MathCritique: Enhancing LLM Reasoning via Critique Models with Test-Time and Training-Time Supervision

Zhiheng Xi[1,†], Dingwen Yang[1], Jixuan Huang[1], Jiafu Tang[1], Xin Guo[1], Guanyu Li[1], Yiwen Ding[1], Wei He[1], Boyang Hong[1], Shihan Dou[1], WenYu Zhan[1], Xiao Wang[1], Xiaowei Shi[2], Yitao Zhai[2], Rongxiang Weng[2], Jingang Wang[2], Rui Zheng[1], Tao Ji[1], Tao Gui[1,3,†], Zuxuan Wu[1], Qi Zhang[1], Xipeng Qiu[1,3], Xuanjing Huang[1] and **Yu-Gang Jiang[1]**
[1]Fudan University
[2]Meituan (China)
[3]Shanghai Innovation Institute
{zhxi22, tgui}@m.fudan.edu.cn

Abstract

Training critique models to provide useful feedback for actor models is an effective approach in scalable oversight, especially for complex tasks like math reasoning. However, current research lacks suitable datasets for effectively training critique models and integrating them in a principled way at both test time and training time. To bridge this gap, we first propose AutoMathCritique, an automated and scalable framework for collecting critique data. Using it, we create MathCritique-76k, a dataset of $76,321$ instances paired with step-level feedback, which is then used to train critique models for generating natural-language feedback. We show that critique models consistently improve the actor's performance—particularly on difficult queries at test time—with larger gains as inference compute scales. Building on the findings, we propose a critique-in-the-loop self-improvement method that incorporates critique-based supervision into the actor's self-training process. Extensive experiments demonstrate that this method improves the actor's exploration efficiency and solution diversity, especially on challenging queries, leading to a stronger actor model. Our code and datasets are at https://mathcritique.github.io/.

1 Introduction

With the rapid development of large language models (LLMs) [OpenAI, 2023; LlamaTeam, 2024], providing supervision that is both scalable and effective for them has become a key challenge (i.e., the scalable oversight problem), especially for complex tasks such as mathematical reasoning [Kenton *et al.*, 2024; Sun *et al.*, 2024]. Training critique models is a promising approach, as they can provide reliable and useful feedback to help actor models identify issues and refine their responses [Welleck *et al.*, 2023; Wang *et al.*, 2024; Yao *et al.*, 2024]. However, the open-source community currently lacks a suitable dataset for effectively training critique models and for integrating them into actor models in a principled way during both training and inference.

To bridge this gap, we first propose an automated and scalable framework **AutoMathCritique** to collect diverse and high-quality step-level critique data without substantial human annotation (Section 3). It consists of three stages: (1) flawed reasoning construction, (2) critique generation, and (3) data filtering. First, we synthesize diverse erroneous reasoning paths using controlled error injection (e.g., specifying error location or content), which also yields hints for critique. Next, annotator models label step-level correctness and provide constructive feedback given the reasoning path and optional hints. Finally, the reasoning model revises its response based on the critiques, and we apply Monte Carlo sampling [Wang *et al.*, 2024] to improve critique quality. A case is illustrated in Figure 1 of Appendix A.

Next, using AutoMathCritique, we create **MathCritique-76k**, a critique dataset containing $76,321$ samples, which is subsequently used to fine-tune a critique model. With the critique model's assistance, the actor model improves its exploration efficiency and reasoning quality at test time, leading to a significant enhancement in final performance (Section 4). Through in-depth analysis, we find that the critique models are particularly effective in difficult queries, and notably, these performance gains scale favorably with increased test-time compute [Snell *et al.*, 2024; Brown *et al.*, 2024].

Motivated by test-time insights, we further propose a **critique-in-the-loop** self-improvement method (Section 5), which integrates a critique model into the actor's exploration and learning. With critique supervision and increased exploration compute on hard queries, our approach improves exploration efficiency and solution diversity, mitigating tail narrowing [Ding *et al.*, 2024] during iterative training. We validate its effectiveness through extensive experiments and provide further analysis (Section 6).

In summary, our main contributions are:

- We propose AutoMathCritique, a scalable framework for collecting step-level critique data, and use it to build MathCritique-76k.

[†]Corresponding authors.

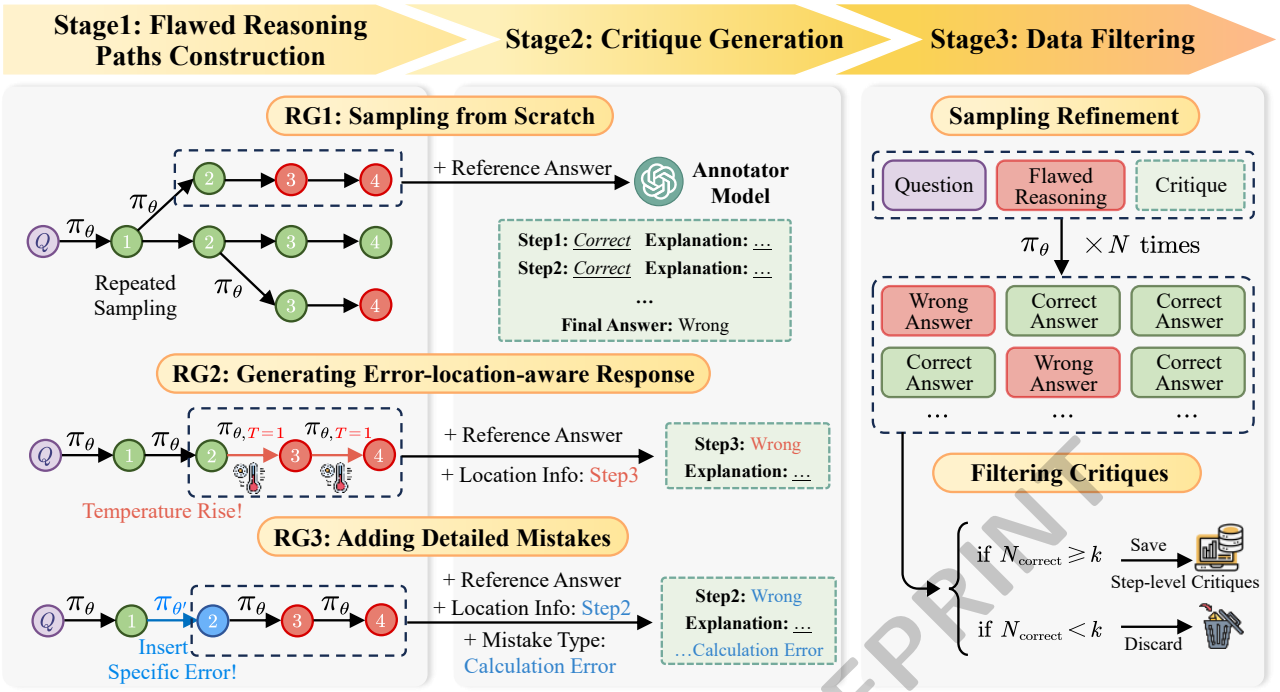

Figure 1: The overview of AutoMathCritique framework.

- We fine-tune critique models and show that their feedback improves actor reasoning at test time, especially on hard queries and with more inference compute.
- Inspired by test-time insights, we propose critique-in-the-loop self-improvement method, which integrates critique models into actor self-training.
- We conduct extensive experiments and analyses to validate our method. The results demonstrate its effectiveness in enhancing exploration efficiency and solution diversity.

2 Preliminaries

In the two-player setting studied in this paper, there are two roles: the actor model and the critique model. Also, there are three tasks [Saunders *et al.*, 2022]: reasoning, critique, and refinement.

In the reasoning task, the actor model π_θ is given a problem x and is prompted to generate a response $y = \pi_\theta(x)$.

Next, the critique model π_ϕ performs the critique task, where, given the problem and response, it generates critical feedback $c = \pi_\phi(x, y)$. Notably, if the oracle reward of the response is not given, the critique task consists of two subtasks: the discriminative task and the feedback generation task. The former determines whether the response contains flaws, while the latter generates constructive natural language feedback.

Finally, for the refinement task, given the problem, response, and critique, the actor generates a new response $y' = \pi_\theta(x, y, c)$. Alternatively, we can define direct refinement $y' = \pi_\theta(x, y)$, where the actor provides an improved answer only based on an existing answer, which is also referred to as "self-correction" [Welleck *et al.*, 2023].

Dataset	Query	Golden reasoning path	Critique
GSM8K	7,473	7,473	37,873
MATH	7,500	7,498	38,448
Total	15,973	15,971	76,321

Table 1: Statistics of MathCritique-76k.

3 AutoMathCritique: A Framework for Critique Data Collection

To train critique models that offer step-level supervision and constructive feedback for reasoning, we introduce AutoMathCritique—an automated and scalable framework for collecting critique data. As shown in Figure 1, this framework consists of three main stages: flawed reasoning path construction, critique generation, and data filtering.

Using AutoMathCritique, we create a dataset containing 76,321 samples named MathCritique-76k. The construction queries are drawn from the training sets of two widely used datasets in the field of mathmatical reasoning: GSM8K [Cobbe *et al.*, 2021] and MATH [Hendrycks *et al.*, 2021b]. The statistics are listed in Table 1, and further details are provided in Appendix C.

3.1 Construction of Flawed Reasoning Paths

To construct a dataset of flawed reasoning paths, we leverage several distinct response generation (RG) approaches that induce diverse errors (i.e., location or specific details), with Llama3-8B [LlamaTeam, 2024] serving as the primary sampling model.

RG1: sampling from scratch. The actor generates a response solely from the query. We use repeated sampling to obtain flawed responses.

RG2: generating error-location-aware response. Starting from a specific step of a correct response, we modify the model's hyperparameters (e.g., increasing temperature) to sample a flawed response. This ensures all preceding steps are exactly the same as the original correct response, while subsequent steps may contain errors.

RG3: introducing detailed mistakes. The model is explicitly prompted to implant mistakes into a correct solution. Inspired by previous work [Lanham *et al.*, 2023; Li *et al.*, 2024], our prompt enumerates common reasoning error types and provides few-shot examples. Once an error is inserted, the model continues reasoning from the erroneous step for up to 16 attempts, until it reaches a flawed answer. Finally, we balance the number of data among different error types in the dataset.

3.2 Generation of Critiques

Step-level critique generation. We enhance quality by checking each step to identify the first error in the solution. Specifically, we employ two methods to generate step-level critique data: (1) We instruct the critique annotator (GPT-4o) to directly identify the location of the first error and provide corresponding feedback. (2) We instruct the annotator model to label step by step, stopping the process once the first error is detected, at which point they provide the corresponding feedback.

Critique generation based on varying information about errors. Our construction strategy varies the hint information provided, depending on the correctness of the response and its generation method. For correct responses, no hints are given, and the critique is collected only if every step is correctly labeled by the annotator. For flawed responses, the prompts are designed accordingly: RG1 prompts include a correct reference; RG2 prompts are augmented with the likely starting point of the errors; and RG3 prompts further incorporate detailed information about the mistakes.

3.3 Data Filtering

To enhance data quality, we apply a filtering process through Monte Carlo sampling: each (query, response, critique) tuple is refined by the actor model for 10 trials, and the critique data is retained only if its accuracy exceeds a predefined threshold $\tau = 0.3$. To further assess the quality, we conduct a human evaluation on 500 randomly sampled critiques from the dataset, repeated 5 times. The rate of low-quality data is approximately 1.2%, suggesting the high quality of our dataset.

4 Critique Models Improves Reasoning through Test-Time Supervision

Using MathCritique-76k, we train critique models to provide step-level supervision and useful feedback on reasoning paths, along with the actor models with reasoning and refinement ability, with details provided in Appendix D.

Then, we investigate the impact of trained critique models in supporting the reasoning model at test-time (see top of Figure 2). Detailed setups can be found in Appendix E.

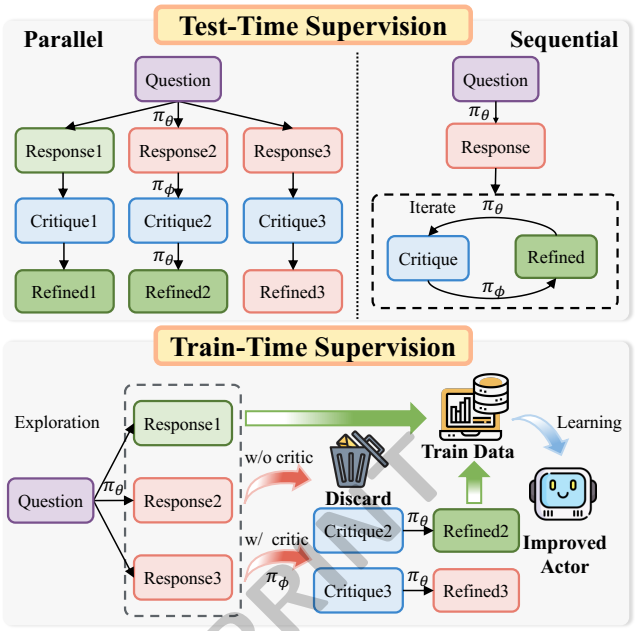

Figure 2: Illustration of how critique models provide supervision and feedback for actor models.

Critique models are highly effective at identifying the correctness of reasoning, offering constructive feedback for erroneous responses, and improving the overall accuracy of the actor. First, we compare our critique models with SOTA models used as critics. Results in Table 2 show that our 8B critique model significantly outperforms GPT-3.5, while our 70B model achieves performance on par with the GPT-4 series. Specifically, our 8B model surpasses GPT-3.5-Turbo in reasoning path judgment accuracy, and demonstrates a helpfulness improvement of 17.70% on GSM8K and 1.93% on MATH. The 70B model advances further: its discriminability exceeds GPT-4-Turbo and GPT-4o on GSM8K and closely matches them on MATH; its correction accuracy on both datasets approaches that of GPT-4 series models, ultimately guiding the actor to comparable levels of accuracy.

Critique models assist the actor in better handling challenging queries. Next, we investigate the distribution of performance gains brought by the critique model across different difficulty levels. The results are illustrated in Figure 3, where the queries are categorized into 5 difficulty levels based on the number of correct responses among 100 samples [Hendrycks *et al.*, 2021b]. It is evident that on both training and testing sets, critique models provide minimal benefit for simpler queries—where the actor already performs well—but significantly improve performance on more challenging problems. This phenomenon is more pronounced on the training set, offering valuable insights for incorporating critique supervision during training (Section 5) [Tong *et al.*, 2024].

Scaling up test-time compute consistently improves reasoning performance. Moreover, we investigate whether incorporating critique models can further elevate the performance ceiling as test-time compute scales. A widely used metric is majority voting (MV@K) [Wang *et al.*, 2023], which measures

Critique Model	GSM8K			MATH		
	Acc.	Discrimin.	Helpfulness	Acc.	Discrimin.	Helpfulness
No Critique Model	54.81	-	-	17.22	-	-
GPT-3.5-Turbo	58.38	62.9%	13.3%	25.56	51.3%	14.3%
GPT-4-Turbo	77.86	91.6%	57.5%	36.00	87.6%	26.2%
GPT-4o	79.52	91.5%	59.7%	39.98	85.4%	30.9%
Critique Model-8B	63.31	79.4%	31.0%	24.26	75.7%	16.2%
Critique Model-70B	76.88	92.3%	55.3%	33.94	82.3%	23.9%

Table 2: Test-ime evaluation results of critique models on GSM8K and MATH. "Acc." represents accuracy; "Discrimin." refers to the accuracy of determining whether a reasoning path contains errors; "Helpfulness" indicates the proportion of cases in which the actor model corrects its wrong answers with the help of critique models.

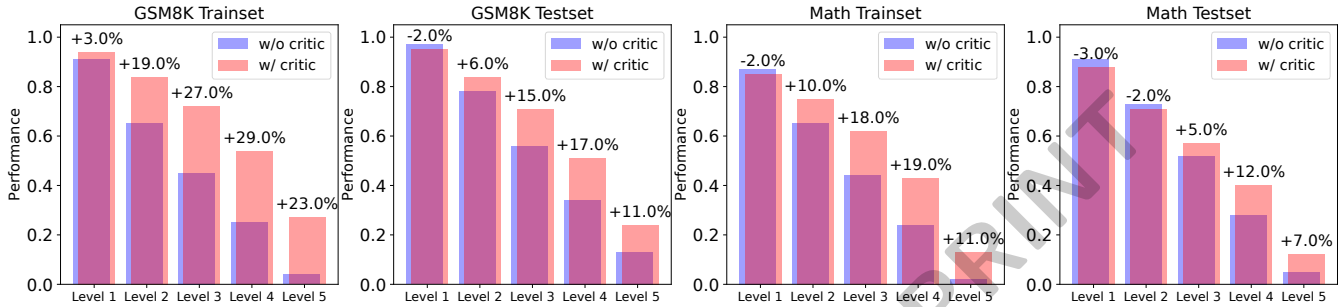

Figure 3: The impact of using critique models on performance across different difficulty levels.

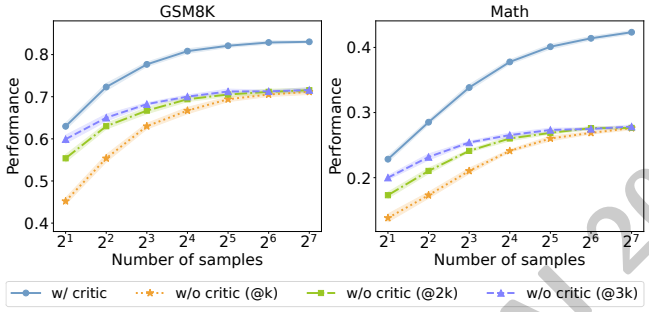

Figure 4: Majority voting (MV@K) performance when scaling test-time compute. "$2K$" and "$3K$" denote using $2\times$ and $3\times$ the sampling amount, respectively.

whether the most frequent answer among K parallel samples is correct, reflecting the model's consistency in generating high-quality responses. As shown in Figure 4, without critique models, MV@K performance improves with increased compute but quickly plateaus, even at higher levels like MV@2K and MV@3K. In contrast, with critique models, performance surpasses the baseline significantly under the same computing budget. These findings indicate that critique models effectively improve the exploration efficiency and quality of the actor, extending the performance ceiling when allocated more inference-time compute.

5 Critique-in-the-loop Self-Improvement for Better Reasoning Models

In Section 4, we observe that critique models significantly aid in raising the reasoning performance when scaling up inference compute. To further leverage their strengths, we present a critique-in-the-loop self-improvement method, which scales up exploration compute on challenging queries and leads to the development of stronger reasoning models (see Figure 2).

5.1 Critique-in-the-loop Self-improvement

Vanilla self-improvement. Self-improvement is an iterative method that leverages the actor model's correct responses to progressively enhance its performance [Huang *et al.*, 2023; Tian *et al.*, 2024]; each of the T iterations consists of two steps: exploration and learning.

In the **exploration** step of iteration t, we sample N responses for each query $x_j \in \mathcal{D}_{\text{reason}}$ from the previous model π_θ^{t-1}, i.e., $\hat{y}_j = \pi_\theta^{t-1}(x_j)$. Each data point is then filtered using the reward function $r(x, y)$, where only correct solutions are retained to form a new dataset $\mathcal{D}^t = \{(x, y)\}_{j=1}^{|\mathcal{D}^t|}$.

In the **learning** step of iteration t, the new dataset $\mathcal{D}^t$ is used to fine-tune the actor reasoning model π_θ. The training loss is as in Equation 2 and we also include the original reasoning set $\mathcal{D}_{\text{reason}}$ and refinement set $\mathcal{D}_{\text{refine}}$ [Gülçehre *et al.*, 2023].

A key challenge in self-improvement is obtaining diverse, correct answers per query during the exploration step [Huang *et al.*, 2023; Tian *et al.*, 2024]. However, models often suffer from "tail narrowing": over-sample easy queries while under-sample hard ones. This bias accumulates across iterations, resulting in a long-tail distribution that ultimately leads to performance plateau or even degradation [Ding *et al.*, 2024].

Introduce critique models for high-coverage exploration. Our method enhances vanilla self-improvement by integrating critique models π_ϕ during exploration. At iteration t, π_ϕ provides feedback on incorrect responses from actor π_θ^t, prompting π_θ^t to refine them. The resulting correct solutions are then added to the next training set. This process broadens solution coverage for harder queries, effectively mitigating the tail-narrowing problem [Ding *et al.*, 2024].

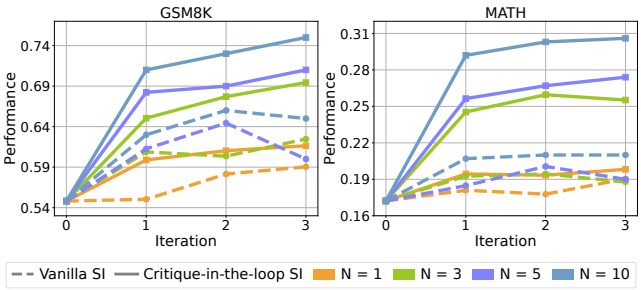

Figure 5: The evaluating results of critique-in-the-loop self-improvement. "SI" means self-improvement.

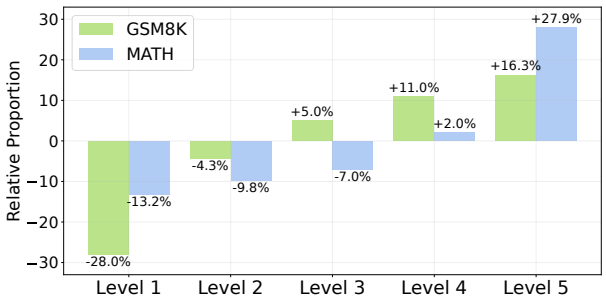

Figure 6: The difference in the proportion of training data obtained from the exploration steps of the critique-in-the-loop compared to the vanilla self-improvement.

Difficulty-aware compute allocation for exploration. Building on our findings in Section 4, we allocate more compute to harder problems. This enables additional response generation, critique, and refinement to obtain high-quality and diverse solutions. Concretely, we adopt a simple difficulty-based strategy: for incorrect initial responses, we perform L times of critique and refinement; for correct ones, we add them directly to the training set. This mitigates the long-tail issue of self-improvement, enhances sampling quality, and improves overall performance [Ding *et al.*, 2024].

We summarize the critique-in-the-loop self-improvement in Algorithm 1 of Appendix B.

5.2 Experiments

Implementation Details

To mitigate overfitting, we follow [Zelikman *et al.*, 2022] and always fine-tune the original model π_θ^0 instead of the previous model π_θ^{t-1}. After the learning step, a new dataset of better quality samples can be created once again [Xi *et al.*, 2024]. More details are discussed in Appendix F.

Empirical Results and Findings

Critique-in-the-loop self-improvement consistently improves reasoning performance. The evaluation results in Figure 5 reveal that: (1) Performance improves with increasing sampling number N during exploration, demonstrating the benefit of enhanced compute. (2) Our method consistently outperforms vanilla self-improvement, especially with larger N. (3) The vanilla method quickly suffers from the tail narrowing issue [Ding *et al.*, 2024] and reaches a bottleneck, while our method maintains steady improvement.

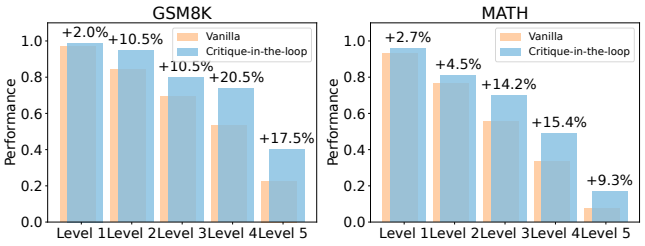

Figure 7: The performance differences between our method and vanilla self-improvement on test sets.

Critique-in-the-loop self-improvement balances the solution distribution across difficulty levels, and enhances performance on challenging queries in the test set. Figure 6 show that our method samples a higher proportion of solutions for challenging queries during exploration. This significantly balances the data distribution, mitigates the tail-narrowing issue, and consequently results in stronger test-time performance, with our approach significantly outperforms the vanilla baseline on harder problems (see Figure 7).

Combining test-time supervision with training-time supervisions yields more performance gains. We evaluate the combined use of critique models at both training- and test-time. We include two baselines: (1) self-correction, where the actor model directly refines its own reasoning (an SFT-based version of SCoRe [Kumar *et al.*, 2024]); and (2) LMSI [Huang *et al.*, 2023] as the vanilla self-improvement baseline. The training data consist of original datasets (GSM8K and MATH) and correction data derived from our refinement data by removing critiques, and reformatted into (query, original response, new response) triplets.

Results in Table 3 reveal that: (1) Integrating critique models during test-time consistently enhances performance, especially without training-time critique supervision. (2) With training-time critique, the additional benefit of test-time critique supervision becomes marginal, suggesting successful "distillation" from the critic into the actor during training. (3) Using separate critique models outperforms self-correction, consistent with prior work that highlights models' limited self-refinement capability without external feedback [Huang *et al.*, 2024; Xu *et al.*, 2024] and potential objective conflicts [Huang *et al.*, 2024]. (4) Compared to vanilla self-improvement + response-only, the approach of supervised fine-tuning + test-time critique supervision achieves better performance with less training compute (though more test-time compute), particularly on the more challenging MATH dataset.

6 Discussion and Analysis

Ablation study. Table 5 presents an ablation on two design aspects: (1) difficulty-aware compute allocation and (2) critique v.s. refinement generation compute allocation. We find both are essential: performance drops significantly without difficulty-aware allocation, highlighting its role in countering tail narrowing [Ding *et al.*, 2024]; and also drops when allocating more compute to refinement generation, showing that critique variety, not refinement depth, is critical for final gains.

Training-time	Test-time	GSM8K			MATH		
		Acc.	Pass@5	MV@5	Acc.	Pass@5	MV@5
Supervised Fine-tuning	response only	54.8	75.2	54.5	17.2	35.0	15.6
	w/ critique model	63.3	87.6	75.4	24.3	47.4	30.7
Self-Correction Fine-tuning	response only	54.2	73.1	53.4	18.1	32.4	16.6
	self-correction	60.1	81.5	67.2	24.2	41.7	26.1
Vanilla Self-Improvement	response only	64.6	83.4	70.6	20.2	38.5	23.0
	w/ critique model	70.2	90.8	78.2	27.0	48.8	31.4
Critique-in-the-loop Self-Improvement	response only	75.5	89.1	80.1	31.3	51.0	35.1
	w/ critique model	**75.8**	**91.8**	**82.8**	**31.4**	**53.1**	**36.8**

Table 3: Evaluation of training-time and test-time method combinations. "Self-Correction Fine-tuning" refers to training a model with both reasoning and correction capabilities. For test-time, "response only" represents the actor model directly generation; "w/ critique model" indicates using a critique model to provide feedback; and "self-correction" refers to the model performing correction by itself. The best performance is in **bold** and underlined, while the second-best performance is underlined.

Training-time	Test-time	AQUA-RAT	MMLU (Math)	MMLU (Physics)
Supervised Fine-tuning	response only	31.5	40.1	33.2
	w/ critique model	36.2	47.2	37.5
Self-Correction Fine-tuning	response only	29.5	42.8	32.4
	self-correction	26.8	46.0	27.7
Vanilla Self-Improvement	response only	34.3	47.5	40.0
	w/ critique model	38.6	51.5	43.9
Critique-in-the-loop Self-Improvement	response only	42.5	50.9	46.6
	w/ critique model	**45.7**	**53.7**	**49.0**

Table 4: Evaluation results of different methods on out-of-domain tasks. The best performance is in **bold** and underlined, while the second-best performance is underlined.

N	Method	GSM8K	MATH
N=3	Ours	**69.1**	**25.5**
	w/o Difficulty Aware	65.8	23.9
	More Refinement	68.8	25.5
N=5	Ours	**72.3**	**28.0**
	w/o Difficulty Aware	69.4	26.6
	More Refinement	71.2	27.7
N=10	Ours	**75.4**	31.3
	w/o Difficulty Aware	70.1	27.9
	More Refinement	73.8	**31.6**

Table 5: Ablation study of critique-in-the-loop.

Generalization of actor models and critique models. We evaluate the generalization of our method on out-of-domain reasoning tasks in university-level mathematics and physics, using three evaluation tasks: AQUA-RAT [Ling *et al.*, 2017], MMLU (Math), and MMLU (Physics) [Hendrycks *et al.*, 2021a]. Results of accuracy in Table 4 demonstrate strong out-of-domain generalization. Our actor model consistently outperforms baselines, and integrating the critique model yields additional gains across all training-time methods. Notably, these gains are slightly larger than those observed on in-domain tasks (Table 3), highlighting the critique model's critical role in generalization. Meanwhile, the self-correction method degrades performance on two datasets, further illustrating models' limitations in self-evaluation and self-refinement.

Scaling properties of critique models. Furthermore, we explore the potential of our critique models when supervising actors with different scales. Results in Figure 3 in Appendix H show that the 3B critique model successfully supervises actor models with various scales, with performance further enhanced with oracle reward functions. This indicates that smaller critique models can effectively guide larger actors, and oracle reward models provide additional supervisory signal. More details are provided in Appendix H.

How do critique models improve correct frequency and majority voting? Following [Brown *et al.*, 2024], we investigate the relationship between the correct frequency in multiple samples and the performance of majority voting, particularly under the influence of critique models. Figure 8 reveals a critical failure mode without critique models: a high-frequency incorrect answer (e.g., 45%) may outweigh a relatively frequent correct answer (e.g., 40%), leading majority voting to fail. In contrast, critique models address this through their discriminative ability and helpful feedback to suppress incorrect paths and boost the relative frequency of correct answers, thereby improving both the overall correct frequency and the performance of majority voting.

Should test-time compute be scaled sequentially or in parallel? We compare two implementations of test-time compute: **parallel sampling** of (response, critique, refinement) triplets and **sequential generation**. Figure 9 indicates that the parallel strategy achieves higher Pass@K, likely due to the greater sampling diversity of the actors. Furthermore, majority voting analysis in Figure 10 demonstrates a performance trade-off [Snell *et al.*, 2024]: parallel implementations perform better in lower compute budget, while sequential implementations scale better with higher computation. This inspires us in the development of the critique-in-the-loop self-improvement method. More details are discussed in Appendix I.

Moreover, for detailed computational cost analysis of our method, please refer to Appendix G.

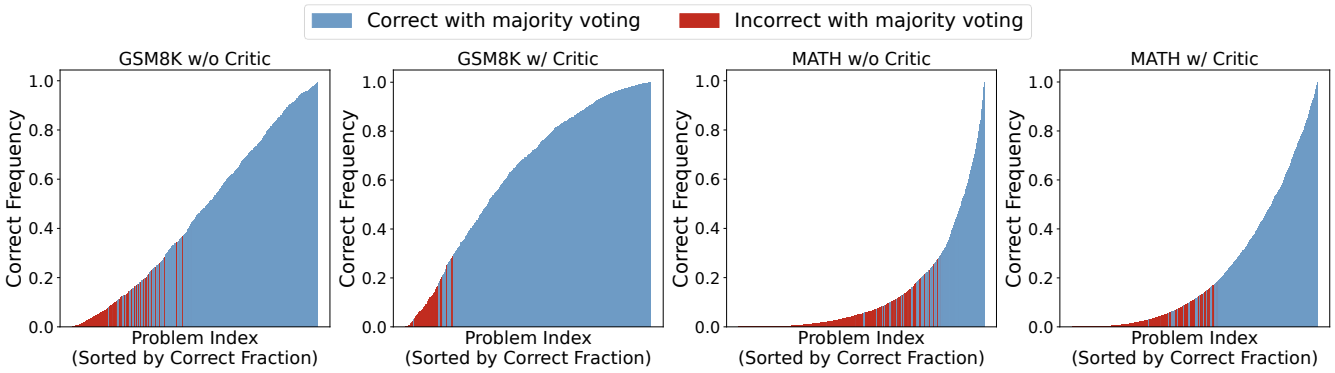

Figure 8: Fraction of correct samples (out of 1,000 per query) on test sets, with bars sorted accordingly. "w/o Critic" and "w/ Critic" refer to settings without and with a critique model, respectively.

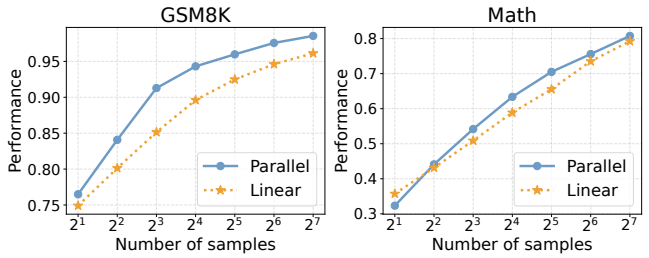

Figure 9: Pass@K performance of different strategies. Generating (response, critique, refinement) in parallel outperforms generating them sequentially.

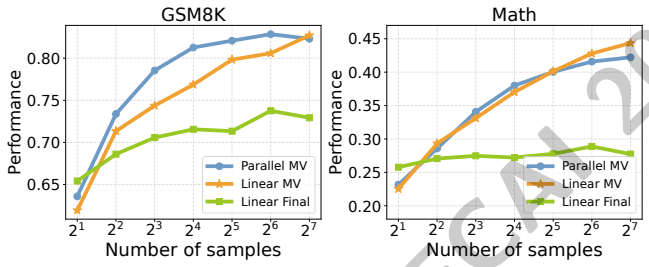

Figure 10: Evaluation results of different answer selection techniques. "Parallel MV" refers to generating K triplets in parallel and selecting the most frequent answer; "Sequential MV" refers to selecting the most frequent answer from linearly generated responses and refinements; "Sequential Final" selects the K-th response.

7 Related Work

Developing models for critique, reflection and correction. Building models that can critique, reflect, and correct is a promising route to scalable supervision across diverse domains [Welleck *et al.*, 2023; Xi *et al.*, 2023; Paul *et al.*, 2024]. Early efforts primarily rely on prompting techniques to elicit self-correction; however, these methods often underperform without oracle rewards, as models struggle to reliably indentify their own mistakes [Huang *et al.*, 2024; Xu *et al.*, 2024]. To address this, recent studies shift toward fine-tuning strategies to instill critique and correction capabilities. Yet, these methods are constrained by the need for substantial human annotation, while reinforcement learning depending on carefully engineered rewards [Kumar *et al.*, 2024; Gao *et al.*, 2024; Ankner *et al.*, 2024]. Moreover, suitable reasoning datasets for training critique models remain scarce. In this paper, we introduce AutoMathCritique, a scalable pipeline for synthesizing high-quality critique data. The trained critique models provide effective supervision that yields stable gains for the actor, both at training- and test-time.

Self-improvement for LLMs. Prior work improves LLM reasoning by training on expert-labeled step-by-step trajectories, but large-scale annotation is costly and difficult to scale [Song *et al.*, 2023]. While self-improvement methods leveraging self-generated data offer a more scalable alternative, they often suffer from tail narrowing, leading to diminishing returns [Shumailov *et al.*, 2023; Alemohammad *et al.*, 2024; Ding *et al.*, 2024]. Some works train correction models to allocate more compute toward sequential self-corrections during test-time [Ye *et al.*, 2023; Paul *et al.*, 2024]. In this paper, we fine-tune critique models to provide reliable step-level supervision and actionable feedback during both training and inference. This improves sampling efficiency and quality, ultimately strengthening the actor's reasoning performance.

8 Conclusion

In this work, we study scalable oversight through training critique models. We first propose AutoMathCritique, an automated and scalable framework for constructing critique data, and use it to build the MathCritique-76k dataset. We then train critique models capable of providing step-level supervision and effective feedback. By incorporating critique-based supervision at both test time and training time, we show that critique models can significantly improve actor model performance. Finally, we conduct extensive experiments and analyses, and hope our findings offer valuable insights to the research community.

Acknowledgements

The authors wish to thank all anonymous reviewers. This work was partially funded by National Natural Science Foundation of China (No.62576106, 62476061, 62376061).

Contribution Statement

Zhiheng Xi and Dingwen Yang share co-first authorship.

References

[Alemohammad *et al.*, 2024] Sina Alemohammad, Josue Casco-Rodriguez, Lorenzo Luzi, Ahmed Imtiaz Humayun, Hossein Babaei, Daniel LeJeune, Ali Siahkoohi, and Richard G. Baraniuk. Self-consuming generative models go MAD. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[Ankner *et al.*, 2024] Zachary Ankner, Mansheej Paul, Brandon Cui, Jonathan D. Chang, and Prithviraj Ammanabrolu. Critique-out-loud reward models. *CoRR*, abs/2408.11791, 2024.

[Brown *et al.*, 2024] Bradley C. A. Brown, Jordan Juravsky, Ryan Saul Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *CoRR*, abs/2407.21787, 2024.

[Cobbe *et al.*, 2021] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021.

[Ding *et al.*, 2024] Yiwen Ding, Zhiheng Xi, Wei He, Zhuoyuan Li, Yitao Zhai, Xiaowei Shi, Xunliang Cai, Tao Gui, Qi Zhang, and Xuanjing Huang. Mitigating tail narrowing in llm self-improvement via socratic-guided sampling. *arXiv preprint arXiv:2411.00750*, 2024.

[Gao *et al.*, 2024] Bofei Gao, Zefan Cai, Runxin Xu, Peiyi Wang, Ce Zheng, Runji Lin, Keming Lu, Junyang Lin, Chang Zhou, Wen Xiao, Junjie Hu, Tianyu Liu, and Baobao Chang. LLM critics help catch bugs in mathematics: Towards a better mathematical verifier with natural language feedback. *CoRR*, abs/2406.14024, 2024.

[Gülçehre *et al.*, 2023] Çaglar Gülçehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. Reinforced self-training (rest) for language modeling. *CoRR*, abs/2308.08998, 2023.

[Hendrycks *et al.*, 2021a] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[Hendrycks *et al.*, 2021b] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In Joaquin Vanschoren and Sai-Kit Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021.

[Huang *et al.*, 2023] Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 1051–1068. Association for Computational Linguistics, 2023.

[Huang *et al.*, 2024] Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[Kenton *et al.*, 2024] Zachary Kenton, Noah Y. Siegel, János Kramár, Jonah Brown-Cohen, Samuel Albanie, Jannis Bulian, Rishabh Agarwal, David Lindner, Yunhao Tang, Noah D. Goodman, and Rohin Shah. On scalable oversight with weak llms judging strong llms. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[Kumar *et al.*, 2024] Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D. Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M. Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal M. P. Behbahani, and Aleksandra Faust. Training language models to self-correct via reinforcement learning. *CoRR*, abs/2409.12917, 2024.

[Lanham *et al.*, 2023] Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamile Lukosiute, Karina Nguyen, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Robin Larson, Sam McCandlish, Sandipan Kundu, Saurav Kadavath, Shannon Yang, Thomas Henighan, Timothy Maxwell, Timothy Telleen-Lawton, Tristan Hume, Zac Hatfield-Dodds, Jared Kaplan, Jan Brauner, Samuel R. Bowman, and Ethan Perez. Measuring faithfulness in chain-of-thought reasoning. *CoRR*, abs/2307.13702, 2023.

[Li *et al.*, 2024] Xiaoyuan Li, Wenjie Wang, Moxin Li, Junrong Guo, Yang Zhang, and Fuli Feng. Evaluating mathematical reasoning of large language models: A focus on error identification and correction. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 11316–11360. Association for Computational Linguistics, 2024.

[Ling *et al.*, 2017] Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *ACL*, 2017.

[LlamaTeam, 2024] AI@Meta LlamaTeam. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024.

[OpenAI, 2023] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023.

[Paul *et al.*, 2024] Debjit Paul, Mete Ismayilzada, Maxime Peyrard, Beatriz Borges, Antoine Bosselut, Robert West, and Boi Faltings. REFINER: reasoning feedback on intermediate representations. In Yvette Graham and Matthew Purver, editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2024 - Volume 1: Long Papers, St. Julian's, Malta, March 17-22, 2024*, pages 1100–1126. Association for Computational Linguistics, 2024.

[Saunders *et al.*, 2022] William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. Self-critiquing models for assisting human evaluators. *CoRR*, abs/2206.05802, 2022.

[Shumailov *et al.*, 2023] Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, and Ross J. Anderson. The curse of recursion: Training on generated data makes models forget. *CoRR*, abs/2305.17493, 2023.

[Snell *et al.*, 2024] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *CoRR*, abs/2408.03314, 2024.

[Song *et al.*, 2023] Yisheng Song, Ting Wang, Puyu Cai, Subrota K. Mondal, and Jyoti Prakash Sahoo. A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities. *ACM Comput. Surv.*, 55(13s):271:1–271:40, 2023.

[Sun *et al.*, 2024] Zhiqing Sun, Longhui Yu, Yikang Shen, Weiyang Liu, Yiming Yang, Sean Welleck, and Chuang Gan. Easy-to-hard generalization: Scalable alignment beyond human supervision. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[Tian *et al.*, 2024] Ye Tian, Baolin Peng, Linfeng Song, Lifeng Jin, Dian Yu, Haitao Mi, and Dong Yu. Toward self-improvement of llms via imagination, searching, and criticizing. *arXiv preprint arXiv:2404.12253*, 2024.

[Tong *et al.*, 2024] Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. Dart-math: Difficulty-aware rejection tuning for mathematical problem-solving. *CoRR*, abs/2407.13690, 2024.

[Wang *et al.*, 2023] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[Wang *et al.*, 2024] Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 9426–9439. Association for Computational Linguistics, 2024.

[Welleck *et al.*, 2023] Sean Welleck, Ximing Lu, Peter West, Faeze Brahman, Tianxiao Shen, Daniel Khashabi, and Yejin Choi. Generating sequences by learning to self-correct. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[Xi *et al.*, 2023] Zhiheng Xi, Senjie Jin, Yuhao Zhou, Rui Zheng, Songyang Gao, Jia Liu, Tao Gui, Qi Zhang, and Xuanjing Huang. Self-polish: Enhance reasoning in large language models via problem refinement. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 11383–11406. Association for Computational Linguistics, 2023.

[Xi *et al.*, 2024] Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Dingwen Yang, Chenyang Liao, Xin Guo, Wei He, Songyang Gao, Lu Chen, Rui Zheng, Yicheng Zou, Tao Gui, Qi Zhang, Xipeng Qiu, Xuanjing Huang, Zuxuan Wu, and Yu-Gang Jiang. Agentgym: Evolving large language model-based agents across diverse environments. *CoRR*, abs/2406.04151, 2024.

[Xu *et al.*, 2024] Wenda Xu, Guanglei Zhu, Xuandong Zhao, Liangming Pan, Lei Li, and William Wang. Pride and prejudice: LLM amplifies self-bias in self-refinement. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 15474–15492. Association for Computational Linguistics, 2024.

[Yao *et al.*, 2024] Weiran Yao, Shelby Heinecke, Juan Carlos Niebles, Zhiwei Liu, Yihao Feng, Le Xue, Rithesh R. N., Zeyuan Chen, Jianguo Zhang, Devansh Arpit, Ran Xu, Phil Mui, Huan Wang, Caiming Xiong, and Silvio Savarese. Retroformer: Retrospective large language agents with policy gradient optimization. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[Ye *et al.*, 2023] Seonghyeon Ye, Yongrae Jo, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, and Minjoon Seo. Selfee: Iterative self-revising llm empowered by self-feedback generation. Blog post, May 2023.

[Zelikman *et al.*, 2022] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. Star: Bootstrapping reasoning with reasoning. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.