Week 10-11 Report

Mu Junrong

Abstract

This report focuses on the application of Reinforcement Learning (RL) to fine-tune large language models (LLMs) to solve mathematical problems from the GSM8K dataset. The Generalised Reparameterized Policy Optimisation (GRPO) (DeepSeek-AI (2024)) algorithm is implemented, as used in DeepSeek-RL, to train the Qwen2.5-0.5B model. GRPO enables direct optimisation of task-specific rewards by leveraging the differentiability of reparameterized sampling, offering a promising alternative to traditional supervised fine-tuning. The experiments include the Countdown task (Pan (2024)) and mathematical problems provided by the GSM8K dataset (Cobbe et al. (2021)). The results are discussed to analyse the performance of the model after fine-tuning with the GRPO algorithm, and the cause for the difference in reward obtained.

1 Problem Formulation

The task addressed is solving school-level mathematical problems using the large language model (LLM). The GSM8K dataset (Cobbe et al. (2021)) is used, which consists of questions such as:

```
Q: Mary has 5 apples. She gives 2 to John. How many does she have left?
```

The expected output is a natural language or mathematical answer:

```
A: 5 - 2 = 3
```

Traditional supervised fine-tuning trains a model to predict the next token given a large corpus of correct solutions. However, this approach does not optimise directly for the final outcome — generating the correct answer. Reinforcement learning (RL), by contrast, allows us to fine-tune the model using a task-specific reward that directly measures whether the generated output is correct.

1.1 Application of Reinforcement Learning (RL)

The sequence generation process can be seen as a Markov Decision Process (MDP).

Component	**Definition in GSM8K Task**
State s	The initial prompt (i.e., math question prompt) (e.g., "Mary has 5 apples...")
Action a	The generated answer sequence (token-by-token generation)
Policy $\pi_\theta(a \mid s)$	The language model's generation policy
Reward $r(s, a)$	A scalar signal indicating answer correctness (e.g., 1 for correct answer, 0 otherwise)

The mathematical objective is to train the model to maximise the expected reward under the policy:

$$\max_\theta \mathbb{E}_{s\sim\mathcal{D}, a\sim\pi_\theta(\cdot|s)}[r(s,a)]$$

1.2 Framework used: nano-aha-moment

nano-aha-moment (McGill NLP (2024)) is an educational and lightweight framework that demonstrates the application of Reinforcement Learning to fine-tune Large Language Models (LLMs) (e.g., Qwen2.5-0.5B) for solving reasoning tasks. It is based on and inspired by the DeepSeek-RL approach, while it is much smaller in scale (i.e., "nano").

There is no new architecture created in this repo (i.e., it uses a pretrained LLM such as Qwen2.5-0.5B) (Team (2024)). The model is finetuned using GRPO and is used to solve math reasoning tasks. There is a specific reward function and a prompt template for each task. The model is essentially `LLM + GRPO fine-tuning + task-specific reward + prompt template`

2 Methodology and algorithm

2.1 GRPO

GRPO (Generalised Reparameterized Policy Optimisation) is a variant of Proximal Policy Optimisation (PPO) (Schulman et al. (2017)), which reduces variance by leveraging the reparameterization trick, which is a method commonly used in variational inference and differentiable sampling (DeepSeek-AI (2024)). It enables more stable and direct optimisation by maintaining a differentiable path from the sampled action back to the model parameters.

Let $\pi_\theta(a|s)$ be the probability distribution over sequences (answers) given a prompt s, where a is a generated answer.

The Objective function of GRPO (Generalised Reparameterized Policy Optimisation) is defined as

$$J(\theta) = \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}[r(s,a)]$$

The GRPO gradient estimate is:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\epsilon\sim p(\epsilon)}\left[\nabla_\theta r(s, a_\theta(\epsilon))\right]$$

Where

- ϵ is a noise variable (e.g., from Gumbel or Gaussian noise),
- $a_\theta(\epsilon)$ is the reparameterized action, a differentiable function of θ,
- $r(s,a)$ is the scalar reward.

This formulation allows us to use automatic differentiation through the sampling process, reducing the variance of the gradient estimate compared to REINFORCE.

2.2 Reward function

Since GSM8K problems have concise and unambiguous answers, string matching is used as the primary reward signal.

The reward function as:

$$r(s,a) = \begin{cases} 1, & \text{if normalize}(a_{\text{pred}}) = \text{normalize}(a_{\text{gt}}) \\ 0, & \text{otherwise} \end{cases}$$

Where a_{pred} is the model's generated answer, a_{gt} is the ground truth answer.

2.3 Prompt template

The prompt template is designed to clearly separate the question from the expected answer format. This is important as it instructs the model to start generating from the right context.

```
Problem: problem
    Solution:
```

3 Experiment and implementation details

Two Reinforcement Learning experiments are conducted using the GRPO algorithm, the Countdown task, and the task of solving math word problems from the GSM8K dataset.

3.1 Datasets used

The Countdown Task is a symbolic arithmetic reasoning challenge inspired by the Countdown numbers game. In this task, the model is given a set of numbers and a target number, and it must generate a valid mathematical expression that evaluates to the target using any subset of the input numbers with allowed arithmetic operations (`+`, `-`, `*`, `/`). For example, given the input prompt `Numbers: 12, 69, 13. Target: 70`, the output should be `69 - 12 + 13`.

The GSM8K dataset contains thousands of grade school-level math problems, each consisting of a question and a final numerical answer. Unlike the countdown task, GSM8K requires both language understanding and multi-step arithmetic reasoning. This task introduces more challenges, including longer and more diverse language inputs, potential reasoning chains, slight variability in answer formatting and a realistic use case for LLM fine-tuning.

3.2 Experiment setup

The experiment is performed with Qwen2.5-0.5B language model. The following are the setup details.

Component	**Definition in GSM8K Task**
Model used	Qwen2.5-0.5B
Optimizer	AdamW
Hardware	Single NVIDIA 8GB GPU
Number of iterations	35
Episodes per iteration	4
Generations per sample	2
Temperature chosen	0.5

4 Experiment results and discussion

With 35 iterations as mentioned in the experiment setup, the reward obtained generally increases, indicating the model is gradually learning to generate more correct answers and follow the template.

4.1 Countdown task analysis

Quantitatively speaking, during training, we observed an increase in the average reward from 0.01 at first to 0.063 after 35 training iterations. The reward curve obtained demonstrates that the model gradually learned to generate more accurate answers as the policy was optimised using GRPO.

Qualitatively speaking, the model generates non-sensical and incorrect outputs that do not follow the template at first, with many foreign keys. After 35 iterations, the output is correct, although the phrasing can be insignificant and misleading (i.e., which is the cause of the very low reward obtained).

Figure 1: The average reward obtained by the Countdown task with the model Qwen2.5-0.5B

4.2 GSM8K result analysis

The reward increased steadily from an initial 0.01 to approximately 0.054 during 35 training iterations, indicating that the model was gradually learning to produce more correct answers GRPO. This indicates that the reinforcement learning derived from string matching the generated answer with the ground truth is effective, despite being sparse and binary.

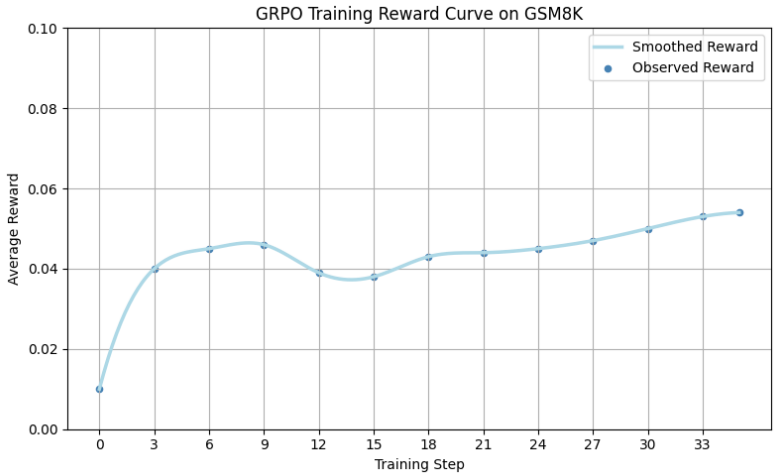

Figure 2: The average reward obtained by the model Qwen2.5-0.5B with the GSM8K dataset.

4.3 Discussion

The main performance difference is caused by the nature of the two tasks. The Countdown task shows faster increasing rate and final reward obtained, since it has simpler prompts, smaller output space and fewer tokens to generate. Thus, the model can quickly learn to

output correct expressions from a small number of examples, which is represented by the faster rewards rising rate in the first iterations.

GSM8K datasets contain more natural language questions. GSM8K requires understanding the question, doing multiple math reasoning internally and outputting the exact final number. Thus, the reward increases at a much slower rate, with a smaller final reward obtained after 35 iterations.

5 Conclusion and future directions

In this project, the application of Generalized Reweighed Policy Optimization (GRPO) is explored, to fine-tune large language models (LLMs) for reasoning tasks. The training pipeline from the nano-aha-moment framework is adapted to two distinct tasks, the symbolic arithmetic-based Countdown task and the grade-school math reasoning dataset GSM8K respectively.

The experiments demonstrate that GRPO can effectively guide LLMs toward better task-specific performance by leveraging structured reward signals during generation. Both tasks shows GRPO's effectiveness even when using smaller-scale models such as Qwen2.5-0.5B.

In the future, other Reinforcement Learning (RL) will be explored to fine-tune LLMs, including combining GRPO and RLHF to improve LLMs' reasoning ability.

References

Karl Cobbe, Vineet Kosaraju, Jacob Hilton, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

DeepSeek-AI. Deepseek-rl: Enhancing llm reasoning via reinforcement learning, 2024. URL https://github.com/deepseek-ai/DeepSeek-RL.

McGill NLP. nano-aha-moment: Grpo on small-scale llms, 2024. URL https://github.com/McGill-NLP/nano-aha-moment.

Jiayi Pan. Countdown tasks 3 to 4, 2024. URL https://huggingface.co/datasets/Jiayi-Pan/Countdown-Tasks-3to4.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Qwen Team. Qwen2.5-0.5b model card, 2024. URL https://huggingface.co/Qwen/Qwen2.5-0.5B.