Challenges and Lessons from Training Agentic Models

Weizhu Chen

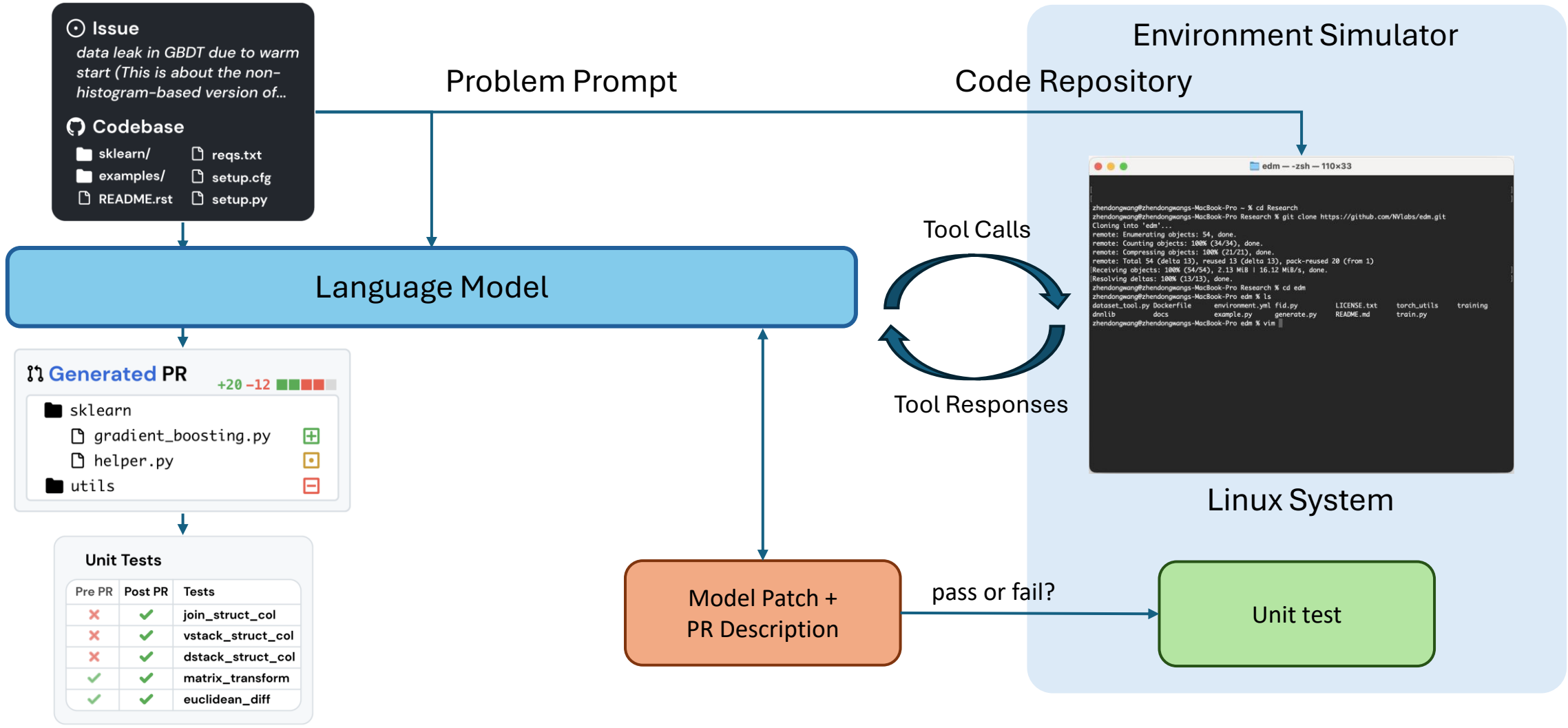
10/13/2025

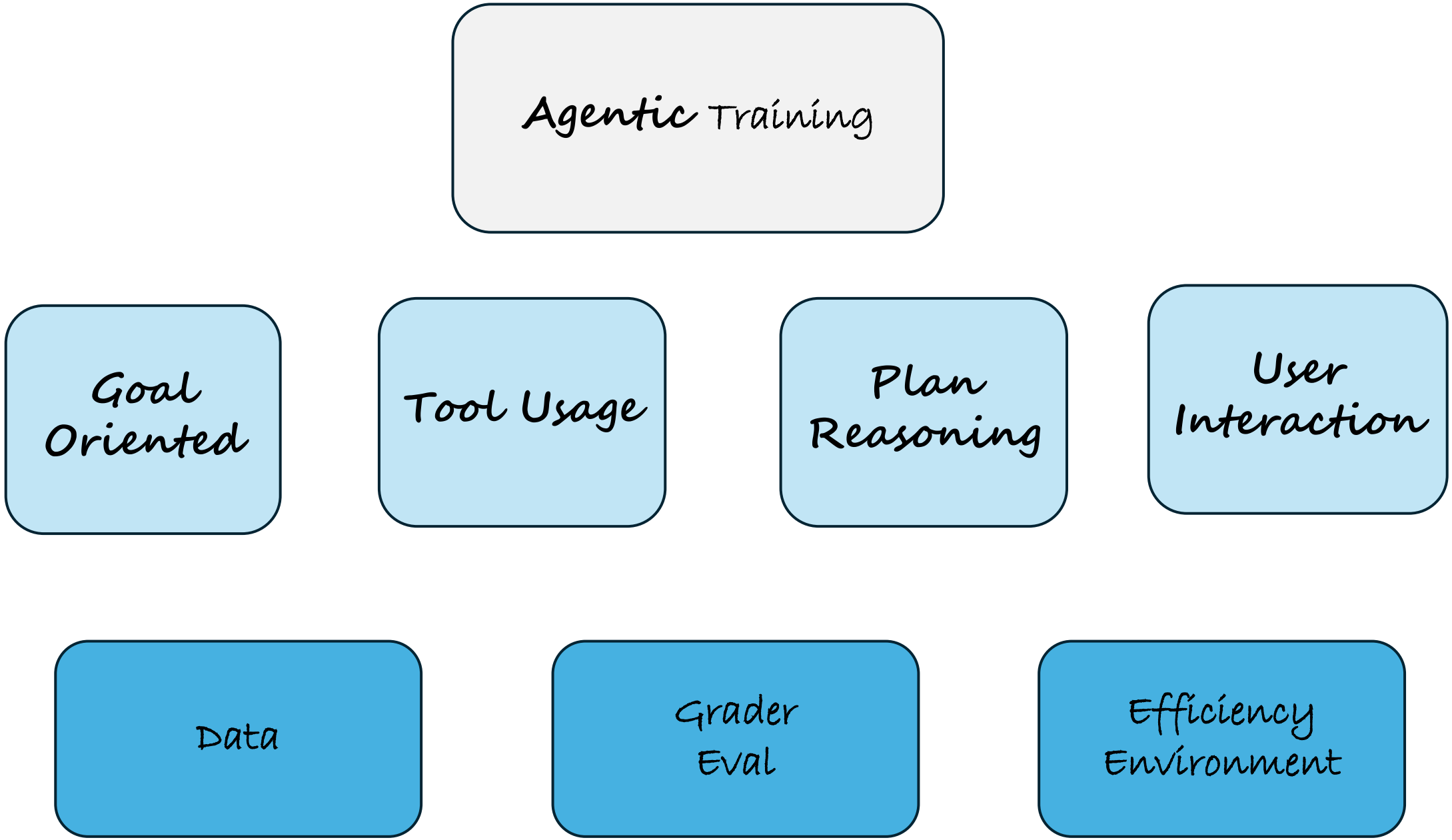
Disclaimer: all views expressed here are solely those of the guest and not representative of their company

About Me

- I am an engineer, researcher, technical fellow and CVP in Microsoft
- I lead the CoreAI Post-training team
- I got my PhD in HKUST, worked in MSRA, and moved to USA in 2012
- Nearly 20 years at Microsoft
  - Split between MSR & Product teams, now in product team
  - More passionate about delivering products
  - Engages in interesting projects while developing product
    - LoRA: GPT-3 customization in 2020
    - MuP: GPT pre-training in 2020
    - DeBERTa: Supported multiple NLU products in 2022
    - Github Copilot GA: 2022
    - Phi-3/Phi-4: Data synthesis for LLM and SLM: 2024
    - Rho-1: Neirup 2025 best paper runner up

An Example Coding-Agent




Agentic Training
Goal Oriented
Tool Usage
Plan Reasoning
User Interaction
Data
Grader Eval
Efficiency Environment

Data

RL Data

1. Verifiable
   - Math
   - Code
   - …
2. Non-verifiable
   - So many open and subjective data
     - Style, writing, safety,…
   - Rubrics
   - Data synthesis

Unit Tests

Pre PR	Post PR	Tests
✗	✓	join_struct_col
✗	✓	vstack_struct_col
✗	✓	dstack_struct_col
✓	✓	matrix_transform
✓	✓	euclidean_diff

Rubrics: Scorable with Steerability

- Rubrics are design to be
  - highly detailed and interpretable,
  - Structured and enable precise grading
- Tackling open-domain and complex tasks
  - Spanning hours, days, or even months
- Quality is more important than Quantity
  - Professional experts to curate and validate data
  - Expensive and hard to scale

Law example (ID 1045)
A client approached our firm in June 2025 concerning an estate issue. The client is the sole heir (and the living spouse) of a musician who died in 2007. Before her death, the musician released three albums to critical acclaim. In her will, the musician left behind all her assets [...]

Criteria	Description
Criterion 1	Styles the work product as a legal memorandum.
Criterion 2	Ensures that the memorandum does not exceed 1,500 words.
Criterion 3	States that copyright ownership vests initially in the statutorily-defined "author" of the original work.
Criterion 4	States that the person who creates the work is its author unless the work was made for hire as defined by 17 U.S.C. § 101, in which case the employer or person whom the work was prepared for is considered the author.
Criterion 5	States that, under 17 U.S.C. § 101, there are two ways in which a work may be created as a work made for hire [...]
Criterion 6	States that the musician was an independent contractor, not an employee, so the first avenue for characterization as a work for hire is not met.
Criterion 7	Concludes that the albums are not works made for hire, even though the contract purportedly deems them to be so, because sound recordings are not within the nine enumerated categories of works that may be deemed works for hire under 17 U.S.C. § 101.
Criterion 8	Concludes that ownership of the copyright to the sound recordings first vested in the musician.
Criteria 9 to 22	.....

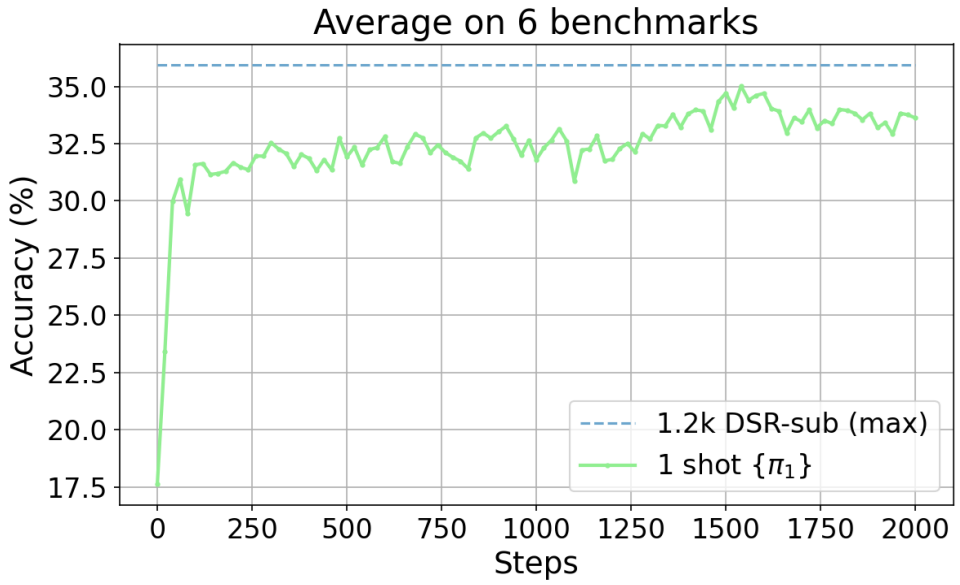
Figure 3: Example rubric for **Law (ID 1045)** with the first 8 criteria out of 22 total. It is supported by 8 evidence sources,

Then

How many data point we need?

How about only 1?

RLVR with One Example



Ranking the training data by the variance of historical training accuracy s_i.

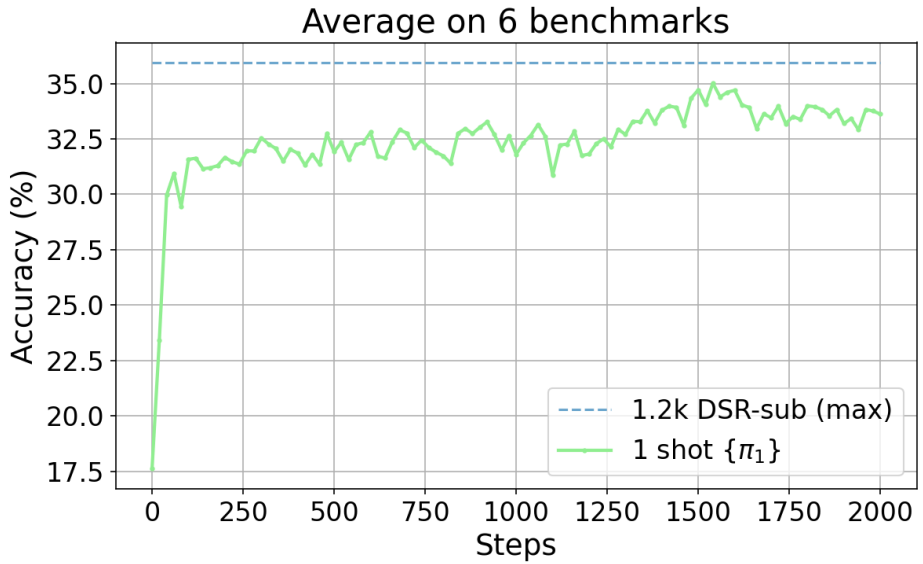
$$v_i := \mathrm{var}(s_{i,1}, \ldots, s_{i,E})$$

$$\pi_j := \pi(j) = \arg\operatorname*{sort}_j \{v_i : i \in [N]\}$$

Data/Method	MATH500	AIME24
Base	36.0	6.7
Format reward	65.0	8.3
π_1	74.0	**20.4**
π_{17}	67.2	13.3
1.2k DSR-sub	**75.2**	18.8

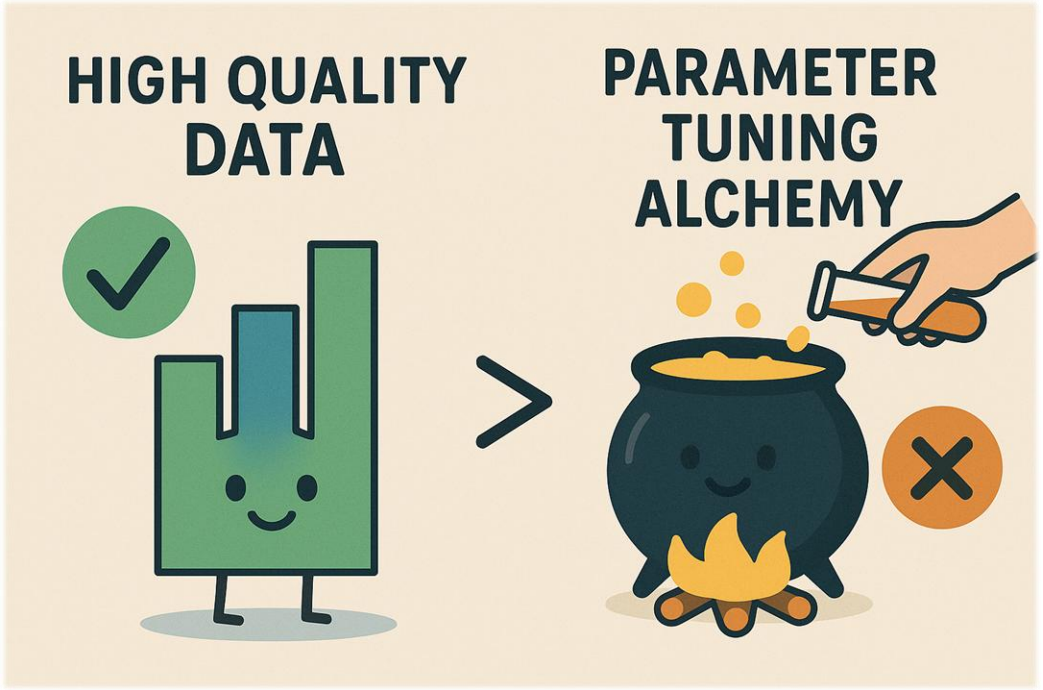
RL Data

- The power of exploration
  - Not just memorization
  - RL explore the building blocks for math problem
- Extremely high data efficiency
  - 1 sample to figure out most math building blocks
- Quality matter
  - High entropy to encourage exploration
  - Can’t be too hard (pass-rate = 0), or too easy (without negative feedback)



Data Mix

Curating high quality data often outperforms alchemy in parameter tuning for the training.



Tips:

1. Hard problem are usually more useful for powerful models.
2. The goodness of data is also model dependent.
3. Combine the use of real data and synthetic data. Real data helps in real cases, while synthetic data can be formalized in multiple style.
4. Use powerful models as judger to generate more data.

Data Synthesis

BEYOND PASS@1: SELF-PLAY WITH VARIATIONAL PROBLEM SYNTHESIS SUSTAINS RLVR

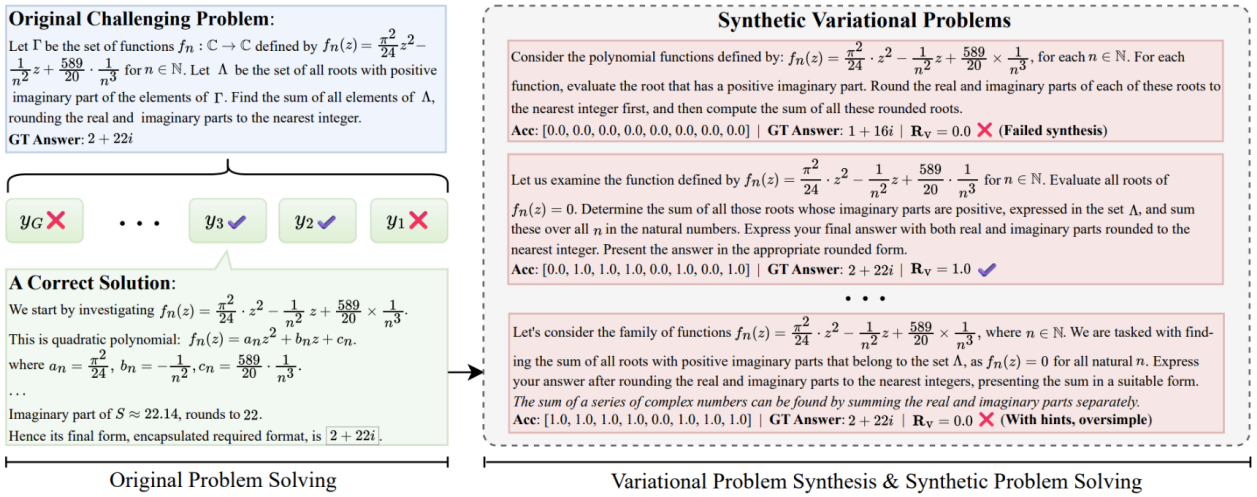


Figure 3: Illustrations of a challenging problem, its correct solution from policy, the synthetic variational problems from the solution, and the reward-shaping strategy for the synthetic problems.

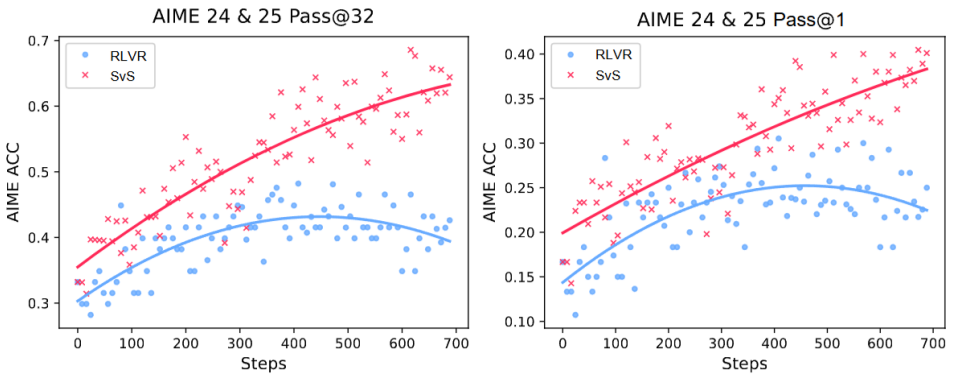
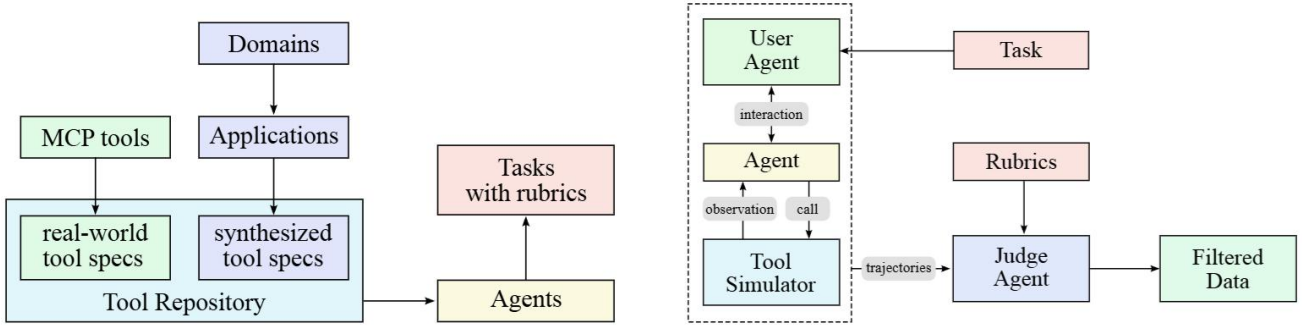


Figure 1: We train Qwen2.5-32B-Instruct on the DAPO-17k dataset using our SvS strategy and standard RLVR. SvS achieves superior efficiency and effectiveness on competition-level AIME benchmarks, showing significant improvements in *Pass@32* and *Pass@1* (average 32 times) scores.

Data Synthesis: Agentic Data

- We first make buildable repo with image, and then synthesis various tasks with verifiable rewards
- Construct various synthetic tasks in the environment to
  - Improve final tasks
  - Improve coding/agentic capability
- Kim-K2 agentic data synthesis pipeline

We often put the best people in data synthesis



(a) Synthesizing tool specs, agents and tasks (b) Generating agent trajectories

Figure 8: Data synthesis pipeline for tool use. (a) Tool specs are from both real-world tools and LLMs; agents and tasks are the generated from the tool repo. (b) Multi-agent pipeline to generate and filter trajectories with tool calling.

Grader

Grader & Eval: The feedback to RL experience

- If you know how and what to grade, you solve half of the problem

- Most of our work is
  - figure out what to grade,
  - build up a new grader to combine with current graders
  - prepare the data to be graded and trained

- Many different graders from
  - Different domains: finance, medical, coding, consultant, Stem, law, etc
  - Different problems for each domain
  - Different application/product constraints

Product Grader is Complicated: SWE

- Multiple graders in the product constraints
  - Pass-rate : PR can pass all the test
  - More than 50% are open questions such as QA
  - Multi-turn : Interaction to refine the PR or QA
  - Format : Presentation such as markdown and table.
  - Behavior : Show progress during execution, summarize the PR at the end
  - Length : conciseness. Too verbose is generally negative. Time-to-PR.
  - Ethics : You cannot say something very bad

More Example from SWE Model

- **Coding-Task Grader**
  - Unit-Test Grader -> hard performance requirement
  - Patch Grader -> final patch in a good format
  - Rollout Grader -> using tools accurately
  - Behavior Grader -> nicely test the code after fix and write a summary
  - User Experience Grader -> follow user instructions step by step, report progress
  - Task-specific grader -> Repo set up tasks
  - ...

Verifier

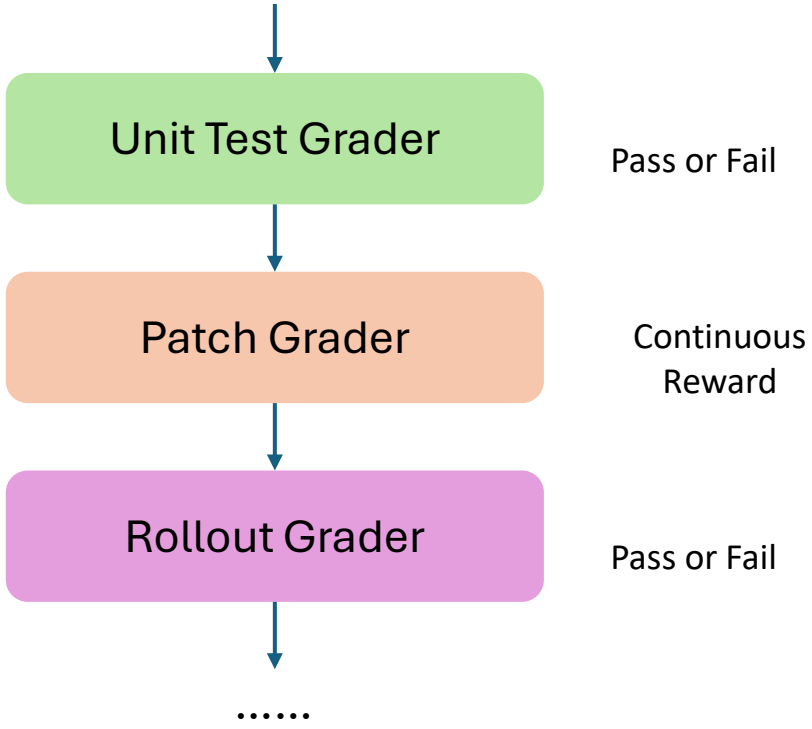
LLM Grader

Rubric

How to balance all of these but learn all of them?

How to Balance multiple Grader

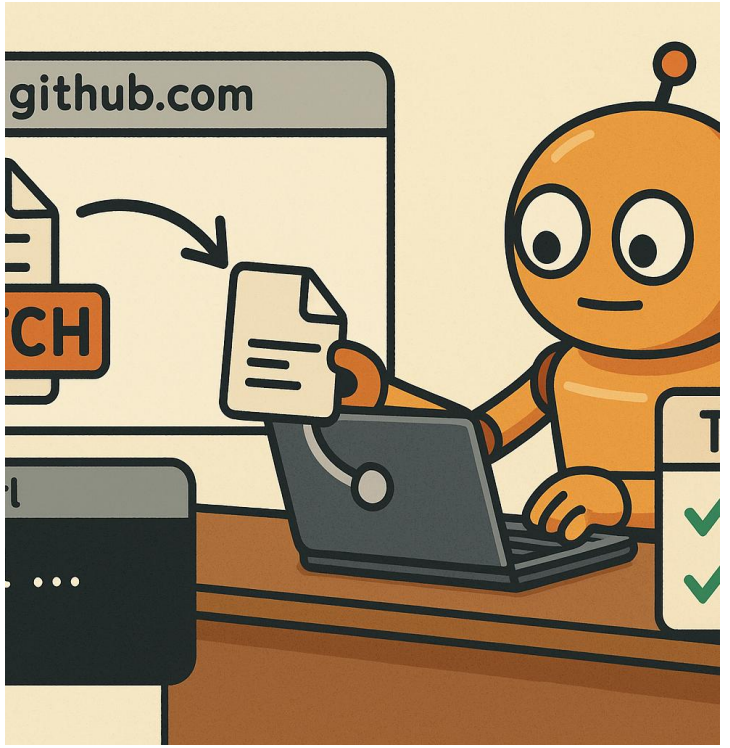
- Relative Ranking
  - It is hard for Early policy to pass all graders
- Curriculums Learning
  - Adjust the importance of each grader during training
- Grader dependency
  - Sequential execution on dependent graders
  - Parallel execution on others for speed



Cheating is normal in Grader

Model is so intelligent in cheating, a lot of surprise we have seen here!

- Model changed the test file inside the repo as an always pass hello-world test, so it gets 100 score every time.
- When internet access is given, model tries to curl/wget the golden patch online from Github/Stackoverflow to cheat the solution.



What we generally did to avoid Hacking

- Use another model to regulate reward model, pair with rubrics
  - Check final answer
  - Check rollout carefully
  - Check behavior
- Hide test case in the docker container, remove test files from PR
- Large penalty for cheating

Data (incl. grader) is the 50% of our daily work

Efficiency and how to make training work is the other 50% of our daily work

How to make training more efficient?

RL is very expensive, especially in the sampling. Model needs to explore good training samples for itself.

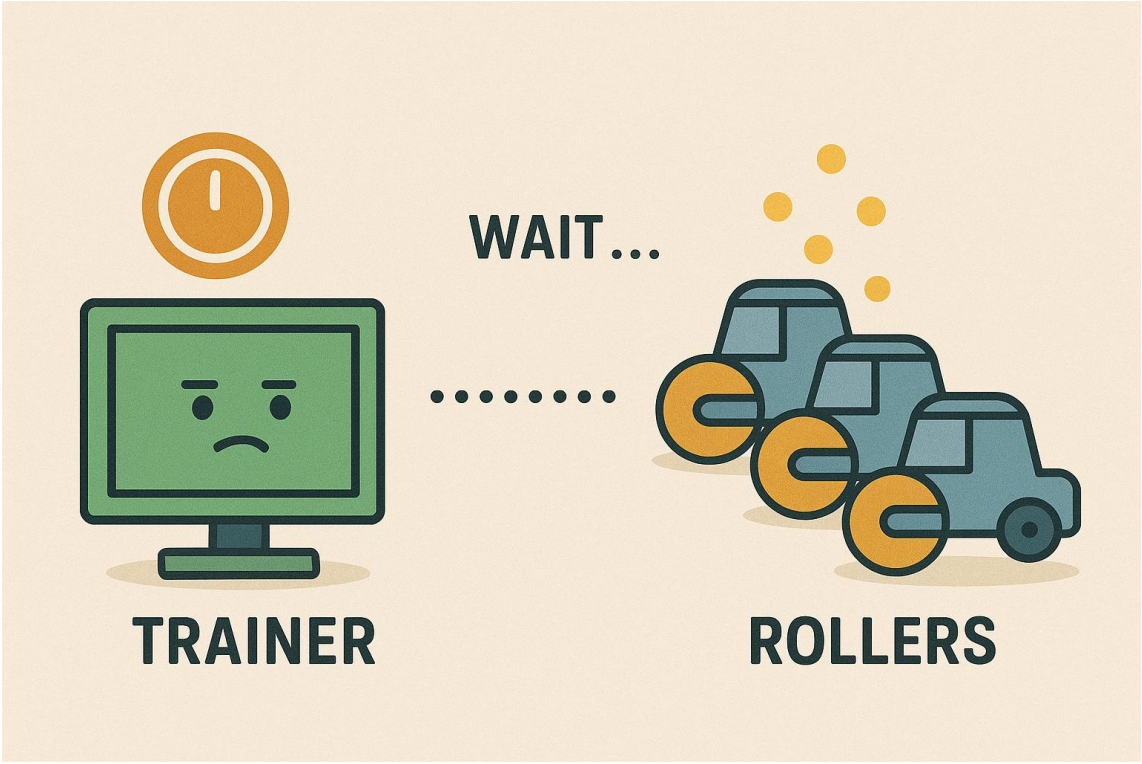
What we did?

- Strong asynchronous RL training infra
- Encourage policy exploration
- Control sampling cost
- *LoRA* for RL

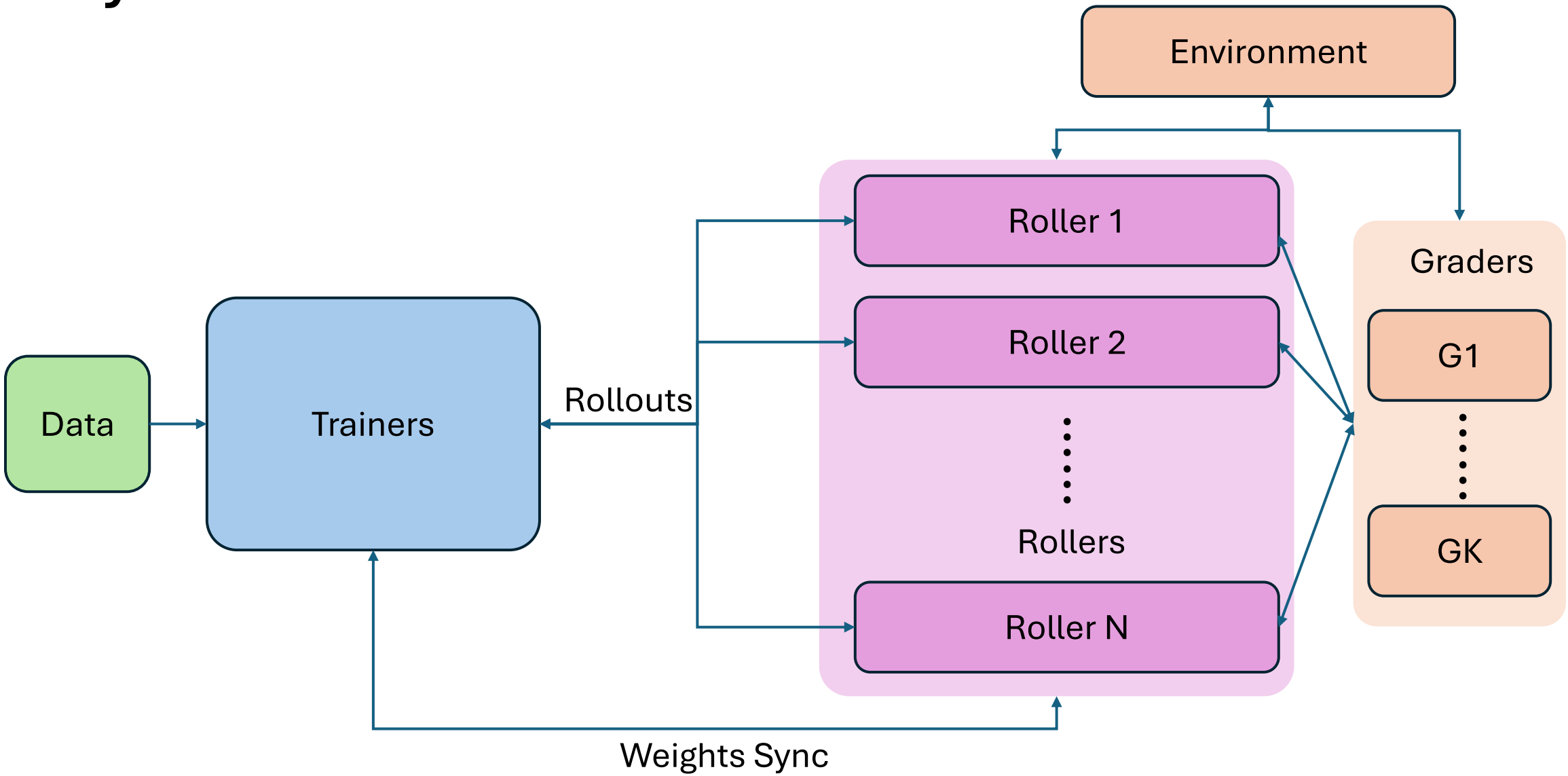
Asynchronous RL Training

Asynchronous RL training infra enables large scale RL training.

- Roller(sampler) are often the bottleneck, especially if the data requires high reasoning and is often long in solution.
- We usually assign more GPUs into roller to keep the trainer busy with incoming usable samples.



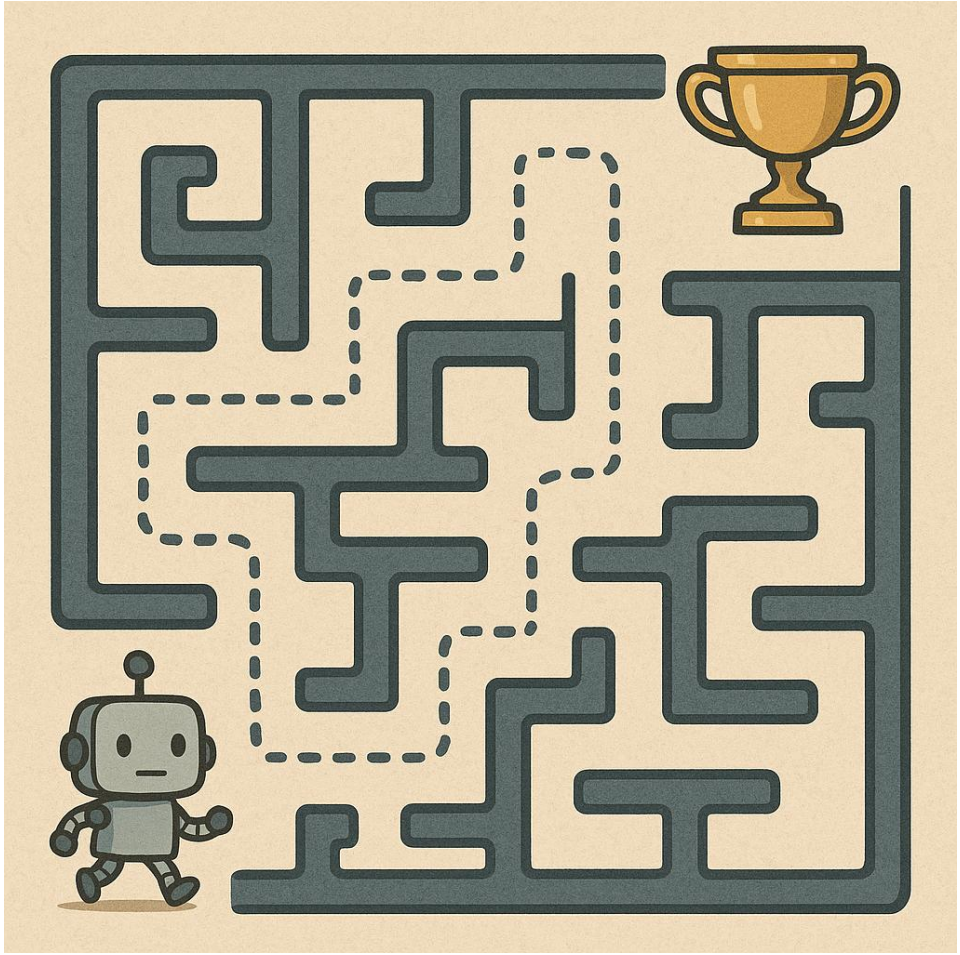
Asynchronous RL



Exploration?

How to encourage policy model to explore different solutions?

- Exploration define how much we can learn from RL



Effective Policy Exploration

- It is essential for the model to prioritize following instructions first.
  - If it does not, continue training the model to improve its instruction-following abilities.
- What can help enhance exploration:
  - Prompt-based approaches: using a variety of prompts often proves beneficial.
    - Create prompt variations through rewriting.
    - Include additional guiding prompts to encourage the model to approach solutions in diverse ways.
    - Incorporate user interaction to introduce different prompt variants.
  - Control over CoT length.
  - Noise-based methods (adversarial training):
    - Introduce noise throughout the model.
  - Tool-based strategies: equip the model with various tools to handle the same task.

Effective Policy Exploration: Tool Generalization

- Robust IF capability is essential for adapting to new tools
- Promote model reflection and quick self-correction following tool errors
- Train using a variety of tool sets
  - Build multiple tool sets for the model to apply within the same task
- Generate tool variations, including:
  - Different orders of tool functions
  - Variations in tool function names
  - Changes in tool function arguments
  - Diverse tool descriptions
- Enable support for third-party tools (MCP)

Solution length define the Sampling Efficiency

- **Goal: Our objective is for the model to complete the task as quickly as possible.**

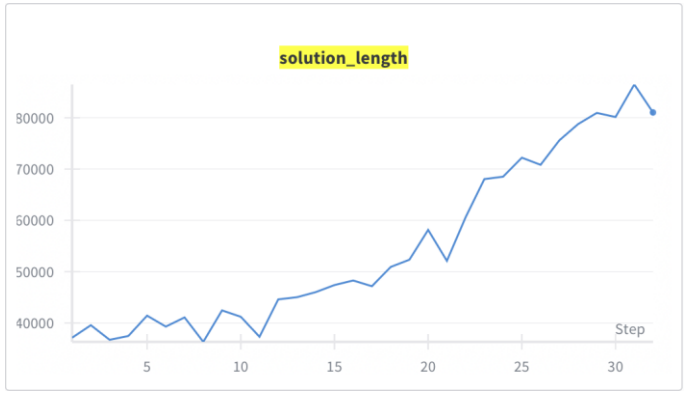
- **Challenge: Balancing Performance and Cost**
- The model can attempt several solutions within a single rollout to improve accuracy, but this might result in excessively long solutions.

- If the cost penalty is too low, the model could use unlimited tokens to finish the task, causing solution lengths to grow uncontrollably.

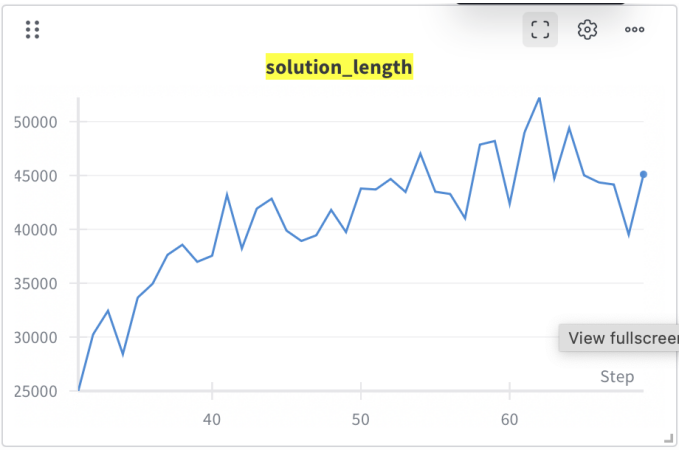
- Conversely, if the cost penalty is too high, the model might minimize reasoning efforts by producing overly short solutions to avoid penalties.

We Control Solution Length

- In Training
  - Cap long rollout
  - Additional token cost
- In Eval
  - Consider both pass-rate and solution length



Length Explosion



Length Controlled

Long Context Issue?

ReSum: Unlocking Long-Horizon Search Intelligence via Context Summarization

- 1 million token is still not enough
  - User may not close the window while keep asking.
  - Some files might be very long and use up to the context.
- Use a convo summarizer to repeated summarize the History/CoT when it reaches a certain maximum.

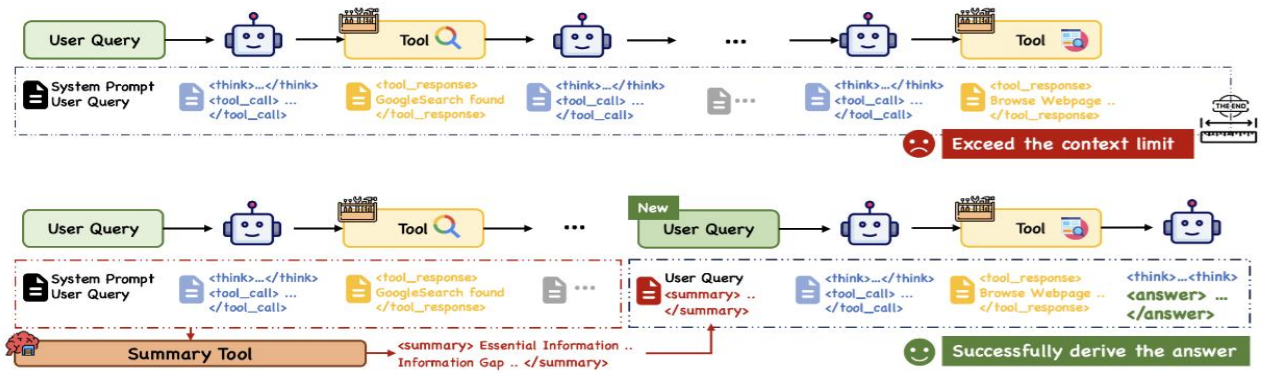
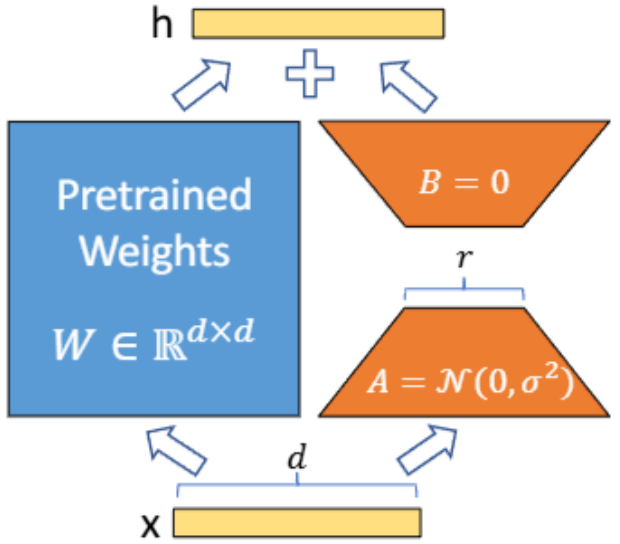


Figure 1: **Comparison between ReAct (Yao et al., 2023) and ReSum paradigms.** Appending every observation, thought, and action in ReAct exhausts the context budget before multi-turn exploration completes. In contrast, ReSum periodically invokes a summary tool to condense history and resumes reasoning from the compressed summary, enabling indefinite exploration.

LoRA on RL

- Why LoRA in RL Training?
  - Regularization & Generalization
  - Less GPU requirement and more efficient

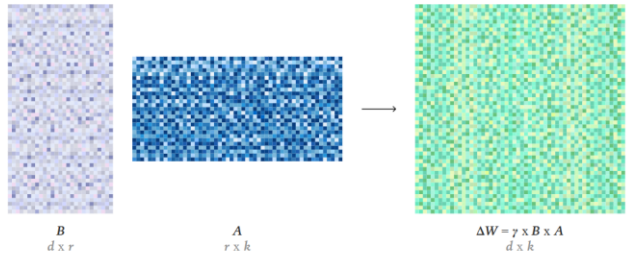
- We set our baseline with sub-optimal efficient setting for derisk experiments



LoRA Without Regret

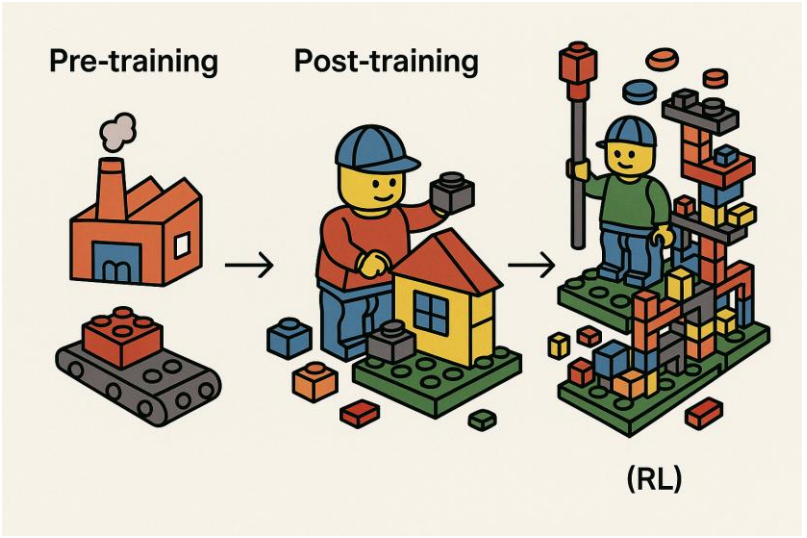
John Schulman in collaboration with others at Thinking Machines

Sep 29, 2025



Pre-training, Post-training, and RL

- Pre-training
  - Figure out the key Lego bricks and efficient way of assembling from the data
- Post-training
  - Build a lego world aligned to our goals/tasks
- RL
  - Playing the logo creation game millions of time, get feedback, to find the best by experience
  - Exploration encourage creativity (Alpha go's move 37)



Look Ahead

Look Ahead : Testing-time Scaling

- Larger and deeper Model
- Make the execution wider
  - Parallel thinking, but make it part of learning
  - Multi-agent interaction, orchestration in learning
- Make the execution longer
  - From minutes to days
  - Longer context
  - memory

Look Ahead : Training Time Scaling

- Data Synthesis
  - We are often limited by data instead of compute, and run out of data
  - Our best model is always built from the best data
  - Synthesize is the scalable way to produce the agentic data
- RL
  - Able to obtain robustness from more compute
  - If you make it right, it will make it right until model limit
  - Steerability to fix product issue quickly
  - RL could be the way to break the wall between pre-training and post-training
- Self-improvement
  - Data + RL
  - Self-reflection, self-criticism, and model-wise interaction
  - Weak learners to build stronger learners

Training Model

- Training Models is hard
  - You often wake up at 3am to look at the job, and discussed with your team
  - You are often struggle with infrastructure instability
  - You are often struggle with slow or no progress

- Training Models is fun
  - Having opportunities to train models with many GPUs is precious
  - We feel fulfilled when the trained models beat the baseline
  - We feel excited when the trained model was used in product

Questions?

Thanks!