Daya Guo, Zhihong Shao, Zhizhou Ren, Qihao Zhu, Bo Liu, Chong Ruan, Wenda Li, and Xiaodan Liang. 2024. Deepseek-prover: Advancing theorem proving in llms through large-scale synthetic data. *CoRR*, abs/2405.14333.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. 2024. MAGPIE: Alignment data synthesis from scratch by prompting aligned LLMs with nothing. *CoRR*, abs/2406.08464.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024a. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *CoRR*, abs/2409.12122.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. 2024b. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *ICLR*. OpenReview.net.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. 2022. Star: Bootstrapping reasoning with reasoning. In *NeurIPS*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *ACL (1)*, pages 4791–4800. Association for Computational Linguistics.

Shun Zhang, Zhenfang Chen, Yikang Shen, Mingyu Ding, Joshua B. Tenenbaum, and Chuang Gan. 2023. Planning with large language models for code generation. In *ICLR*. OpenReview.net.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *CoRR*, abs/2311.07911.