How do you design good rewards for RL on LLMs?

Justin Qiu

Reinforcement Learning for LLMs

Various optimization algorithms

- PPO
- DPO
- GRPO
- etc.

Proximal Policy Optimization Algorithms

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov
OpenAI
{joschu, filip, prafulla, alec, oleg}@openai.com

DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models

Zhihong Shao[1,2*†], Peiyi Wang[1,3*†], Qihao Zhu[1,3*†], Runxin Xu[1], Junxiao Song[1]
Xiao Bi[1], Haowei Zhang[1], Mingchuan Zhang[1], Y.K. Li[1], Y. Wu[1], Daya Guo[1*]
[1]DeepSeek-AI, [2]Tsinghua University, [3]Peking University
{zhihongshao,wangpeiyi,zhuqh,guoday}@deepseek.com
https://github.com/deepseek-ai/DeepSeek-Math

Direct Preference Optimization: Your Language Model is Secretly a Reward Model

Rafael Rafailov[*†] **Archit Sharma**[*†] **Eric Mitchell**[*†]
Stefano Ermon[†‡] **Christopher D. Manning**[†] **Chelsea Finn**[†]
[†]Stanford University [‡]CZ Biohub
{rafailov,architsh,eric.mitchell}@cs.stanford.edu

$$J_{\mathrm{GRPO}}(\theta) = \mathbb{E}_{q\sim P(Q),\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\mathrm{old}}}(O|q)}\Bigg[\frac{1}{G}\sum_{i=1}^{G}\Bigg(\min\Bigg(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{\mathrm{old}}}(o_i|q)}A_i,\ \mathrm{clip}\Bigg(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{\mathrm{old}}}(o_i|q)},1-\epsilon,1+\epsilon\Bigg)A_i\Bigg)\Bigg) - \beta D_{\mathrm{KL}}(\pi_\theta \,\|\, \pi_{\mathrm{ref}} f)\Bigg]$$

Reward Modelling

- RLHF: trained neural reward model to mimic human behavior
- R1: rules-based binary rewards for deterministic reasoning problems like math and coding
- RLVR: model-based soft rewards for general reasoning problems
- More…

Training language models to follow instructions with human feedback

Long Ouyang* Jeff Wu* Xu Jiang* Diogo Almeida* Carroll L. Wainwright*
Pamela Mishkin* Chong Zhang Sandhini Agarwal Katarina Slama Alex Ray
John Schulman Jacob Hilton Fraser Kelton Luke Miller Maddie Simens
Amanda Askell† Peter Welinder Paul Christiano*†
Jan Leike* Ryan Lowe*
OpenAI

DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning

DeepSeek-AI
research@deepseek.com

Crossing the Reward Bridge:
Expanding RL with Verifiable Rewards Across Diverse Domains

Yi Su[*,1,2], Dian Yu[1], Linfeng Song[1], Juntao Li[2], Haitao Mi[1], Zhaopeng Tu[1], Min Zhang[2], and Dong Yu[1]
[1]Tencent AI Lab
[2]Soochow University

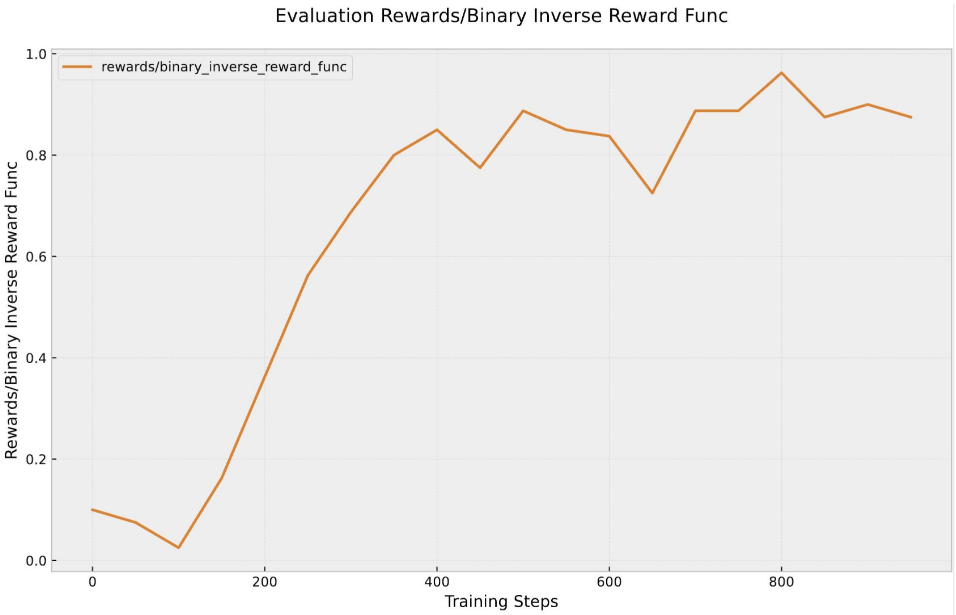
Teaching LLMs Linear Algebra

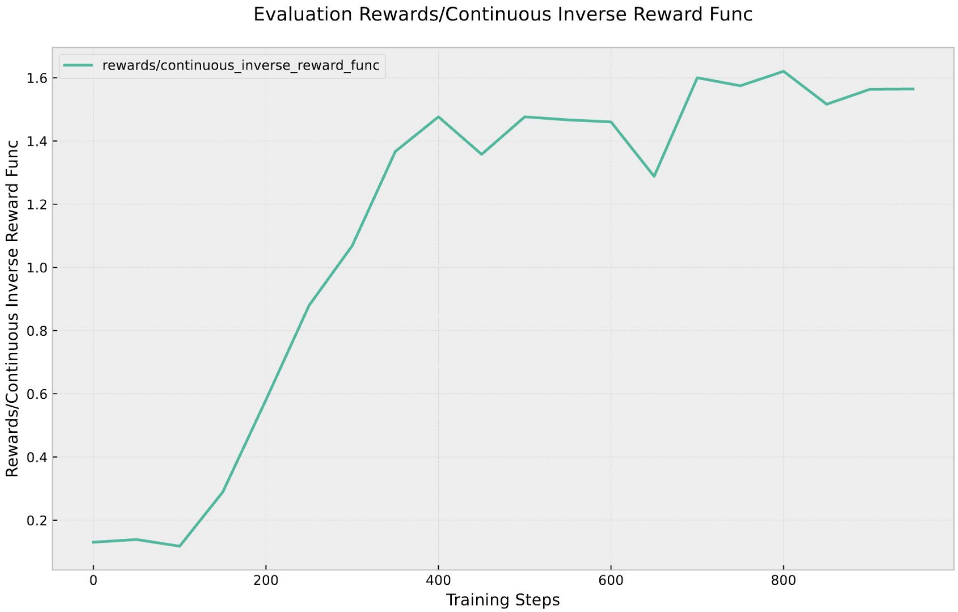
- Matrix inversion and RREF solver
- The problem has clear rewards! We can just design them logically…
  - Binary correctness: I(A * output = I)
  - Row by row correctness: Σ I(row=row*)
  - Continuous correctness: $2 \cdot exp(\frac{norm_{L1}(inv-inv^*)}{tol})$
  - Various format rewards
- We are able to do this because these problems have clear solutions

Teaching LLMs Linear Algebra

Update from last time (model for 2x2 close identity)

It trains and achieves ~90% evaluation accuracy (chart on left is accuracy)





What about general rewards?

- Desirable behaviors like creative writing, unstructured reasoning, etc. don’t have obvious reward functions!
- Reward desiderata
    - Can be applied to general domain
    - Informative
    - Dense: sparse rewards like binary accuracy make training harder and slower
    - Hard to reward hack (frequent with rewards that are bad proxies, or model based rewards)
- Examples: embedding reward, trained reward model
- New idea: LLMs have a lot of information that can be extracted with their logits!

Perplexity Reward

- Perplexity quantifies model's uncertainty for given sequence of tokens.

$$PP(W) = P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}}$$

- We can calculate perplexity from the logits of the language model
    - (logits are essentially how likely each output is at each given point)
- Therefore, higher perplexity of an output means that the model is less certain of it. We want correct answers to have low perplexity
- This is great! We have a **continuous reward for any text generation domain**!
- Problem: perplexity of specific output is high even if answer is right!

Improving the perplexity reward

- Idea: **relative perplexity reward** (Gurung and Lapata, 2025)

$$I_{\pi^{\mathcal{G}}}(x, y, a) = [\frac{PPL_{\pi^{\mathcal{G}}}(y|x) - PPL_{\pi^{\mathcal{G}}}(y|x, a)}{PPL_{\pi^{\mathcal{G}}}(y|x)}]$$

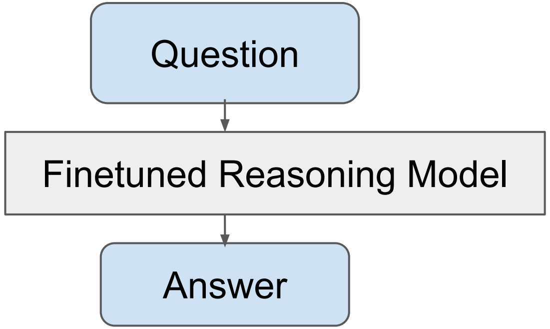
- Reward is based on how much the improved model decreases perplexity!
- Problem: this value ranges from -inf to 1 and is very unstable!
- Another problem: even if perplexity goes from 1e+9 to 1e+7, which gives a reward of 0.99, we still want absolute perplexity to be lower!

Improving the perplexity reward

- Idea: thresholding perplexity to make reward more stable
    - Paper used fancy thresholding but I found that setting min to 0 (representing an increase in perplexity with additional reasoning trace) worked fine, sort of like a ReLU function
- Idea: using hybrid reward that combines relative perplexity gain and absolute perplexity

$$Reward = \frac{max\left(0, \frac{PPL_{\pi^G}(y|x) - PPL_{\pi^G}(y|x,a)}{PPL_{\pi^G}(y|x)}\right) + \frac{1}{PPL_{\pi^G}(y|x,a)}}{2}$$

Old Formulation



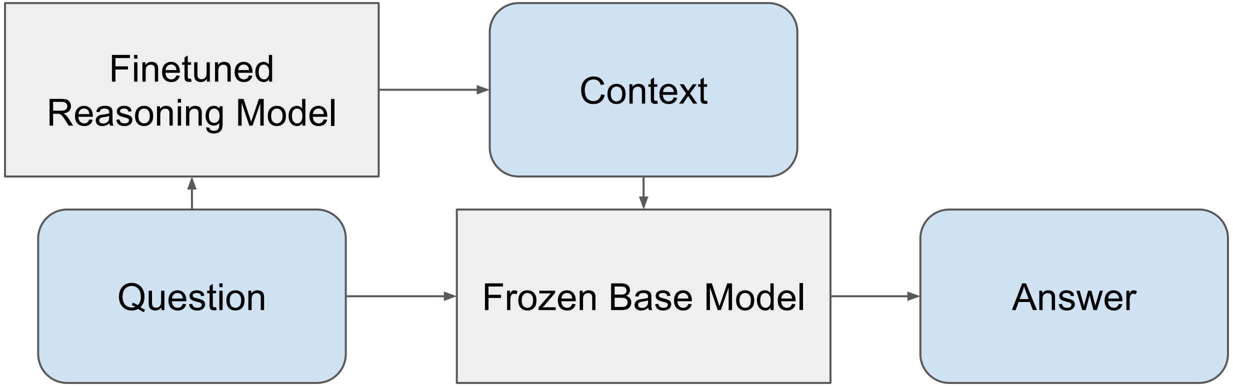
New Formulation

Hypothesis: There exist multiple contexts c such that

P_base(Answer | c, Question) is very high.

-> What if we finetuned a model to just generate reasoning

traces (c) and fed the reasoning as context to a base model?



GSM8k

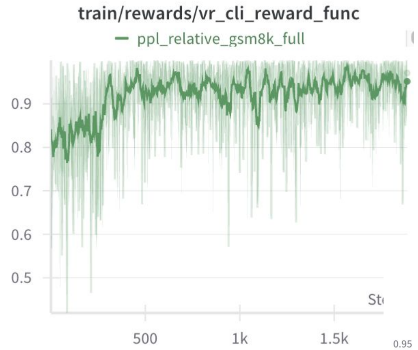
- Math word problem dataset
- GSM8k already has viable rules-based rewards, but it is still a useful dataset to experiment on

question string · *lengths* 137↔232 46%	answer string · *lengths* 50↔168 19.9%
Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May?	Natalia sold 48/2 = <<48/2=24>>24 clips in May. Natalia sold 48+24 = <<48+24=72>>72 clips altogether in April and May. #### 72
Weng earns $12 an hour for babysitting. Yesterday, she just did 50 minutes of babysitting. How much did she earn?	Weng earns 12/60 = $<<12/60=0.2>>0.2 per minute. Working 50 minutes, she earned 0.2 x 50 = $<<0.2*50=10>>10. #### 10
Betty is saving money for a new wallet which costs $100. Betty has only half of the money she needs. Her parents…	In the beginning, Betty has only 100 / 2 = $<<100/2=50>>50. Betty's grandparents gave her 15 * 2 = $<<15*2=30>>30. This…

Results

Baseline accuracy: 0.72

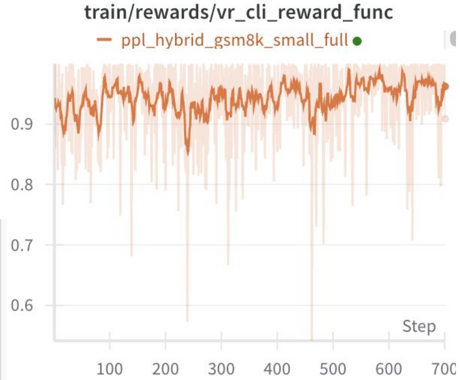
Relative perplexity



Training reward: 0.94

Eval accuracy: 0.73

Hybrid perplexity (starting from relative perplexity checkpoint @ 1k)



Training reward: 0.97

Eval accuracy: 0.73

Absolute perplexity only

Model didn't really train for 500 steps, reward was too sparse.

Inference Examples

Question: Tabitha has 25 dollars. She gives her mom 8 dollars and invests half what is left in a money market. She spends some money on 5 items that costs 50 cents each. How much money does Tabitha have left?

Generated Context: To find out how much money Tabitha has left, we need to follow the order of operations and break down the problem into smaller steps.

Step 1: Tabitha starts with 25 dollars, then gives her mom 8 dollars. Tabitha initially had 25 dollars, then she gave away 8 dollars, so we need to subtract 8 from 25. 25 - 8 = 17

Step 2: Tabitha invests half of what is left in a money market. Since she had 17 dollars left, half of that is 17 / 2 = 8.5

Step 3: She invested 8.5 in the money market, so we need to subtract that from the amount she had before investing. 17 - 8.5 = 8.5

Step 4: Tabitha spends 5 * 50 cents (5 dollars) on 5 items that cost 50 cents each. 5 * 0.50 = 2.5

Step 5: Tabitha had 8.5 dollars and spent 2.5 dollars. 8.5 - 2.5 =

Inference Examples

Question: Claire wants to make 2 cakes for her mother. Two packages of flour are required for making a cake. If 1 package of flour is $3, how much does she pay for the flour that is enough to make 2 cakes?

Generated Context: To find out how much flour Claire needs for 2 cakes, we first need to determine how many packages of flour are required for 2 cakes. Since each cake requires 2 packages of flour and she is making 2 cakes, she needs 2 cakes * 2 packages per cake = 4 packages of flour.

The cost of 1 package of flour is $3, so the cost of 4 packages of flour would be 4 packages * $3 per package = $12.

Therefore the answer is:

Compared to before finetuning

Question: Janet’s ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?

Generated Context: To solve this problem, we need to follow these steps:

1. First, we need to calculate the total number of eggs laid by the ducks in a day, which is given as 16 eggs.

2. Since Janet eats three eggs for breakfast and bakes four for her friends, she uses a total of 3 + 4 = 7 eggs.

3. To find the number of eggs remaining, we subtract the used eggs from the total number of eggs laid: 16 - 7 = 9 eggs.

4. Since Janet sells the remaining eggs at $2 per egg, we multiply the number of remaining eggs by the selling price to find the total amount she makes: 9 eggs * $2/egg = $18.

The answer is 18.

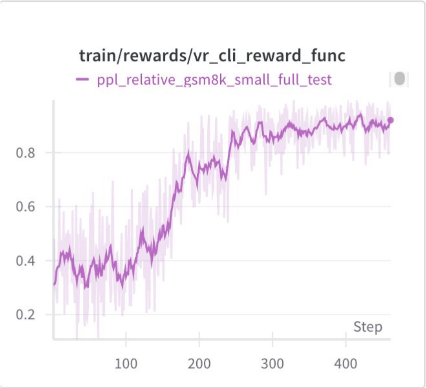
Using smaller base model?

- Would a smaller model work? Currently, the 7B base model was already pretty good at solving GSM8k (accuracy of ~70% without finetuning). 0.5B model accuracy is 10%.
- Wanted to also experiment with small base model but large context model; ran out of time

Small Model Results

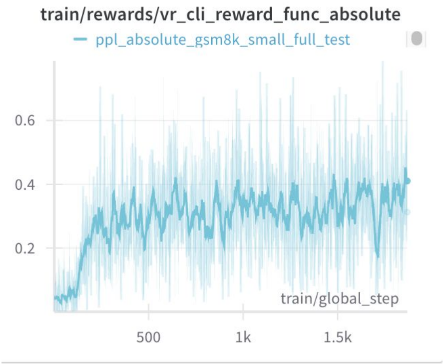
Baseline accuracy: 0.11

Relative



Hybrid

Absolute



Training reward: 0.92

Eval accuracy: 0.14

Training reward: 0.64

Eval accuracy: 0.15

Training reward: 0.41

Eval accuracy: 0.11

Has problems with repetition

To calculate her annual salary, we need to multiply her hourly salary by the number of hours she worked as a teacher and her hours worked as a cheerleading coach.
Her weekly salary is 50 weeks x $20 = $1000.
Her monthly salary is 35 weeks x $20 = $700.
Her annual salary is 35 weeks x $20 = $700.
Her weekly salary is 15 weeks x $30 = $450.
Her monthly salary is 15 weeks x $30 = $450.
Her annual salary is 15 weeks x $30 = $450.
Her annual salary is 35 weeks x $30 = $1050.
Her annual salary is 15 weeks x $30 = $450.
Her annual salary is 35 weeks x $30 = $1050.
Her annual salary is 15 weeks x $30 = $450.
Her annual salary is 35 weeks x $30 = $450.
Her annual salary is 35 weeks x $30 = $450.
Her annual salary is 35 weeks x $30 = $450.
Her annual salary is 35 weeks x $30 = $450.
Her annual salary is 35 weeks x $30 = $450.
Her annual salary is 35 weeks x $30 = $450.
Her annual salary is 35 weeks x $30 = $450.

Why weren’t training results very good?

- Overall problem: we were able to decrease the perplexity of desired outputs, but not enough to where the model generates our desired output
  - Model still prefers things like “The answer is {answer}” or “\boxed{answer}” even though we decreased the perplexity of the answer “{answer}” by quite a bit
  - Maybe training for longer would help here, but I was limited by compute
- Try a textual reasoning dataset; I tested with another one (HellaSwag) but it wasn’t a reasoning dataset and our framework for generating reasoning traces wasn’t helpful
- I still think the framework is interesting; paper from Gurung and Lapata last month (inspiration for this) found that the framework works for creative story generation but only does human evaluation

Training details

Did not get a chance to hyperparameter tune too much, there are too many experiments to run and not enough GPUs. So results can probably be marginally improved.

Scheduler/Learning Rate	Cosine scheduler, 0.1 warmup ratio
Learning Rate	0.000005
Optimizer	Paged_adamw_8bit
Optimizer Hyperparameters	beta1=0.9, beta2=0.99
Batch Size	1 (otherwise OOM)
Gradient Accumulation Steps	4
GRPO configs	6 generations per batch
Regularization	0.1 weight decay, 0.1 max grad norm (clipping)

Other Implementation Details

Used Unsloth as general training framework; made finetuning a lot easier

Used TRL and Trainer (huggingface libraries) for training with GRPO

Hardware: 1 RTX A6000 for the 7B model runs, each full run took 2-3 days, 1 RTX 2080 for the 0.5B model runs, each full run took ~12 hours

Models: Llama-3.2 8B instruct (for large experiment), Qwen2.5 0.5B Instruct (for small experiment)

Parameter efficient methods

- Problem: my GPU has only 48GB VRAM, and I only have one
- Solution: parameter efficient methods

Quantization: all models loaded in 4 bit precision -> huge memory savings with minimal performance degradation

LoRA: used low-rank (rank 16) matrices to approximate weight updates instead of updating all model weights -> finetuning is much faster and less resource intensive!

Reflection

What was the number one technical or conceptual difficulty?

- Compute limitations BY FAR! My training scripts spent almost a week each time in slurm queue towards the end

What part of the project workflow was easier than expected? Harder?

- Easier: working with Unsloth and TRL. They did a great job with making the frameworks easy to use and relatively easy to debug! The same can’t be said for VERL…
- Harder: working with the GPUs. Also coming up with new ideas, although Professor Davis helped significantly on this aspect!

How did your project goals or approach evolve based on self-critiques or intermediate results?

- My project significantly changes throughout the semester. I began the semester exploring rewards in easily verifiable settings like linear algebra problems, but pivoted towards experimenting with flexible and more general-purpose rewards. A large part of this was that Professor Davis had the interesting idea of training a context model to generate reasoning traces.

How did AI tools assist your project (e.g., coding, debugging, writing, brainstorming)?

- AI helped a lot with initial coding iterations and debugging GPU and SLURM problems.

Individual Contributions

What was the most surprising result or finding during the project?

- My formulation didn’t work as well as I expected. I think the biggest problem was compute (common trend!) which prevented me from iterating on ideas quickly.

Which specific lecture topic, concept, or technique from the course was most useful for your project? Explain how.

- The last few lectures on best practices for training large language models were very helpful, as they helped me better understand the workflow. Even though I wasn’t able to apply it super well for this project due to compute and time limitations, it was still tremendously helpful

How has your perspective on applying optimization in practice changed since the beginning of the course?

- I’ve gotten a deeper understanding and appreciation of the math behind the optimization and how optimization makes all of modern deep learning work.

If you had two more weeks, what specific next step would you take on the project?

- Test on other datasets! Was going to use some textual reasoning datasets like HellaSWAG but the model didn’t finish training :(

If you could restart the project, what is the single biggest change you would make to your approach or plan?

- I would’ve tried to find a teammate interested in the same thing. In the beginning of the semester, I got the wrong impression that most people were working on a finance project after asking only three other people.