Group-Reward Policy Optimization for Mathematical Reasoning

Minh Binh Le
University of Massachusetts Amherst
Amherst, MA
minble@umass.edu

October 2025

Abstract

Group-Reward Policy Optimization (GRPO) is a recent and efficient reinforcement learning algorithm for large language models alignment through relative reward comparison. In my project design, I explore the use of GRPO to enhance mathematical reasoning in decoder-only the base Qwen3-1.7B models. The training objective is to improve reasoning accuracy beyond 37% zero-shot baseline of the 1024 max tokens base model by applying GRPO and a GRPO $\rightarrow$ SFT $\rightarrow$ GRPO training method.

Keywords: GRPO · Reinforcement Fine-Tuning · Reasoning LLMs · Alignment

1 Introduction

Large language models, especially on-device language models, demonstrate strong performance on simple reasoning but are limited in long-chain reasoning. While supervised fine-tuning can improve some capabilities, the model does not generalize very well without sufficient high-quality data from SFT. Reinforcement learning methods like Proximal Policy Optimization (PPO) are inefficient as they require additional critic models to rank the rewards.
GRPO's method makes it more efficient by comparing multiple rollouts within a batch to obtain better reasoning patterns. This is ideal for mathematical tasks where models can explore many different paths creatively that lead to the same result.

2 Background

2.1 Reinforcement Learning for LLM Alignment

Original RL methods on large language models like PPO need a critic model to reward responses based on the current fine-tuning models. This can be computationally expensive when you need to train an additional model to give the rewards and have two models running in parallel. GRPO, instead, uses token-level normalized advantages within rollout groups, enabling faster reward feedback without computational overheads.

2.2 Group-Reward Policy Optimization

$$\mathcal{L}_{\text{GRPO}}(\theta) = E_G \left[\frac{1}{n} \sum_{i=1}^{n} \min(r_i(\theta) A_i,\ \text{clip}(r_i(\theta), 1-\epsilon, 1+\epsilon)\, A_i) \right], \quad A_i = \frac{r_i - \mu_G}{\sigma_G + \delta}.$$

This formula explains that this policy is to get a smaller objective and make the model more stable while continuously reupdating the policy.
Instead of using other models to act as a reward function, it'll use the insight within the local reward function and rank it versus other batches to get a better response overall.

2.3 Experiments

I chose the dataset Countdown-Tasks-3to4 because it's easy to define a good reward function and yields a diverse range of calculations the models need to compute and try out before answering, making it perfect for the experiment.

3 Implementation

3.1 Foundational Model

I used the open-source Qwen3-1.7B architecture in both 256 and 512 maximum token limits
Both were trained using 8-bit quantization on a single NVIDIA A100 GPU.

3.2 Reward Function

Rewards were based on output correctness and format quality:

Condition	**Reward**
Correct numeric result	1.0
Near-correct	0.8–0.3
Include ¡answer¿ but wrong	0.1
Wrong format	0.0

Rewards were group-normalized before computing the policy loss. The per-token PPO loss used a clipping range of 0.25 to stabilize gradients.

3.3 Training

Two experimental setups were tested:

1. **Multi-stage GRPO $\rightarrow$ SFT $\rightarrow$ GRPO (256 max)**: The first GRPO phase taught reasoning structure and tag compliance. The model self-improves by realizing it should cut down on random token generation and focus more on the math, allowing it to go from 0.2% to 37.3% accuracy in the first 80 RL steps.
   SFT phase reinforced correct traces using curated examples. This allows the model to consolidate its previously explored correct reasoning trace and make training more stable
   Final GRPO phase consolidated structured reasoning and made it more stable, allowing for further improvement by a further 10%, reaching the final accuracy of 48.8%

2. **Single-stage GRPO (512 max**: Longer context allowed extended reasoning but no supervised stabilization. It follows the same pattern as 256 max tokens generation. However, because it's still exploring at the temperature of 1.0, the model generates some Laos and

China reasoning tokens, making it less stable and achieving an improvement of up to 39.2% from the base of around 14%.

3.4 Computing Resources

Training ran for 160 steps for 256 max tokens with double GRPO and SFT, and only 80 steps for 512 max tokens took 40 minutes - 1.5 hours of compute on a single Nvidia A100.

4 Results

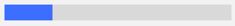
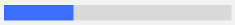
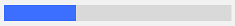
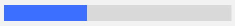
4.1 Performance Trends

The L_{256} model improved from 0% to 48.8% accuracy after the full training. The L_{512} single-stage model reached 39.2% after just one GRPO. This proves that the learning capabilities of a smaller model could be improved by a better-designed and more comprehensive training method.

Table 1: Accuracy Over Training Steps for L_{256} (GRPO $\rightarrow$ SFT $\rightarrow$ GRPO)

Step	**Accuracy (%)**	**Table Chart**
0	0.0	
10	0.1	
20	0.2	
30	0.2	
40	3.3	
50	9.2	
60	13.5	
70	21.5	
80	37.3	
Start of next GRPO phase after SFT		
80	31.6	
90	44.0	
100	44.5	
110	44.5	
120	45.5	
130	44.0	
140	44.6	
150	45.8	
160	45.3	
170	47.1	
180	48.8	

Table 2: Accuracy Over Training Steps for L_{512} (GRPO only)

Step	**Accuracy (%)**	**Table Chart**
0	11.4	
10	21.1	
20	28.1	
30	30.5	
40	30.2	
50	31.6	
60	36.1	
70	36.4	
80	39.2	

4.2 Qualitative Behavior

The L_{256} model learned to produce structured expressions like `(79 - 60) + 17 = 36 <answer>36</answer>` very accurately. Also, it realized how trying to get the correct answer is the most important goal, not generating more tokens. So instead of outputing a full thinking token, it'll try to output more calculations, which helps with the discovery within a limited context. So the behavior of the L256 with my proposed training method, inspired by the DeepSeek AI team, proved that a smaller, efficient model can reach the performance of a much larger model through post-training methods.

The L_{512} model has a stronger base performance because it performed longer chains, but sometimes diverged from the originally generated English-only chain-of-thought because of the more space in the reasoning chain it can use. Thus, this makes the models achieve a slightly lower accuracy overall than the English-only runs.

5 Discussion

The experiments confirm that combining GRPO with SFT methods can help with more stable learning and inference. The SFT stage improves consistency and reinforces good behavior, and the last GRPO runs allow the model to explore the next solution better than noisy GRPO runs.

The larger L_{512} context helped retain more intermediate steps but also introduced more creative reasoning and noise with just one run, suggesting that reinforcement learning with the sole goal of getting one correct answer will sometimes force the model to find shortcuts in the hope of getting the maximum rewards.

6 Future Work

My future work will be trying to apply that technique to help a non-reasoning model like Gemma 3 to solve Calculus problems with a more comprehensive Chain-Of-Thought using GRPO. Also, I'll be exploring the intersection between Reinforcement Learning and Neuroscience-Inspired AI to see how I can further optimize for local LLM.

7 Conclusion

This project shows that Group-Reward Policy Optimization can enhance structured reasoning in compact language models. Even under limited compute, a multi-stage GRPO $\rightarrow$ SFT $\rightarrow$ GRPO framework enables small-context models to outperform larger zero-shot prompting. By encouraging models to learn through relative self-comparison, GRPO provides a lightweight path toward a more stable numerical reasoning improvement.

References

1. Schulman et al., *Proximal Policy Optimization Algorithms*, 2017. arXiv:1707.06347
2. OpenAI, *Group-Reward Policy Optimization for LLM Alignment*, 2024.
3. DeepSeek-R1, *Emergent Reasoning Behaviors via Group-Relative Reinforcement*, 2025.
4. Qwen3 Team, *Qwen3 1.7B Model Card and Training Details*, 2025.
5. Shao, Zhihong et al., *DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models*, 2024. arXiv:2402.03300