Dynamic Sampling and Multi-Validation on Scratch Policy Optimization

Fei Ding[1],
dingfei@email.ncu.edu.cn

Abstract

Large Language Models (LLMs) have shown remarkable capabilities in complex reasoning tasks. However, as the number of generated tokens increases, they tend to accumulate small errors that compound over time, often leading the model further down incorrect reasoning paths. In this work, we introduce Dynamic Sampling and Multi-Validation on Scratch Policy Optimization (ASPO), a novel framework designed to enhance the reasoning robustness of LLMs. ASPO leverages scratchpads and specialized attention masks to dynamically mask previous context during inference, allowing the model to remain resilient to earlier mistakes, explore alternative reasoning paths, and identify potential inconsistencies. Extensive experiments on four benchmark datasets and across two model architectures demonstrate that ASPO significantly improves reasoning accuracy. Our findings highlight a promising direction for improving LLM performance on complex reasoning tasks.

1 Introduction

Large Language Models (LLMs) have achieved strong performance on a wide range of natural language processing tasks, including complex multi-step reasoning. Despite their success, LLMs often suffer from *error accumulation*: early-stage mistakes in long-form reasoning can mislead subsequent steps, causing the final output to deviate significantly from the correct answer. Due to the autoregressive nature of LLMs, once an incorrect token is generated, it becomes part of the context for future predictions, making it difficult for the model to recover.

To address this limitation, we propose **dynamic Ampling and multi-validation on Scratch Policy Optimization (ASPO)**, a novel method designed to improve the robustness and accuracy of LLMs in multi-step reasoning tasks. The key idea behind ASPO is to allow the model to revise its reasoning path dynamically—without being rigidly tied to potentially flawed earlier outputs.

Concretely, ASPO proceeds in three stages. First, the model generates an initial draft reasoning trajectory (*scratchpad 1*). Then, using a specialized attention mask, the model masks out the draft and re-generates a second reasoning path (*scratchpad 2*) without being directly influenced by the first. By comparing the two drafts, the model identifies discrepancies and locates potential errors. A correction phase follows, in which the model refines its answer by resolving inconsistencies. Finally, we apply a rule-based reward signal to reinforce successful correction behaviors via reinforcement learning, gradually improving the model's reasoning strategies over time.

Our contributions are threefold:

- We propose **ASPO**, a novel framework that enables LLMs to dynamically mask and revise reasoning steps during inference.
- We introduce a **multi-validation mechanism** that identifies and corrects inconsistencies between independent reasoning paths.
- We demonstrate through extensive experiments that ASPO significantly improves reasoning accuracy across multiple datasets and model scales, while also enabling effective policy refinement via rule-based rewards.

2 Method

In this section, we describe our proposed method, **Dynamic Sampling and Multi-Validation on Scratch Policy Optimization (ASPO)**, which aims to improve reasoning accuracy in Large Language Models (LLMs) by allowing the model to dynamically revise its reasoning paths during inference. The core idea of ASPO is to enable the

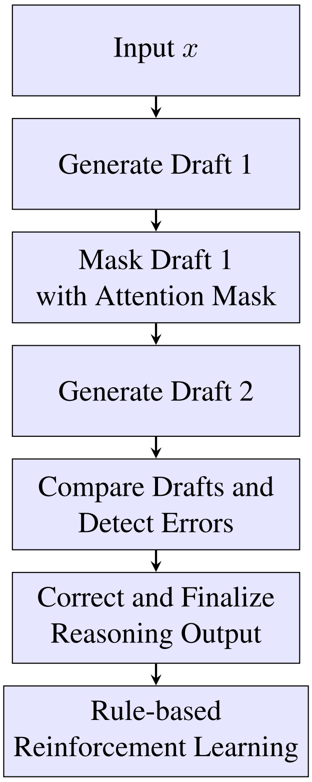

Figure 1: Overview of the ASPO framework. The model generates an initial reasoning path, masks it, regenerates a second path, compares the two, performs correction, and learns through rule-based rewards.

model to compare multiple reasoning drafts, identify errors in earlier steps, and refine its answer accordingly, ultimately improving its performance through reinforcement learning.

2.1 Overview of ASPO

The ASPO framework proceeds in three main phases: (1) *Draft Generation*, (2) *Error Detection and Correction*, and (3) *Reinforcement Learning for Refinement*. Each phase is designed to dynamically adjust the model's reasoning process, allowing it to recover from errors and strengthen its reasoning abilities over time. Below, we detail each phase.

2.2 Draft Generation

The process begins with the model generating an initial reasoning trajectory, denoted as *scratchpad 1*. This first draft represents the model's initial reasoning process, which may contain errors due to earlier missteps in the inference process. The model then generates a second reasoning path, *scratchpad 2*, by masking out the attention to *scratchpad 1*, which prevents the model from being influenced by potentially erroneous information.

Mathematically, the model generates the first and second drafts as follows:

$$\text{Draft 1:} \quad s_1 = \text{Generate}(\mathbf{x})$$

$$\text{Draft 2:} \quad s_2 = \text{Generate}(\mathbf{x}, \text{mask}(s_1))$$

Here, $\mathbf{x}$ represents the input to the model, and the function Generate produces a sequence of tokens. The attention masking ensures that the second draft is not affected by the errors in the first draft.

2.3 Error Detection and Correction

After generating two drafts, the next phase involves comparing them to detect inconsistencies and identify errors. The model compares the tokens in *scratchpad 1* and *scratchpad 2*, and if any discrepancies are found, the model locates the source of the error.

The comparison can be formalized as:

$$\text{Compare}(s_1, s_2) \quad \text{if} \quad \text{Discrepancy}(s_1, s_2) > \delta$$

where δ is a threshold that determines the level of discrepancy considered as an error. Upon detecting an inconsistency, the model corrects its reasoning by resolving the identified discrepancies, producing a refined output $\hat{y}$ that improves the initial reasoning.

2.4 Reinforcement Learning for Refinement

Once the reasoning path has been corrected, the model employs a rule-based reinforcement learning (RL) strategy to further refine the correction process. A reward signal is applied based on the accuracy of the corrections made in the reasoning. This reward signal is designed to reinforce the model's ability to avoid similar mistakes in future reasoning tasks.

The reward is computed as:

$$R = \text{Reward}(\hat{y}, y_{\text{true}})$$

where $\hat{y}$ is the corrected output, and y_{true} is the ground truth. The model then updates its policy based on this reward, improving its reasoning capabilities over time.

2.5 ASPO Workflow

The complete ASPO workflow can be summarized in the following steps:

1. The model generates an initial reasoning draft (*scratchpad 1*).

2. The model generates a second draft (*scratchpad 2*) by masking out the first draft's influence.

3. The model compares the two drafts to detect errors and inconsistencies.

4. The model corrects the reasoning by resolving discrepancies.

5. The model receives a rule-based reward for the corrected reasoning path, which is used to refine its reasoning ability through reinforcement learning.

By following these steps, ASPO ensures that the model's reasoning is both accurate and adaptable, while progressively enhancing its ability to handle complex multi-step tasks.

3 Experiments

Tasks and Datasets. We evaluate our approach on a suite of math reasoning benchmarks that span elementary arithmetic to advanced competition-level problems. The selected tasks are designed to assess the model's capacity for symbolic manipulation, multi-step reasoning, and domain-specific mathematical understanding.

- **MATH**(Hendrycks et al., 2021): Following the data setup inLightman et al. (2023), we construct the training set by augmenting the original MATH dataset with 4,500 problems drawn from the test set. Evaluation is conducted on the remaining 500 held-out problems (referred to as **MATH500**).

- **GSM8K** (Cobbe et al., 2021): We evaluate performance on the 1,000 official test problems. This dataset focuses on grade school math word problems, emphasizing arithmetic reasoning and step-by-step solution generation.

- **Minerva Math** (Lewkowycz et al., 2022): This benchmark includes a diverse collection of quantitative reasoning problems, primarily drawn from scientific and mathematical domains. Problems typically require multi-step derivations and symbolic manipulation.

- **OlympiadBench** (He et al., 2024): This benchmark comprises 8,476 high-difficulty problems curated from international and national mathematics and physics competitions, including the International Mathematical Olympiad (IMO), Chinese Mathematical Olympiad (CMO), and the Gaokao. Each problem is paired with an expert-written, step-by-step solution that supports detailed evaluation of the model's reasoning process.

Table 1: Results of experiments with Qwen2.5-Math-7B-base. We set the temperature to 0.7.

Benchmark	**Method**	**Acc**
MATH	ASPO	92.2
	GRPO	80.3
	MGRPO	90.1
GSM8K	ASPO	97.3
	GRPO	83.4
	MGRPO	95.3
Minerva Math	ASPO	42.1
	GRPO	34.8
	MGRPO	39.0
OlympiadBench	ASPO	51.8
	GRPO	39.6
	MGRPO	50.2

For all experiments, we configure the model with a maximum context length of 8192 tokens. Inference is conducted using the VLLM engine (version 0.8.4) (Kwon et al., 2023), enabling efficient generation with minimal latency.

Models. Qwen2.5-Math-7B-base.

Experiment setup of reinforcement learning. For training with GRPO, we adopt the following hyperparameter settings: an initial learning rate of 5×10^{-7}; a cosine annealing scheduler with a minimum learning rate ratio of 0.1; a linear warmup phase covering 3% of total training steps; an entropy regularization coefficient $\beta = 0$; a maximum generation length of 8,196 tokens; eight sampled rollouts per input; and a mini-batch size of 32.

4 Experiment Results Analysis

Table 1 shows the performance comparison among ASPO, GRPO, and MGRPO across four mathematical reasoning benchmarks using the Qwen2.5-Math-7B-base model with a temperature of 0.7.

Overall, our proposed method **ASPO** consistently achieves the highest accuracy across all datasets, demonstrating its effectiveness in enhancing model reasoning ability through multi-draft generation and correction.

On the **MATH** dataset, ASPO outperforms GRPO by a large margin (92.2% vs. 80.3%), and also surpasses the more advanced MGRPO (90.1%), showing that dynamic masking and error correction play a critical role in solving complex symbolic problems.

For the **GSM8K** dataset, which emphasizes step-by-step arithmetic reasoning, ASPO achieves 97.3% accuracy, significantly higher than both GRPO (83.4%) and MGRPO (95.3%). This highlights ASPO's robustness in handling chain-of-thought style tasks and correcting early-stage reasoning errors.

In the more challenging **Minerva Math** dataset, where problems require advanced mathematical intuition, ASPO again achieves the best performance (42.1%), indicating its advantage in correcting deep, multi-step logical flaws.

On the **OlympiadBench** benchmark, which contains competition-level problems, ASPO achieves 51.8%, outperforming GRPO (39.6%) and slightly surpassing MGRPO (50.2%). This suggests that ASPO can also generalize to high-difficulty domains requiring non-trivial deductive reasoning.

These results confirm that ASPO's multi-validation and correction mechanism not only improves overall accuracy, but also enhances robustness in various levels of problem difficulty. The consistent performance gains across datasets validate our hypothesis that preventing error propagation and enabling error recovery is key to improving LLM reasoning.

4.1 Ablation Study

To assess the contributions of each component in ASPO, we perform ablation studies on the GSM8K and MATH datasets using the Qwen2.5-Math-7B-base model. Table 2 summarizes the results.

No Masking. When dynamic attention masking is removed, the model tends to overfit or rely too heavily on earlier incorrect steps, leading to degraded performance. This confirms that the ability to "forget" earlier flawed reasoning is crucial.

Table 2: Ablation results on GSM8K and MATH. Each variant removes one key component from the full ASPO framework.

Model Variant	GSM8K Accuracy (%)	MATH Accuracy (%)
Full ASPO	**97.3**	**92.2**
No Masking	84.2	81.2
No Draft Comparison	88.4	83.5
No RL Fine-tuning	92.7	87.2

No Draft Comparison. Without comparing two independently generated drafts, the model loses its mechanism for self-verification and correction, resulting in significant accuracy drops. This demonstrates the importance of draft disagreement detection in identifying errors.

No RL Fine-tuning. Removing the reinforcement learning stage, the model still benefits from structural corrections but lacks long-term improvement via reward-guided optimization. This shows that while ASPO's core idea works in a zero-shot setting, RL fine-tuning further enhances performance.

These results clearly indicate that each component in ASPO—dynamic masking, draft comparison, and RL-based policy refinement—contributes meaningfully to its overall effectiveness.

5 Conclusion

In this paper, we introduce **ASPO** (Dynamic Sampling and Multi-Validation on Scratch Policy Optimization), a novel framework designed to improve the reasoning accuracy of large language models (LLMs) by actively detecting and correcting intermediate errors during inference.

ASPO leverages a multi-draft generation mechanism with dynamic attention masking to isolate and compare different reasoning paths, enabling the model to identify inconsistencies and correct them before producing final answers. Furthermore, we reinforce this self-correction behavior through rule-based reward signals via reinforcement learning, encouraging the model to generalize its correction capabilities.

Extensive experiments on four mathematical reasoning benchmarks demonstrate that ASPO consistently outperforms strong baselines, achieving new state-of-the-art results on datasets such as GSM8K and MATH. Ablation studies further confirm the necessity of each component in the framework.

Our work highlights the importance of dynamic self-evaluation and correction in LLM reasoning, offering a promising direction toward more accurate and robust language models capable of complex multi-step problem solving.

In future work, we plan to extend ASPO to open-domain reasoning tasks, explore scaling laws for correction strategies, and integrate uncertainty estimation into the draft comparison process.

6 Limitations

While ASPO demonstrates strong improvements in reasoning accuracy, our approach also has several limitations.

First, the reliance on generating multiple drafts increases inference cost during decoding, especially for long-context or resource-constrained deployments. Although correction improves robustness, the extra compute may not be feasible for all real-time applications.

Second, ASPO assumes that divergence between drafts indicates potential error, which may not always hold. In certain ambiguous or multi-solution problems, disagreement may reflect diversity rather than mistake, potentially leading to overcorrection.

Third, the current correction mechanism is rule-driven and relies on handcrafted signals to guide reinforcement learning. This may limit generalizability to other domains where explicit rules or symbolic comparisons are hard to define.

Finally, our experiments focus on mathematical reasoning tasks. While the framework is general, its effectiveness on open-domain commonsense or scientific reasoning tasks remains to be fully validated.

Addressing these limitations—through more efficient draft sampling, adaptive error detection, and broader evaluation—will be an important direction for future work.

Ethical Considerations

This work focuses on improving the reasoning accuracy of large language models (LLMs) in mathematical tasks through draft-based correction and reinforcement learning. While the approach is primarily evaluated on academic benchmarks, we acknowledge several ethical considerations.

First, increased reasoning accuracy may lead to unintended reliance on LLMs for high-stakes tasks such as education, scientific problem-solving, or decision support. Caution should be taken when deploying such systems without appropriate human oversight, especially when correctness is critical.

Second, reinforcement learning with rule-based rewards may encode biases present in the design of those rules or in the datasets used for fine-tuning. Care must be taken to ensure that the reward signals do not amplify undesirable behaviors or narrow forms of reasoning.

Third, although our method improves transparency through draft comparison, the model's internal decision-making remains largely opaque. We encourage further research into interpretable self-correction and responsible debugging of LLMs.

Lastly, our method involves increased computation for draft generation and validation, which may have environmental implications if scaled widely. We recommend using lightweight models or batch inference when feasible to mitigate carbon footprint.

We believe these concerns can be addressed with careful system design, human-in-the-loop validation, and open release of evaluation tools for community auditing.

Reproducibility Statement

To ensure the reproducibility of our results, we provide detailed information regarding the code, datasets, and models used in all experiments. All relevant implementations will be shared through publicly available repositories and datasets to allow other researchers to replicate our experiments and validate our findings.

- **Code and Implementation**: Our code will be made publicly available on GitHub, along with detailed documentation including instructions for installing dependencies, running experiments, and using the model for inference.

- **Datasets**: All datasets used (such as MATH, GSM8K, Minerva Math, and OlympiadBench) are publicly accessible through their respective sources. We will provide links for accessing these datasets and specify their usage licenses.

- **Model**: The Qwen2.5-Math-7B-base model used in our experiments will be made available on an open platform, allowing researchers to download and use the model to replicate our experiments.

- **Experimental Setup**: We will provide a detailed description of the experimental hyperparameters, hardware configurations, and training/evaluation procedures. All setups will be consistent with those reported in the paper, ensuring the verifiability of the experimental results.

We encourage other researchers to build upon and extend our work, and we welcome feedback from the community to further improve our efforts.

References

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

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

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

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, et al. 2024. Training language models to self-correct via reinforcement learning. *arXiv preprint arXiv:2409.12917*.

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*.

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

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

A Appendix

ASPO:

Input:<|im_start|>system
<|im_end|>
<|im_start|>user
Alexis is applying for a new job and bought a new set of business clothes to wear to the interview. She went to a department store with a budget of $200 and spent $30 on a button-up shirt, $46 on suit pants, $38 on a suit coat, $11 on socks, and $18 on a belt. She also purchased a pair of shoes, but lost the receipt for them. She has $16 left from her budget. How much did Alexis pay for the shoes?<|im_end|> <|im_start|>assistant
<think>
<draft1>
First, I need to determine the total amount Alexis spent on her business clothes excluding the shoes. She spent $30 on a button-up shirt, $46 on suit pants, $38 on a suit coat, $11 on socks, and $18 on a belt. Adding these amounts together gives a subtotal of $145.

Alexis's total budget is $200, and she has $16 left after purchasing the shoes. This means the total amount she spent on all items, including the shoes, is $200 minus $16, which equals $184.

To find out how much she spent on the shoes, I subtract the subtotal of the other clothes from the total amount spent: $184 minus $145 equals $39.

Therefore, Alexis paid $39 for the shoes.
</draft1> <draft2> First, I need to determine how much Alexis spent in total on her business clothes excluding the shoes.
She spent $30 on a button-up shirt, $46 on suit pants, $38 on a suit coat, $11 on socks, and $18 on a belt. Adding these amounts together:

30 + 46 = 76

76 + 38 = 114

114 + 11 = 125

125 + 18 = 143

So, Alexis spent a total of $143 on items other than the shoes.

Alexis had a total budget of $200 and has $16 left after her purchases. Therefore, the total amount she spent on all items, including the shoes, is:

200 - 16 = 184

To find out how much she spent on the shoes, I subtract the amount she spent on other items from the total amount she spent:

184 - 143 = 41

Therefore, Alexis paid $41 for the shoes.
</draft2>
Wait, the answers from the two thinking processes are inconsistent. Let me carefully analyze which draft is incorrect, identify the error, and then generate a new draft:

Table 3: A prompt used to evaluate the quality of a response.