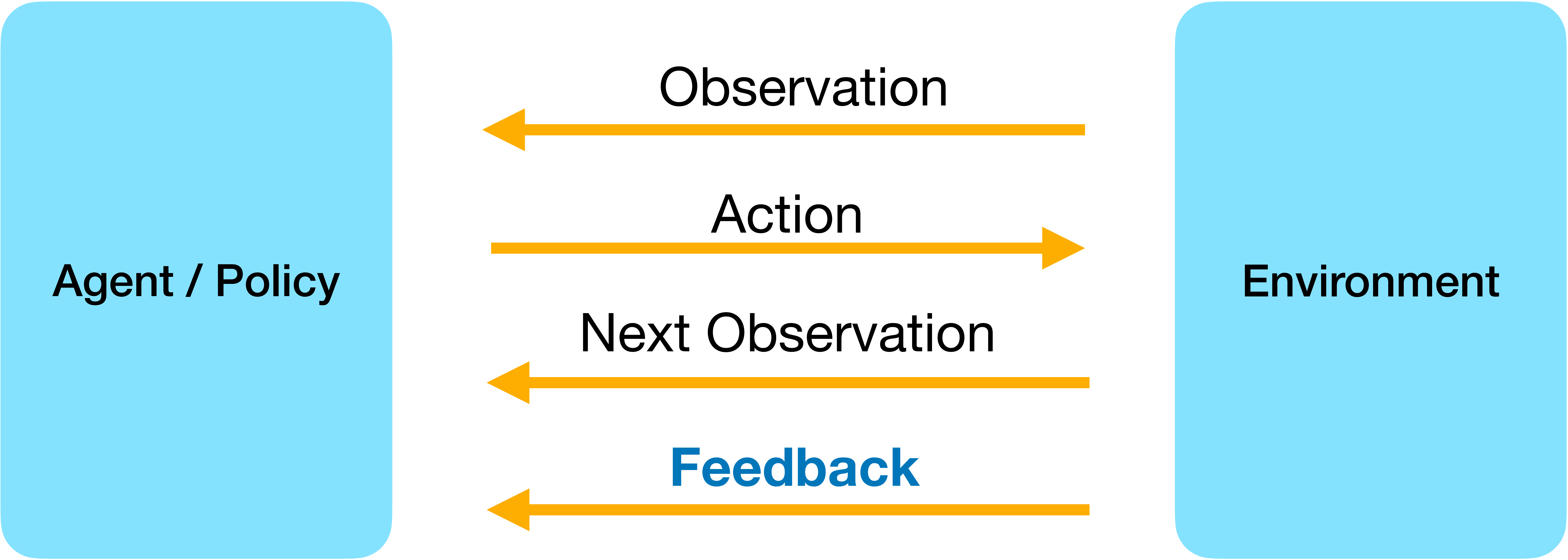
Harnessing Additional Feedback in LLM Post-Training

Or how to think about exploration in LLM-RL

Yuda Song

Oct 29; Albert Gu’s reading group

Interactive Decision Making



Agent / Policy

Observation

Action

Next Observation

Feedback

Environment

What is Feedback? Reward? Where are rewards from?

Harnessing Additional Feedback

1. How do we get feedback when the reward is unreliable?

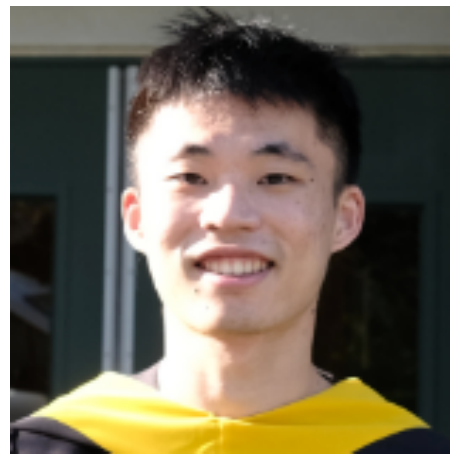
2. How do we get additional signal when there is a reliable reward?

3. How do we efficiently query feedback (with help from synthetic feedback) when it is expensive?

1. How do we get feedback when the reward is unreliable?
2. How do we get additional signal when there is a reliable reward?
3. How do we efficiently query feedback (with help from synthetic feedback) when it is expensive?

Mind the Gap: Examining the Self-Improvement Capabilities of LLMs

ICLR 2025



Yuda Song, Hanlin Zhang, Carson Eisenach, Sham Kakade, Dean Foster, Udaya Ghai

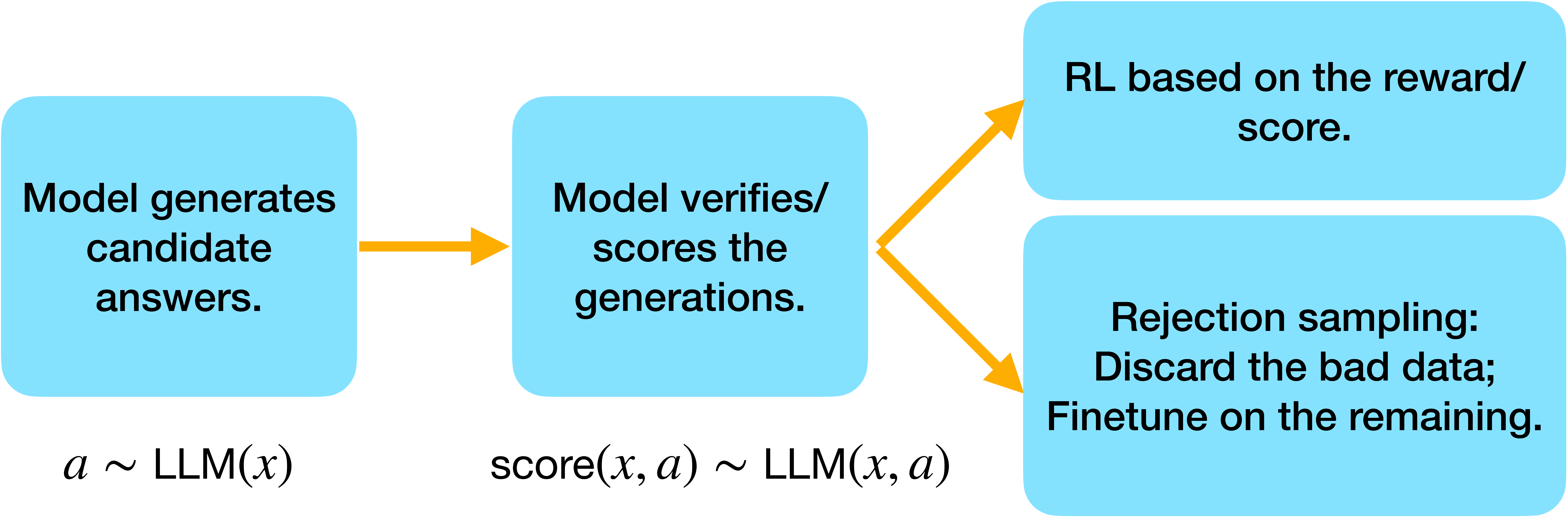
Getting Reliable Feedback is Hard

- Once the models reach super-human level, we might not be able to define the ground truth answer anymore.
- Some problems lack the structure for automatic correctness check (e.g., proofs).
- Problems with structures requires specific parsing.

Why don’t we ask the model itself to provide feedbacks?

A Generate-Verify Pipeline

$$\text{LLM}' = \arg\max \mathbb{E}_a[\text{score}(x, a)]$$



Model generates candidate answers.

Model verifies/ scores the generations.

RL based on the reward/ score.

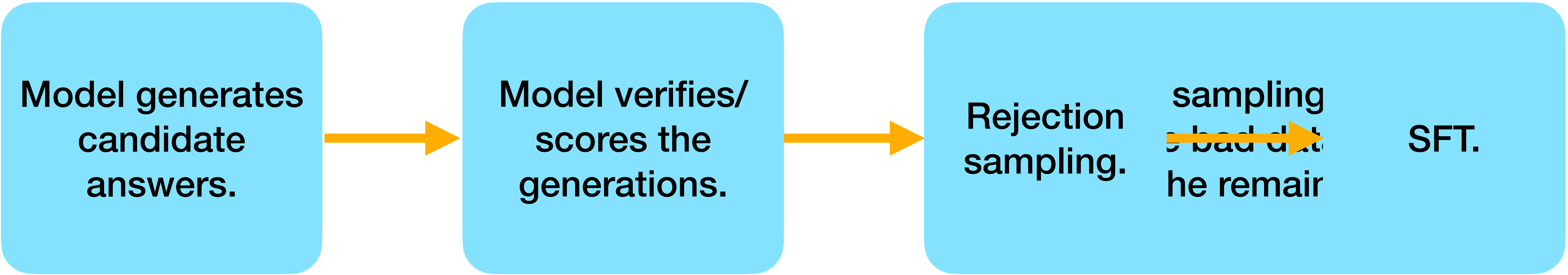
Rejection sampling: Discard the bad data; Finetune on the remaining.

$a \sim \text{LLM}(x)$

$\text{score}(x, a) \sim \text{LLM}(x, a)$

$$\text{LLM}'(a \mid x) = \text{LLM}(a \mid x)\mathbb{I}\{\text{score}(x, a) \geq \tau\}$$

Measuring Self-Improvement



Model generates candidate answers.

Model verifies/ scores the generations.

Rejection sampling.

sampling
e bad dat
he remair

SFT.

$a \sim \mathrm{LLM}(x)$

$\mathrm{score}(x, a) \sim \mathrm{LLM}(x, a)$

$\mathrm{LLM}'(a \mid x) \propto \mathbb{1}\{\mathrm{score}(x, a) \geq \tau\}$

$\mathrm{LLM}'(a \mid x) = \mathrm{LLM}(a \mid x)\,\mathbb{1}\{\mathrm{score}(x, a) \geq \tau\}$

$\mathrm{Acc}_{\mathrm{gen}}$ **Generation-Verification Gap** = $\mathrm{Acc}_{\mathrm{ver}}$ - $\mathrm{Acc}_{\mathrm{ft}}$

$\mathrm{Acc}_{\mathrm{ft}}$ - $\mathrm{Acc}_{\mathrm{gen}}$ is not a very good metric (e.g., format)

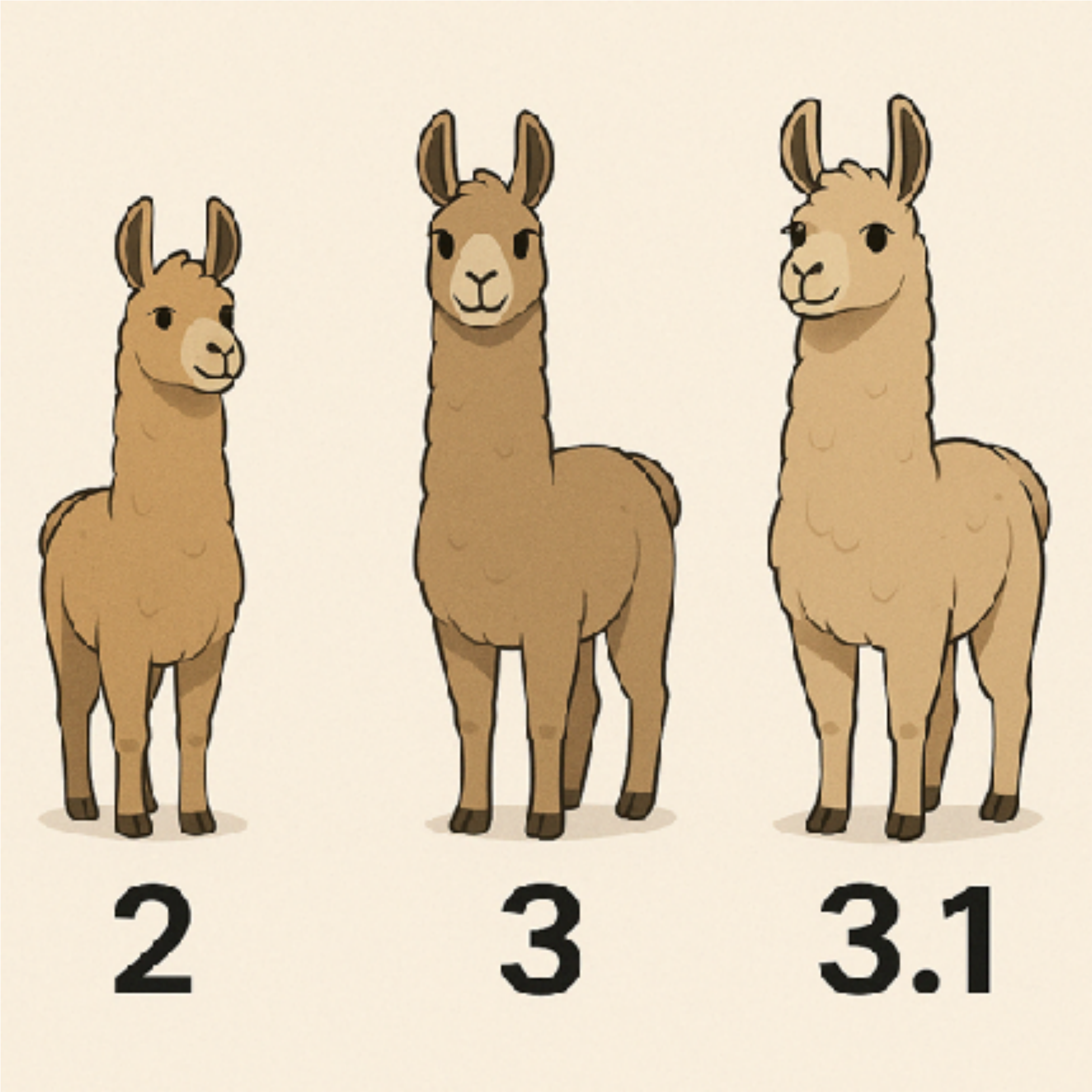
A Large-Scale Scientific Study

- Many models.

Qwen1.5

Qwen2

Qwen2.5

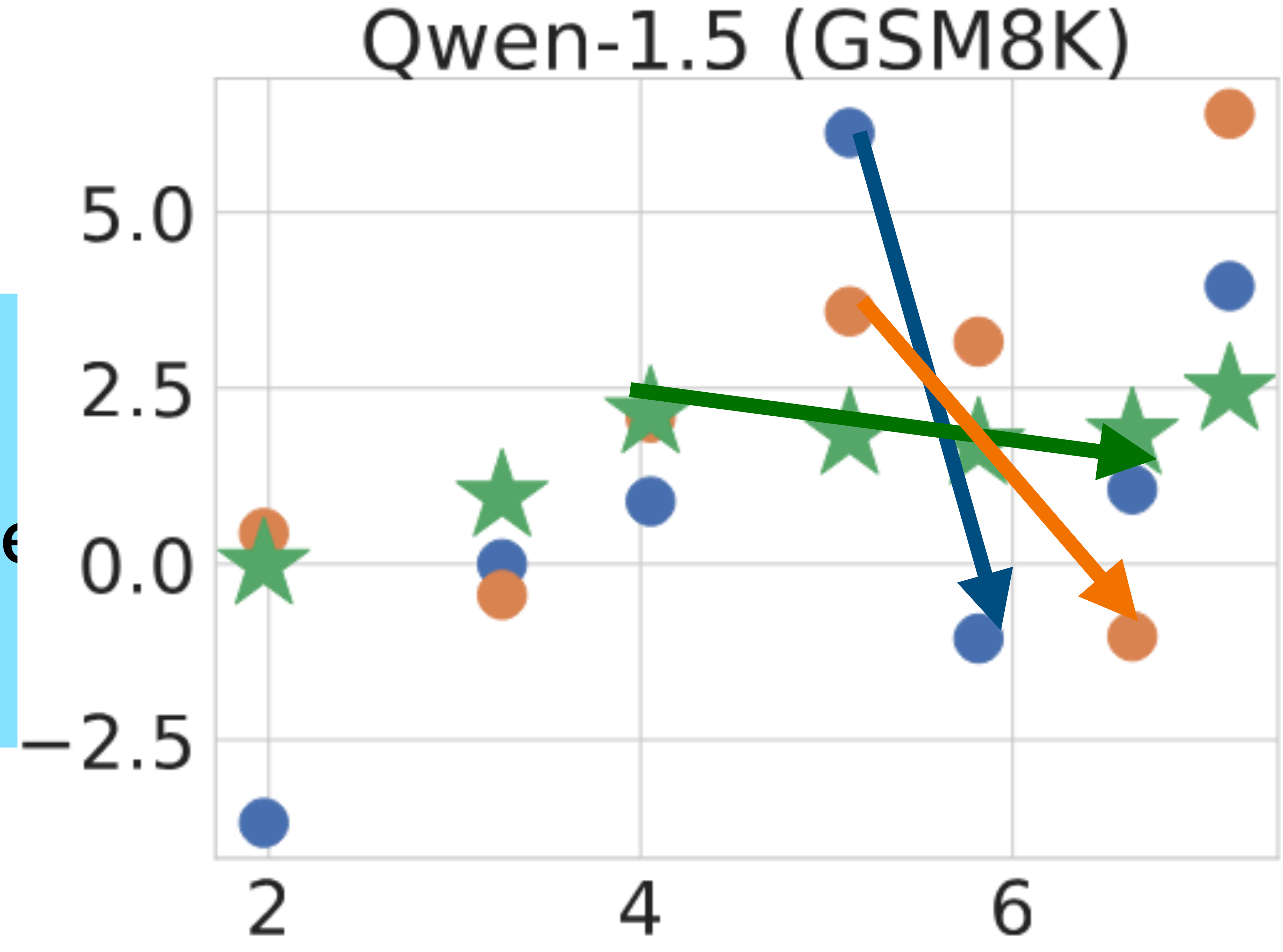


Yi

- Many verification mechanisms.
  - **Logit**: $p(\text{yes}) - p(\text{no})$.
  - **Binary CoT**: “The calculation… Thus the answer is correct/incorrect.”
  - **Score CoT**: “The calculation… Thus the score of the answer is [0-10].”
  - **Tournament**: pairwise comparison and elimination.

Can We Predict Self-Improvement Capability

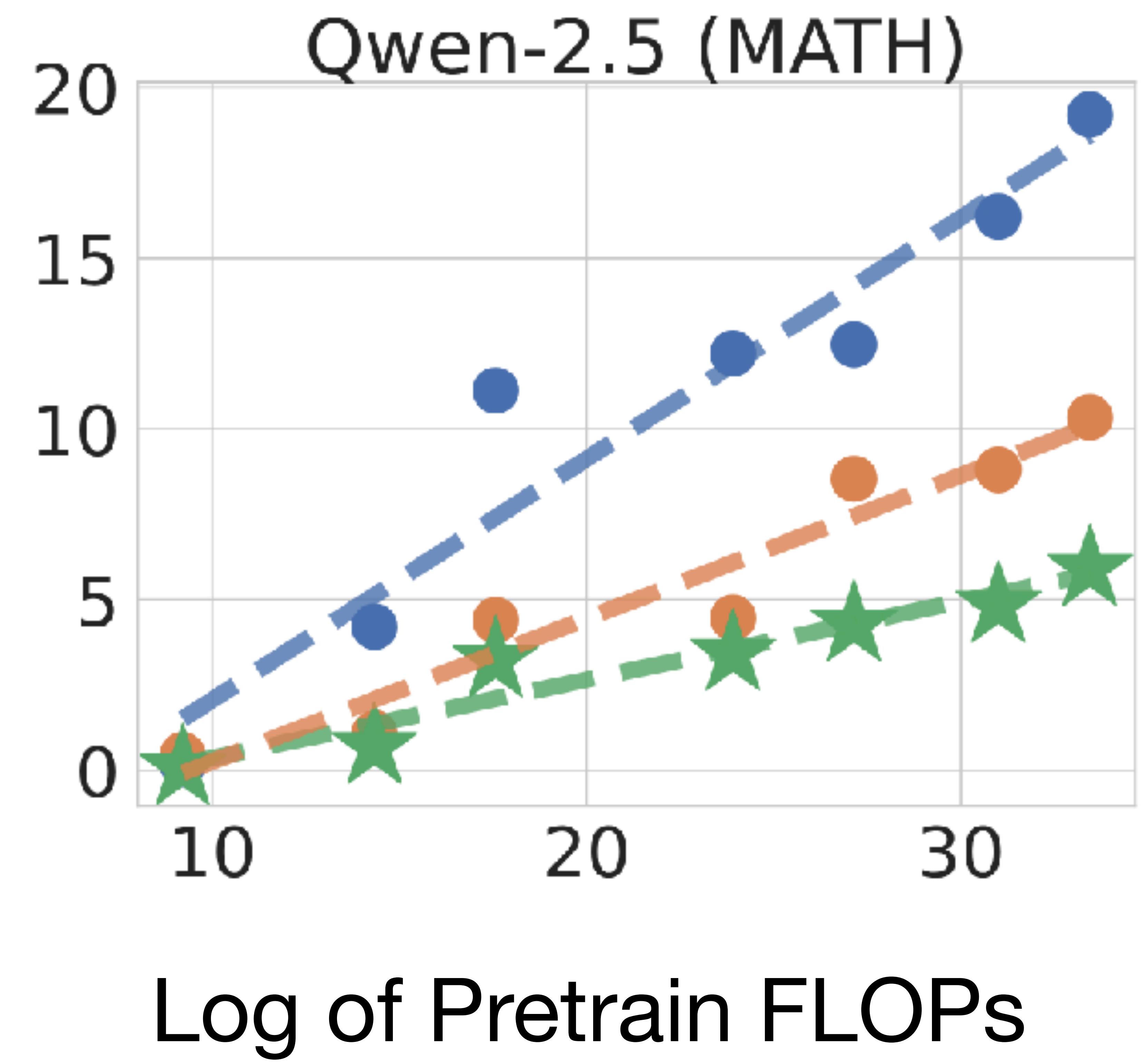
Gap $= \mathrm{Acc}_{\mathrm{ve}}$



Log of Pretrain FLOPs

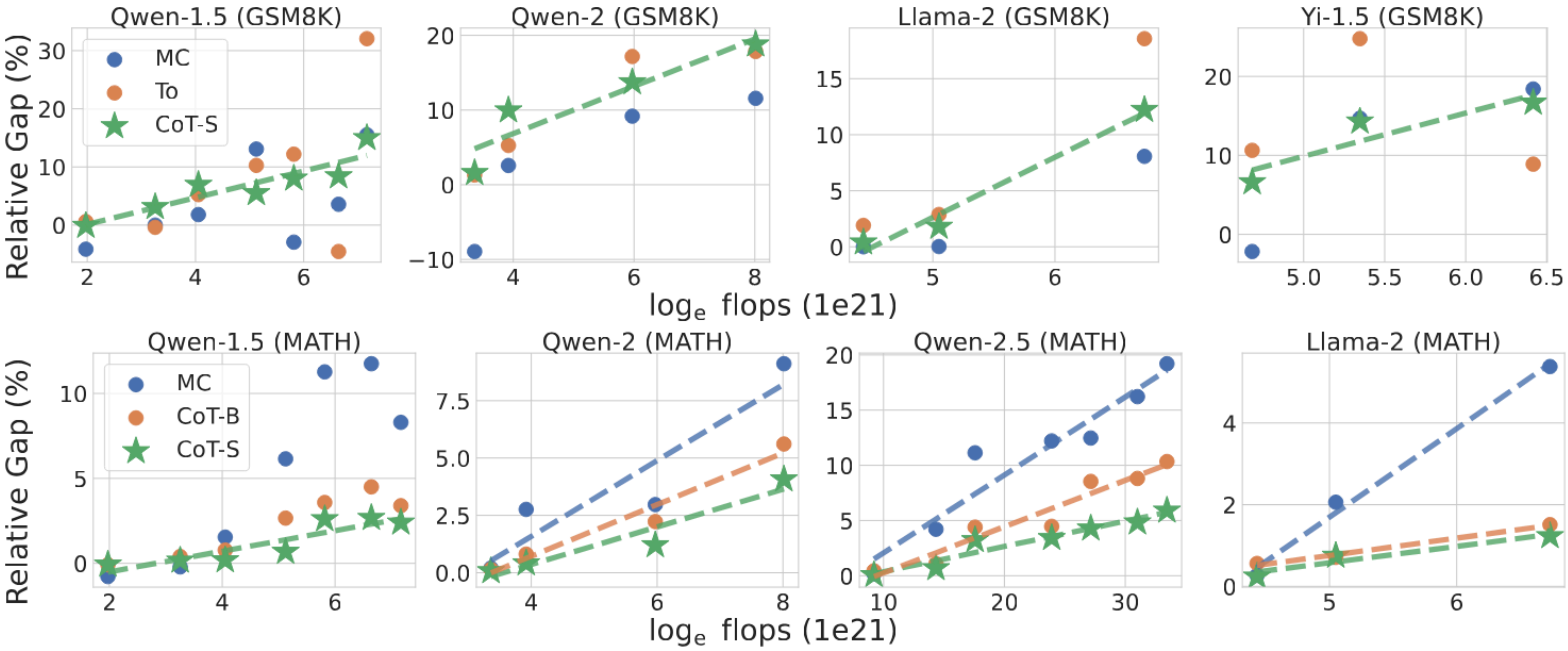
Scaling Law of Relative Gap

Relative Gap $\approx \frac{\quad}{1 -}$

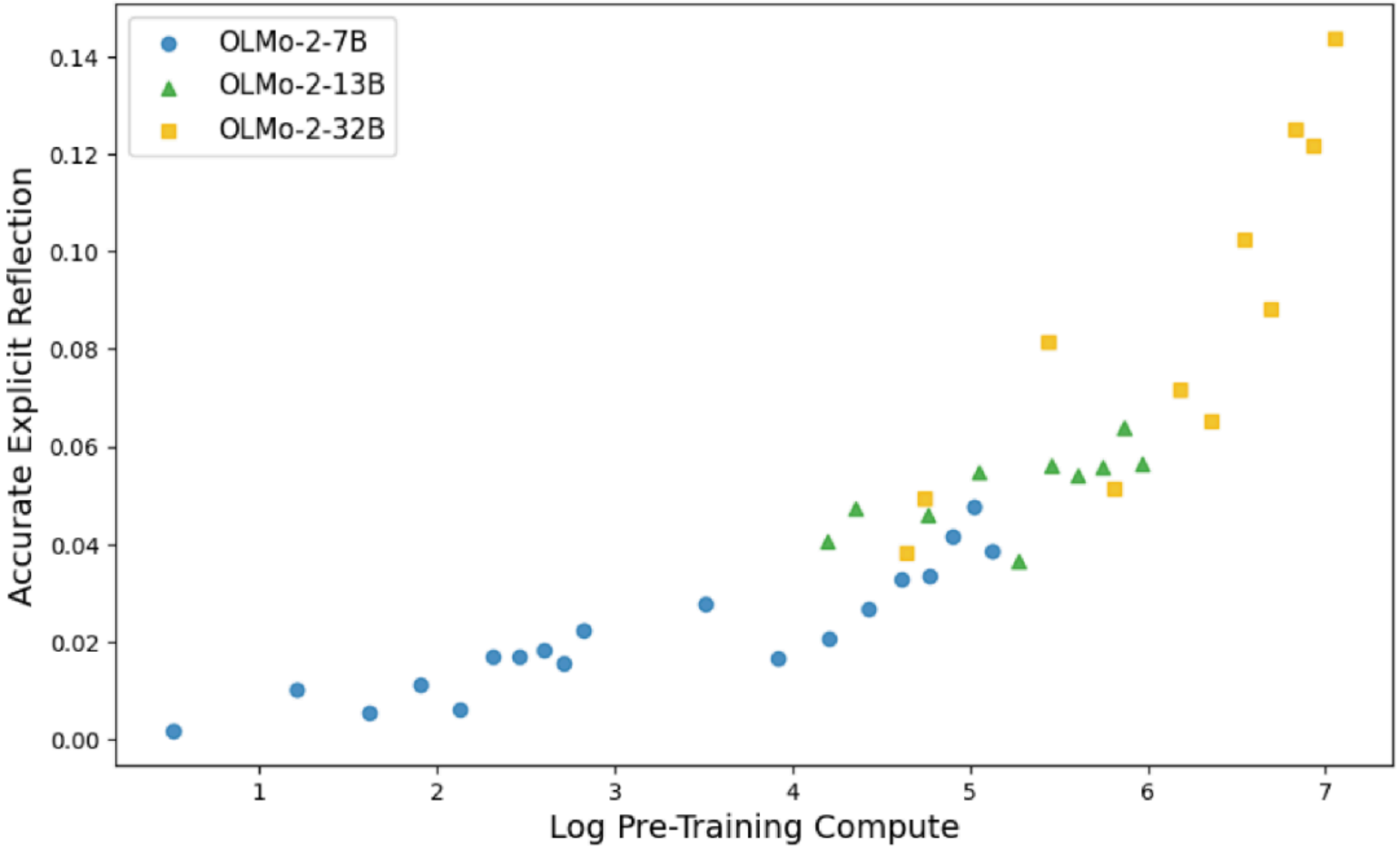


“Conditioned on being wrong”

Scaling Law of Relative Gap

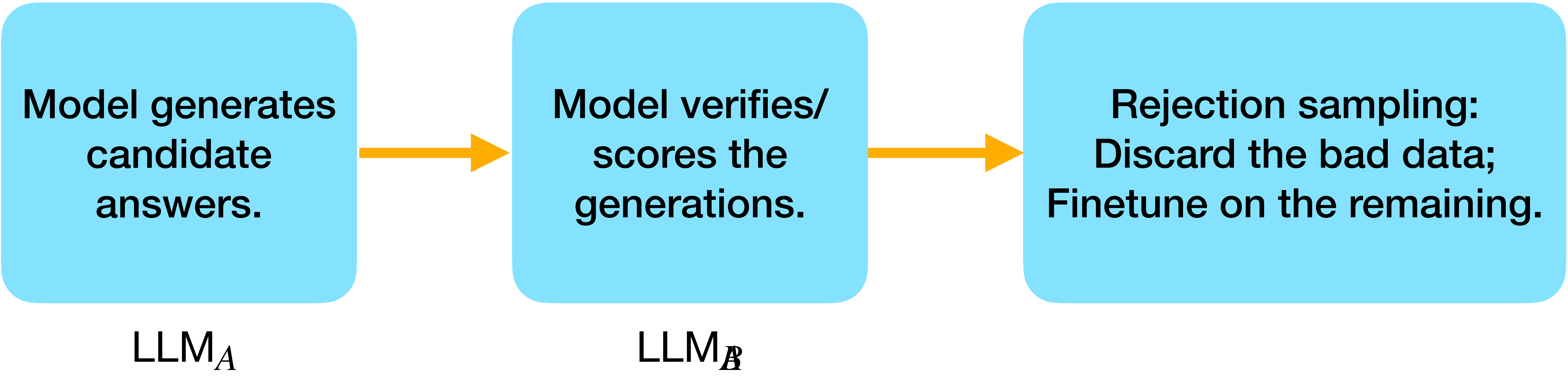


Relative Gap (continued)



Essential AI. "Rethinking Reflection in Pre-Training."

Test-time Improvement

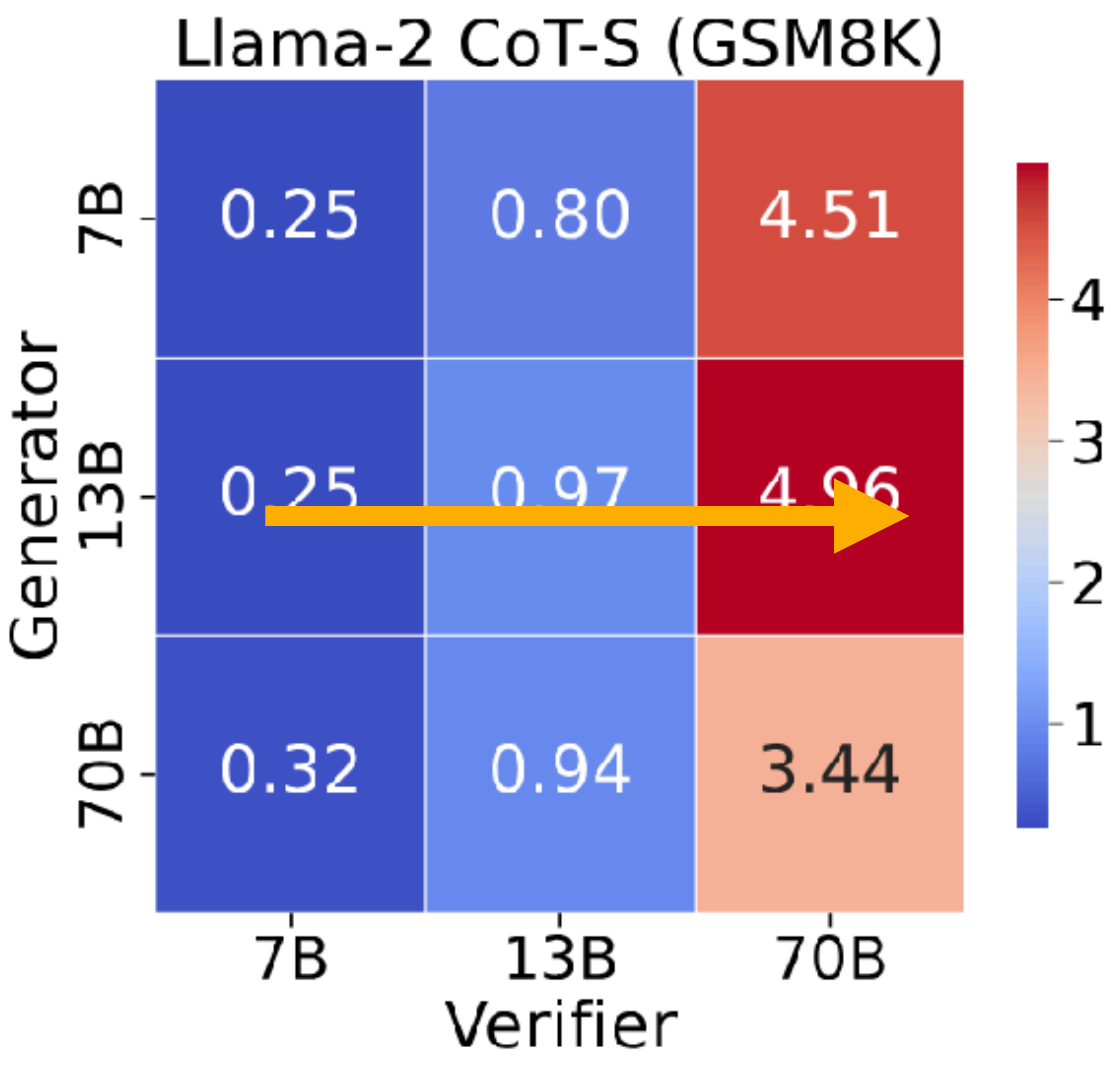


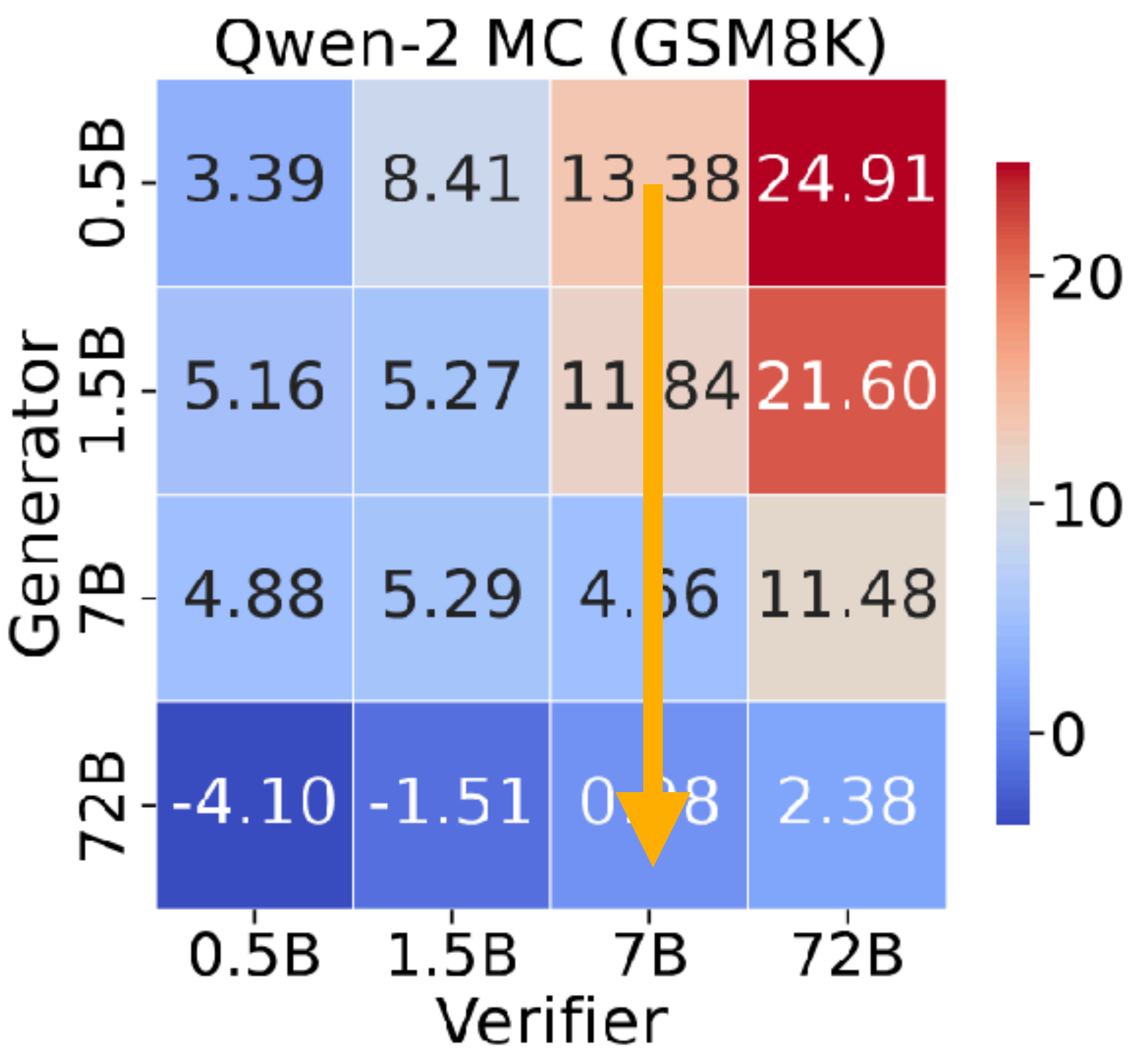
Model generates candidate answers.

Model verifies/ scores the generations.

Rejection sampling: Discard the bad data; Finetune on the remaining.

Cross Verification

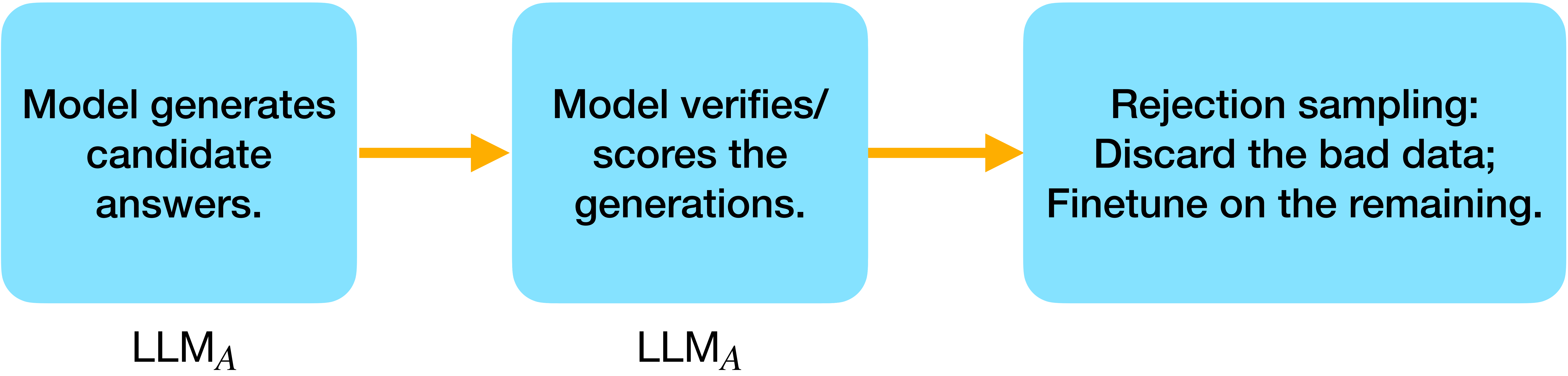




Weak ↓ Strong

Teacher → Student

Test-time Improvement



Model generates candidate answers.

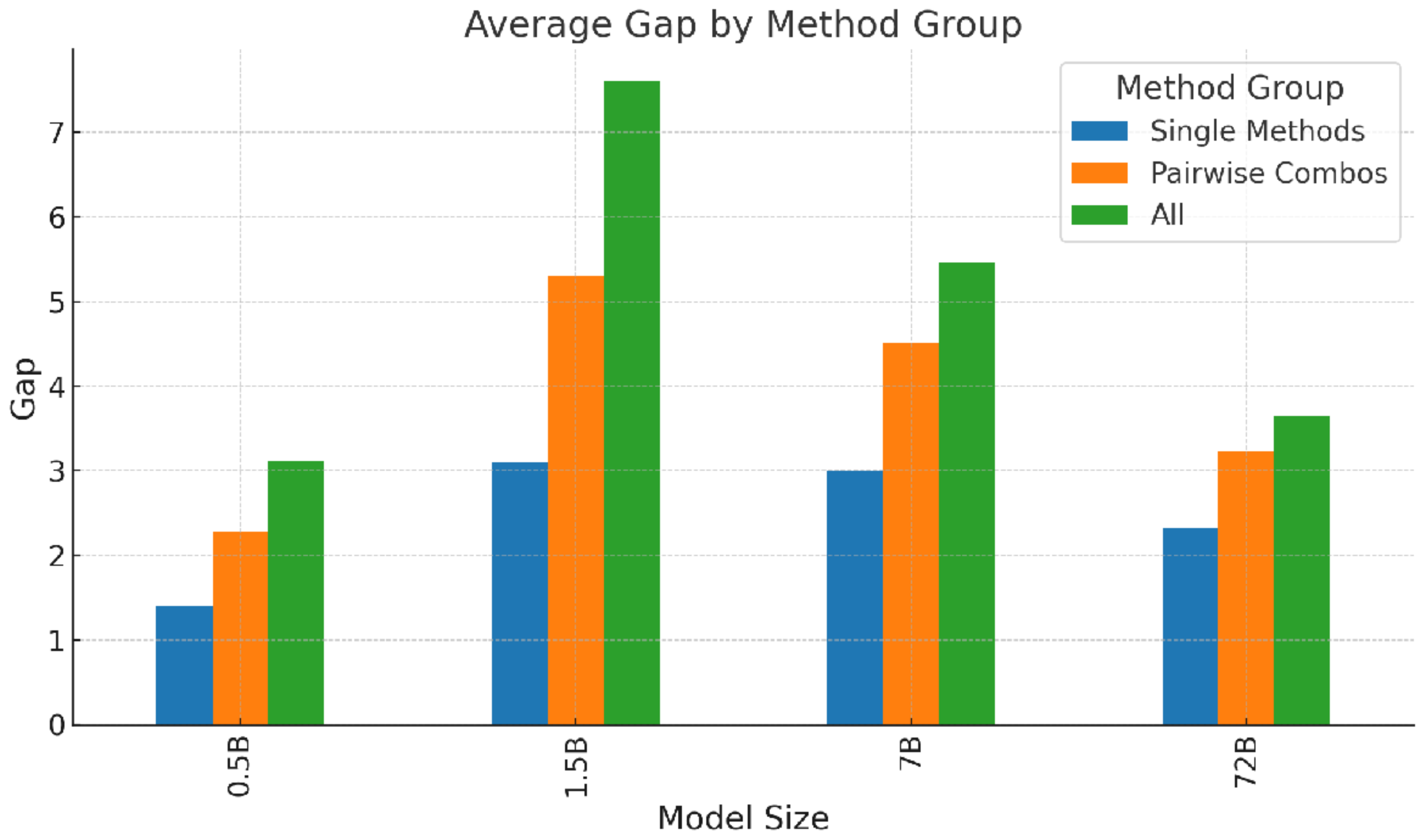
LLM$_A$

Model verifies/ scores the generations.

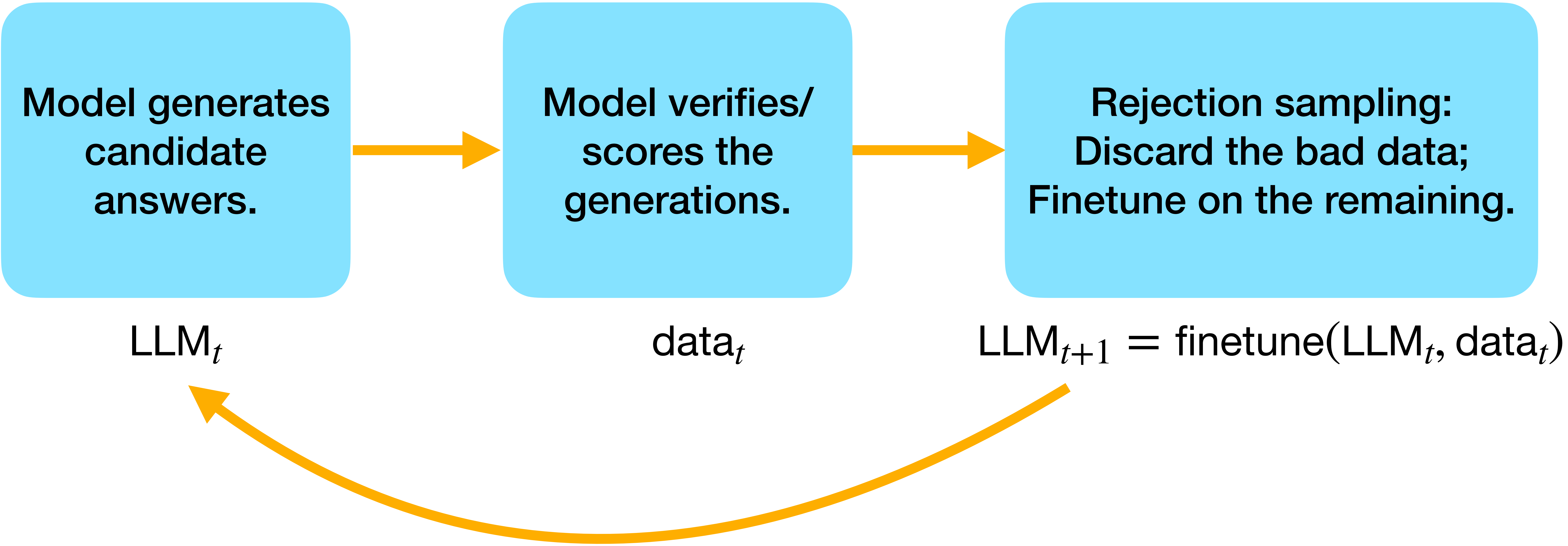
LLM$_A$

Rejection sampling: Discard the bad data; Finetune on the remaining.

Verification Ensemble



Iterative Self-Improvement



Model generates candidate answers.

Model verifies/ scores the generations.

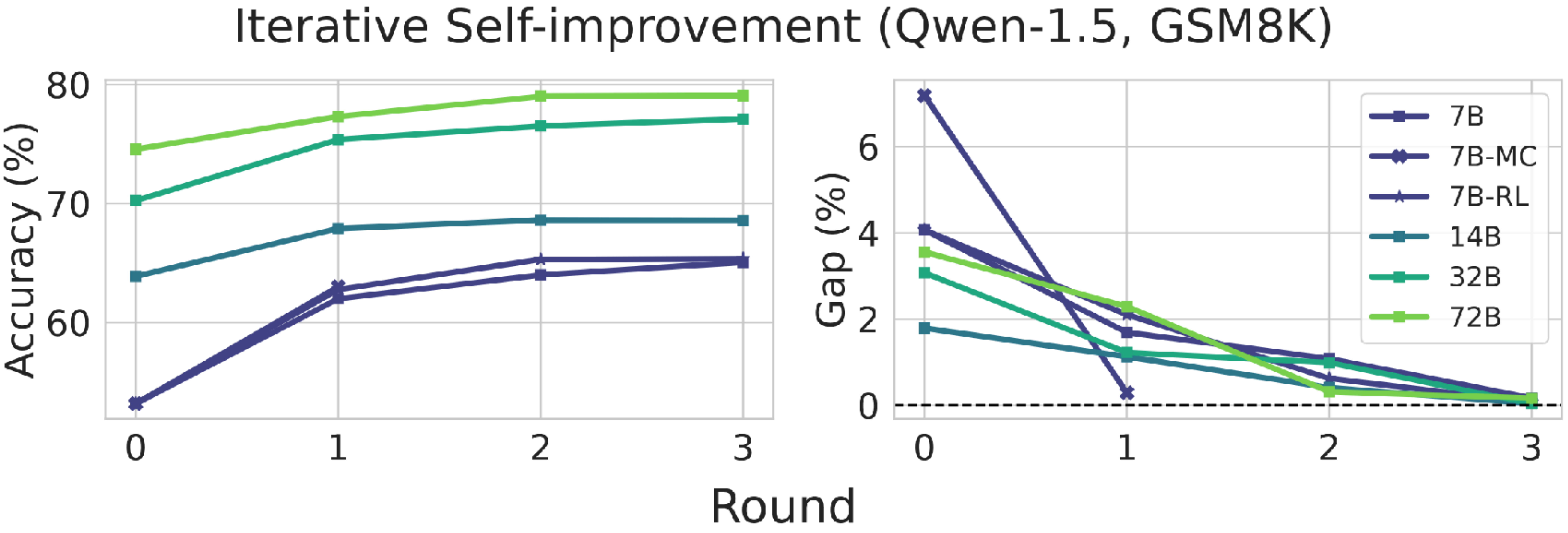
Rejection sampling: Discard the bad data; Finetune on the remaining.

LLM_t

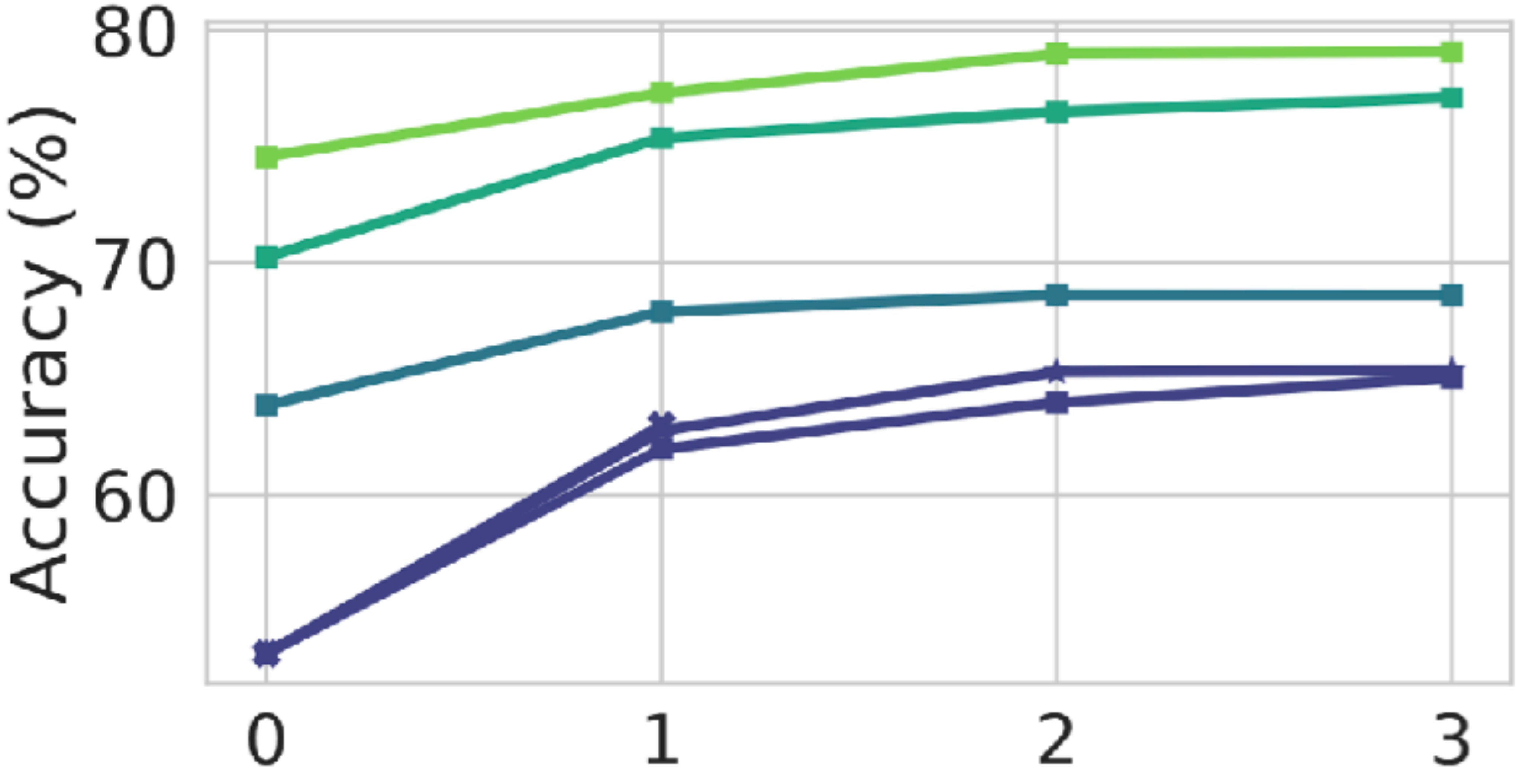
data_t

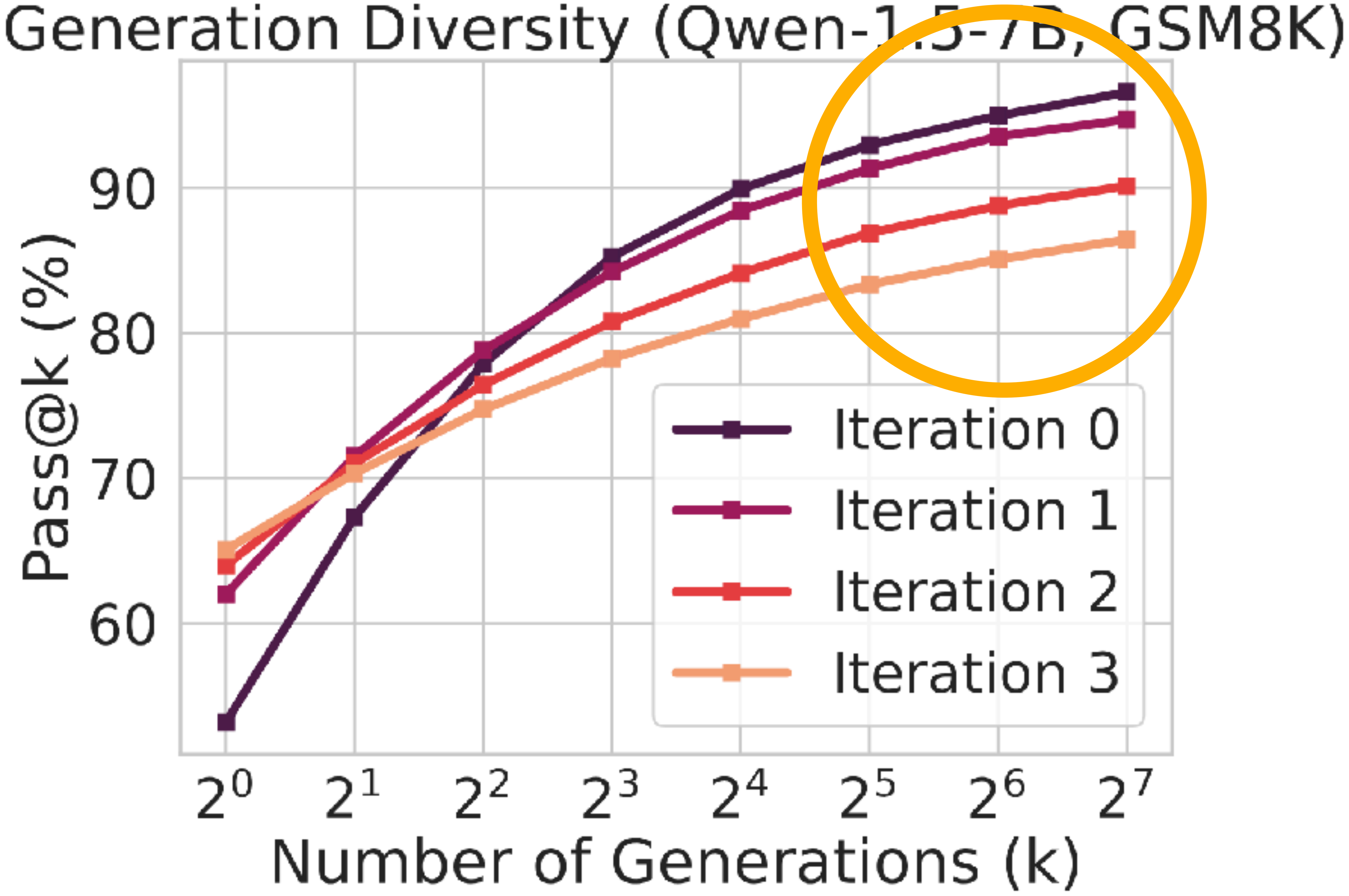
$\text{LLM}_{t+1} = \text{finetune}(\text{LLM}_t, \text{data}_t)$

Iterative Self-Improvement

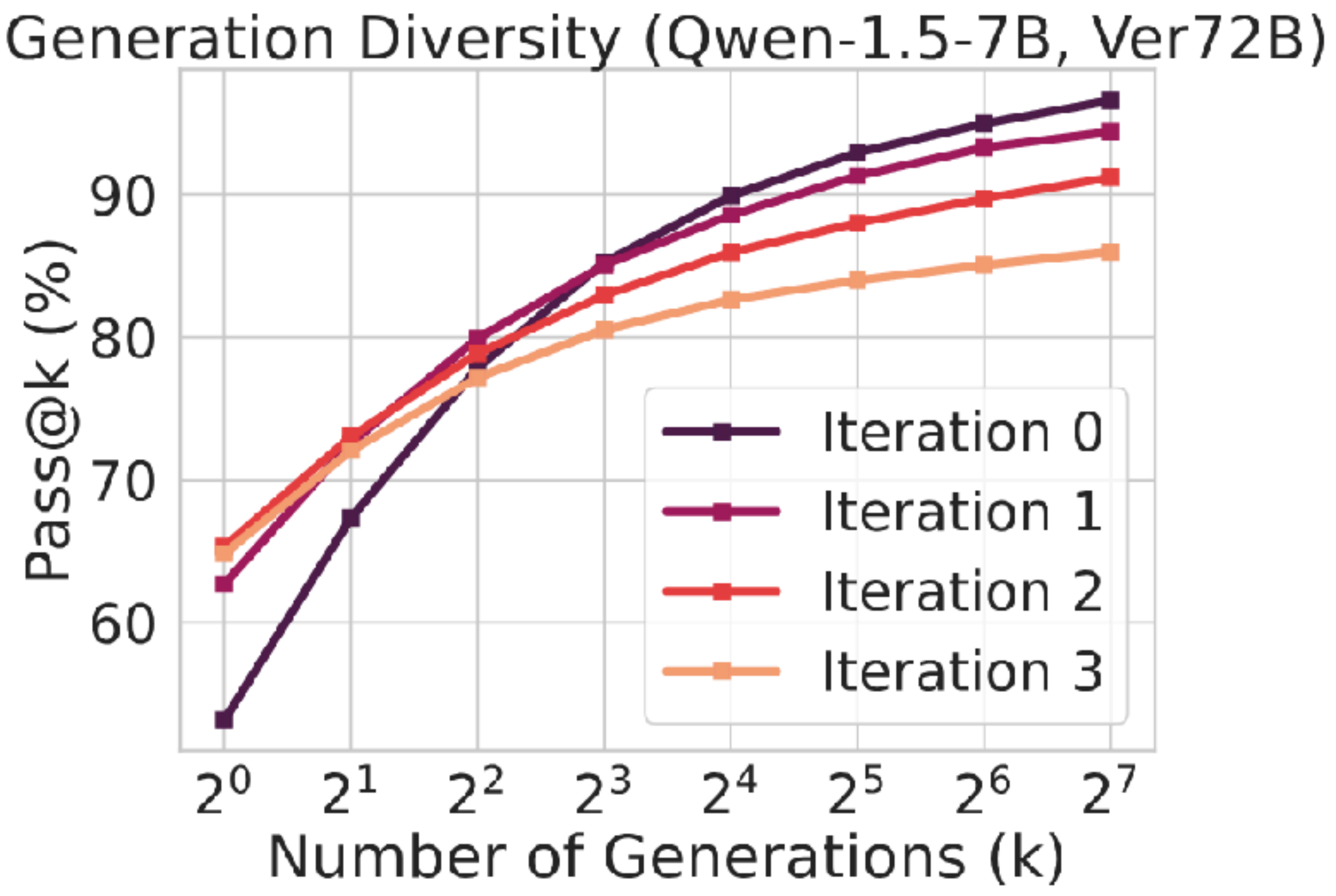


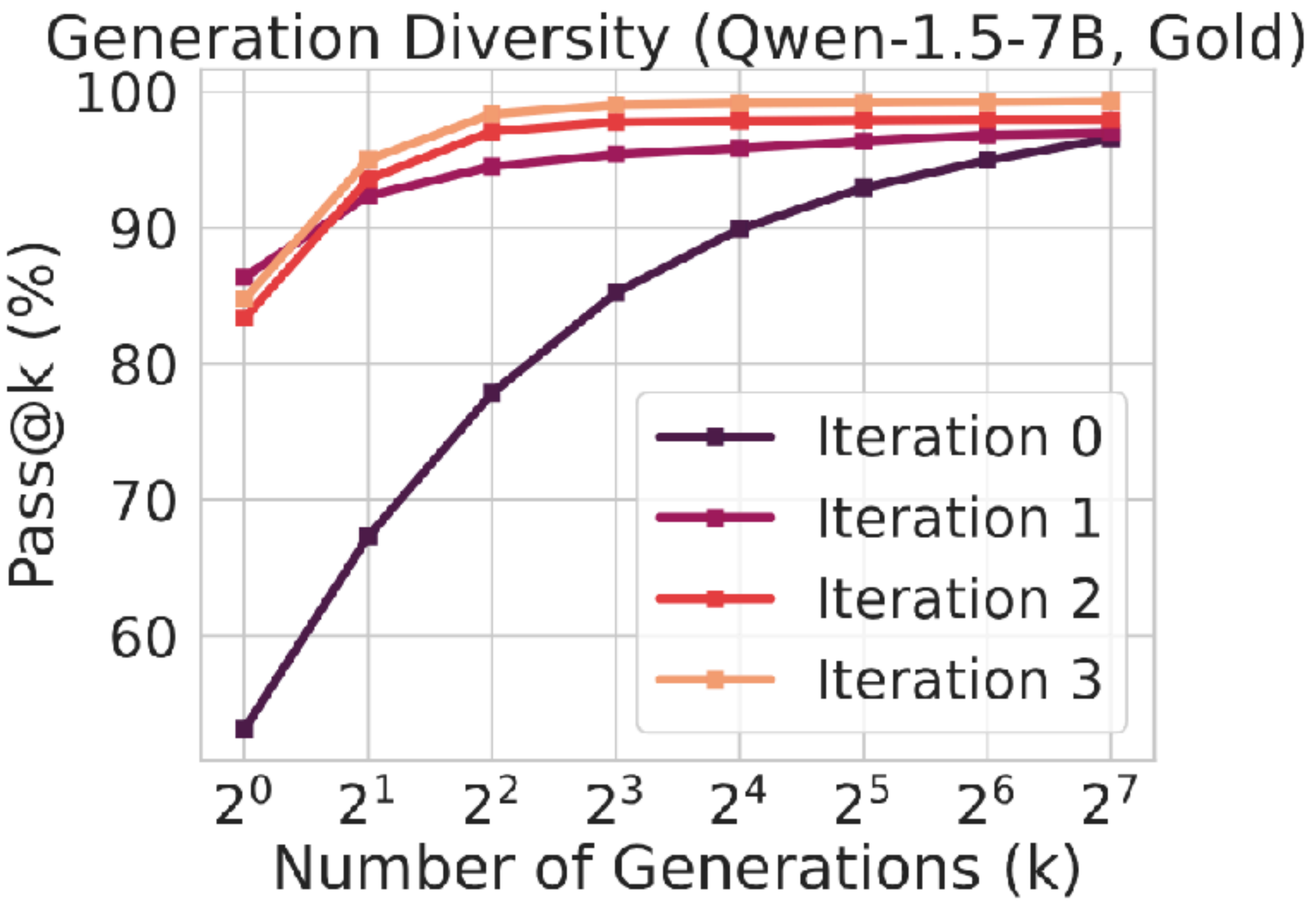
Effective Diversity Collapse



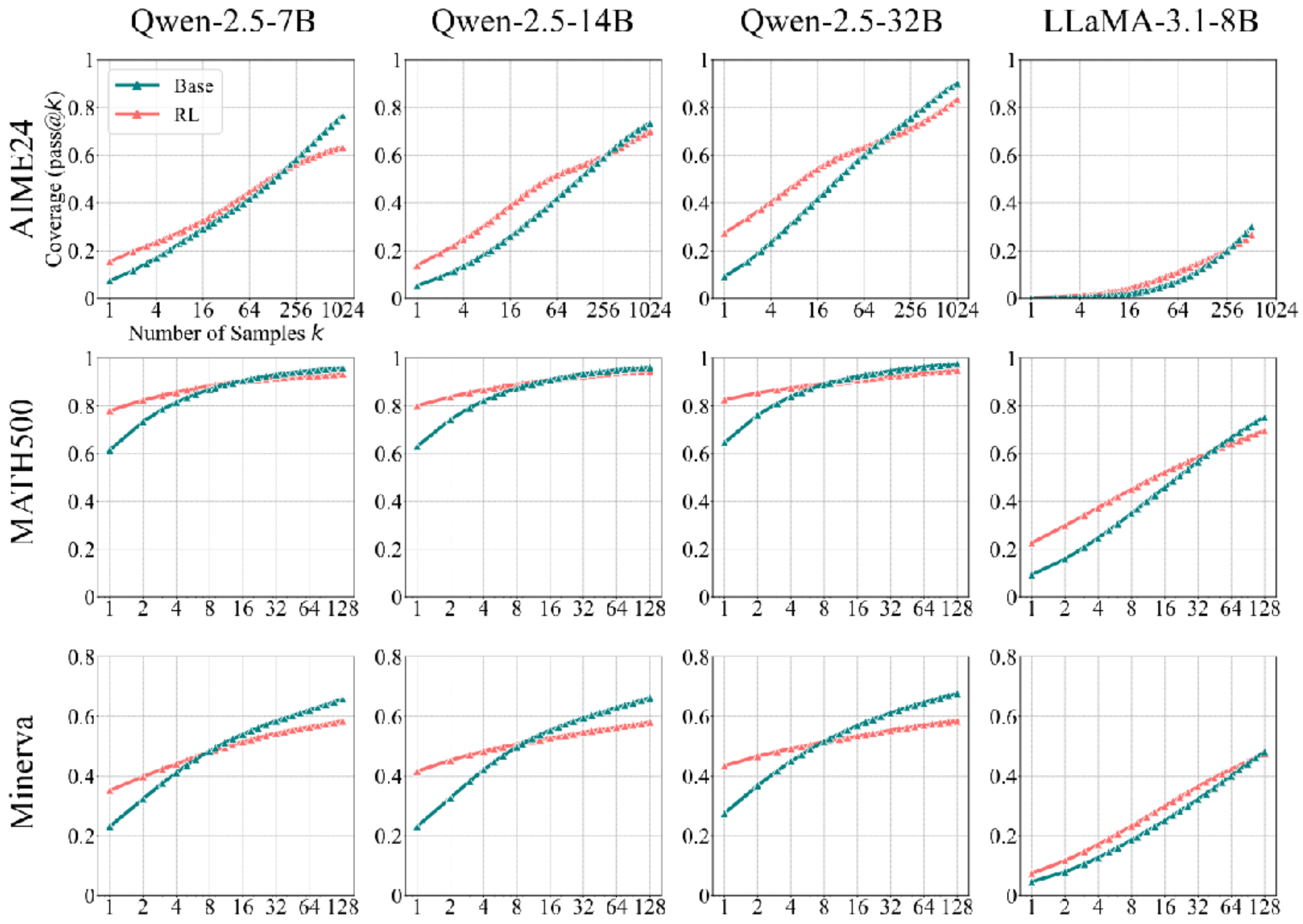


Effective Diversity Collapse





Effective Diversity Collapse (cont.)

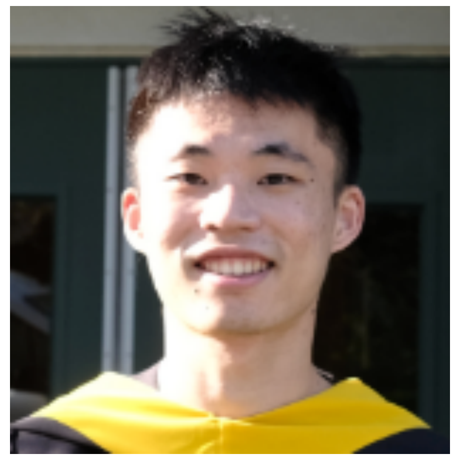


Yue et al. "Does Reinforcement Learning Really Incentivize Reasoning Capacity in LLMs Beyond the Base Model?"

1. How do we get feedback when the reward is unreliable?
2. How do we get additional signal when there is a reliable reward?
3. How do we efficiently query feedback (with help from synthetic feedback) when it is expensive?

Outcome-Based Exploration for LLM Reasoning

In submission



Yuda Song, Julia Kempe, Rémi Munos

RL as Sampling

Previous studies:

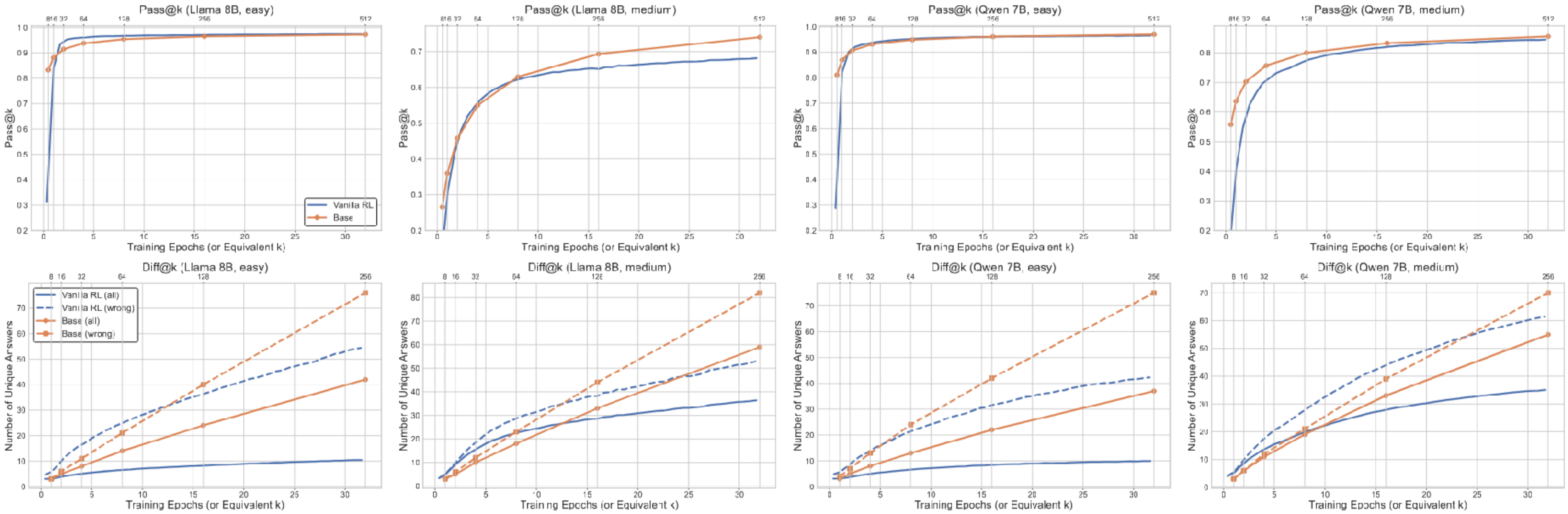
- Test performance
- Single checkpoint

Then how do we understand the **training dynamics** -> algorithmic intervention?

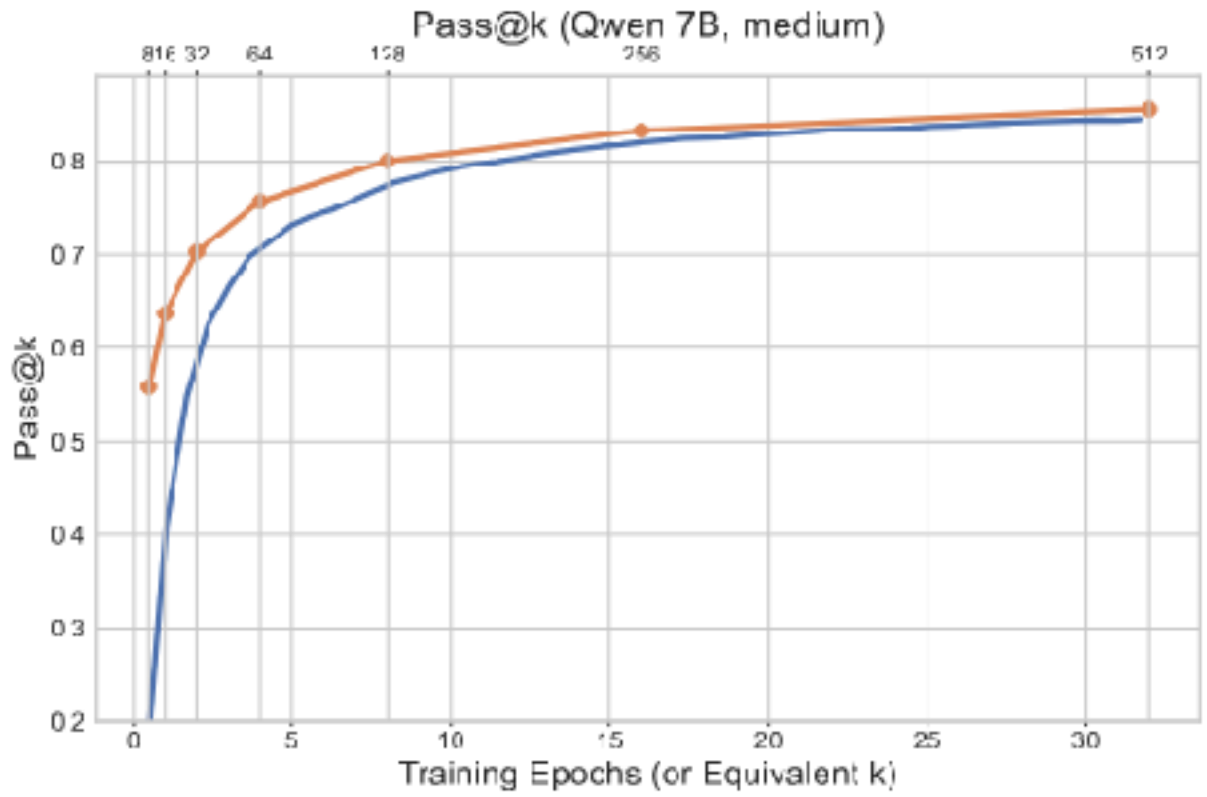
- Online RL: sample then update
- We sample each question x once per epoch
- Then we sample k generations per question
- $y_1, \cdots, y_{k \times N} \sim \mathrm{Alg}(x, N)$

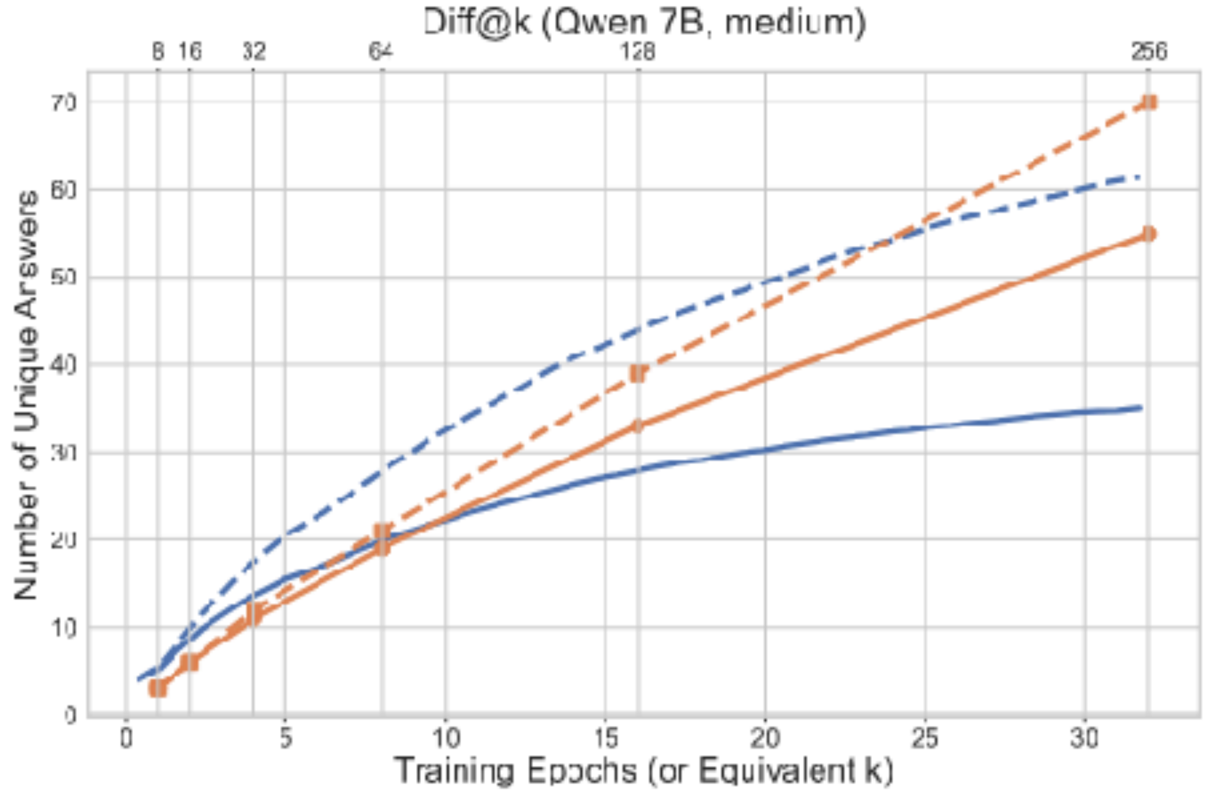
Diversity Collapse during Training

Takeaway #1: RL eventually solves fewer questions than the base model.



Diversity Collapse during Training





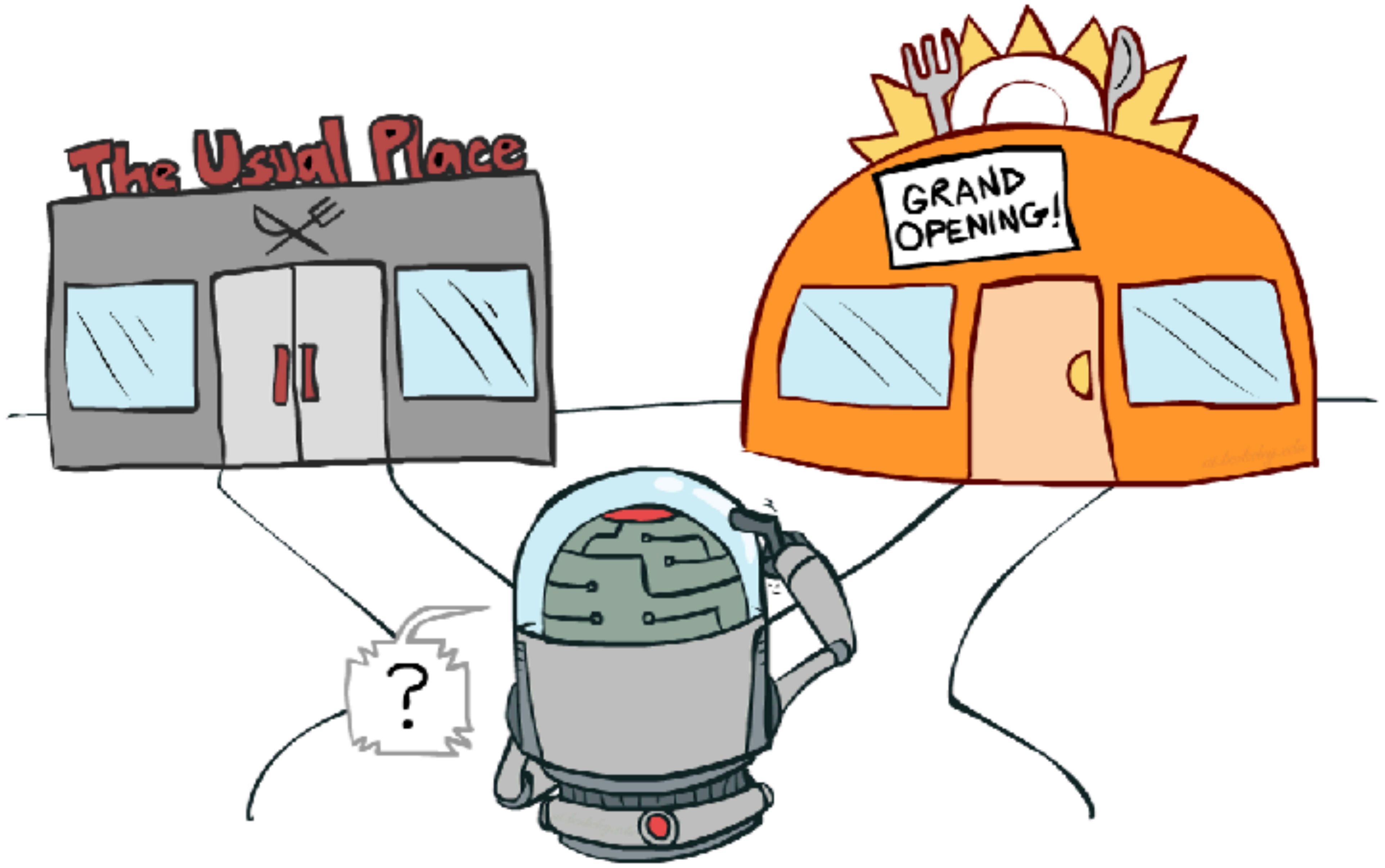
Thought Experiment:

- Imagine prompts are independent
- RL should do no worse on pass@k
- RL should do no worse on diff@k on wrong questions either

Takeaway #2: Transfer of diversity degradation across questions.

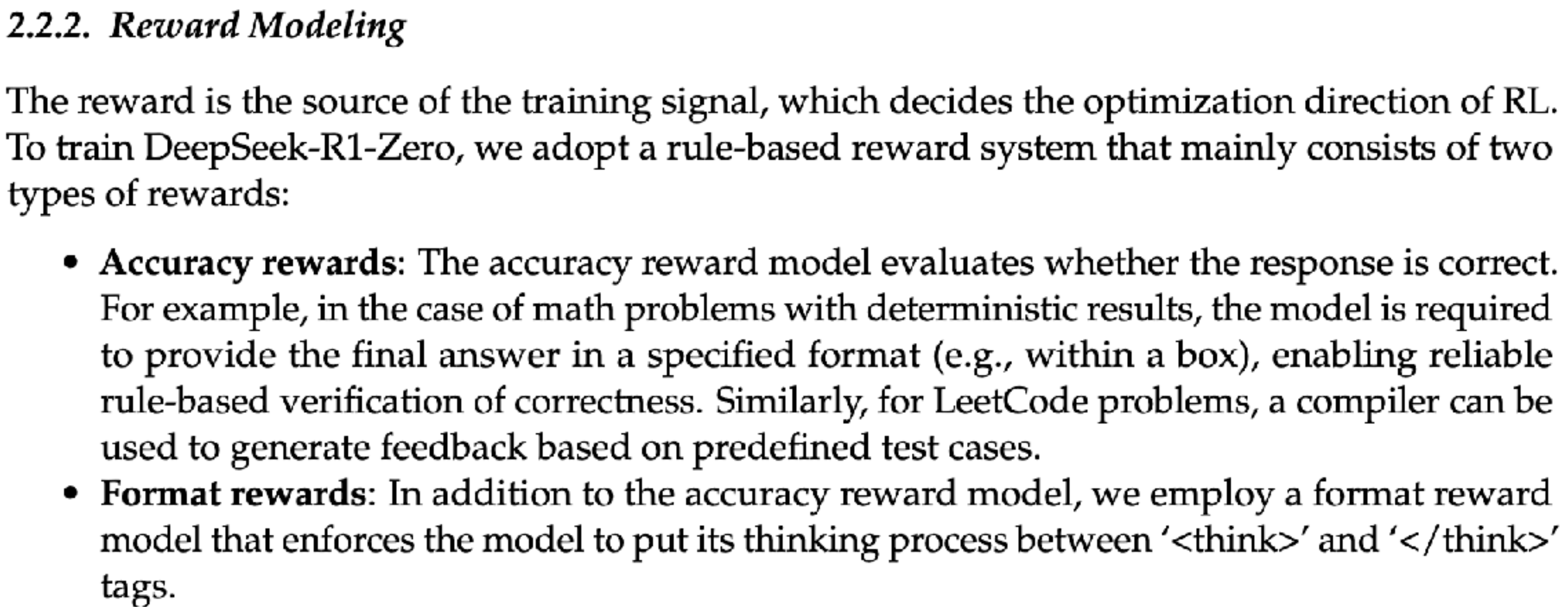
Mitigating Diversity Collapse - Exploration

- Idea: encourage the policy to visit as many states as possible
- In classical RL, this is usually tractable:
  - Finite number of states
  - Easy to cluster states
- In LLM:
  - Exponentially many token permutations



Outcome-Based RL

Takeaway #3: Diversity is tractable on verifiable domains.

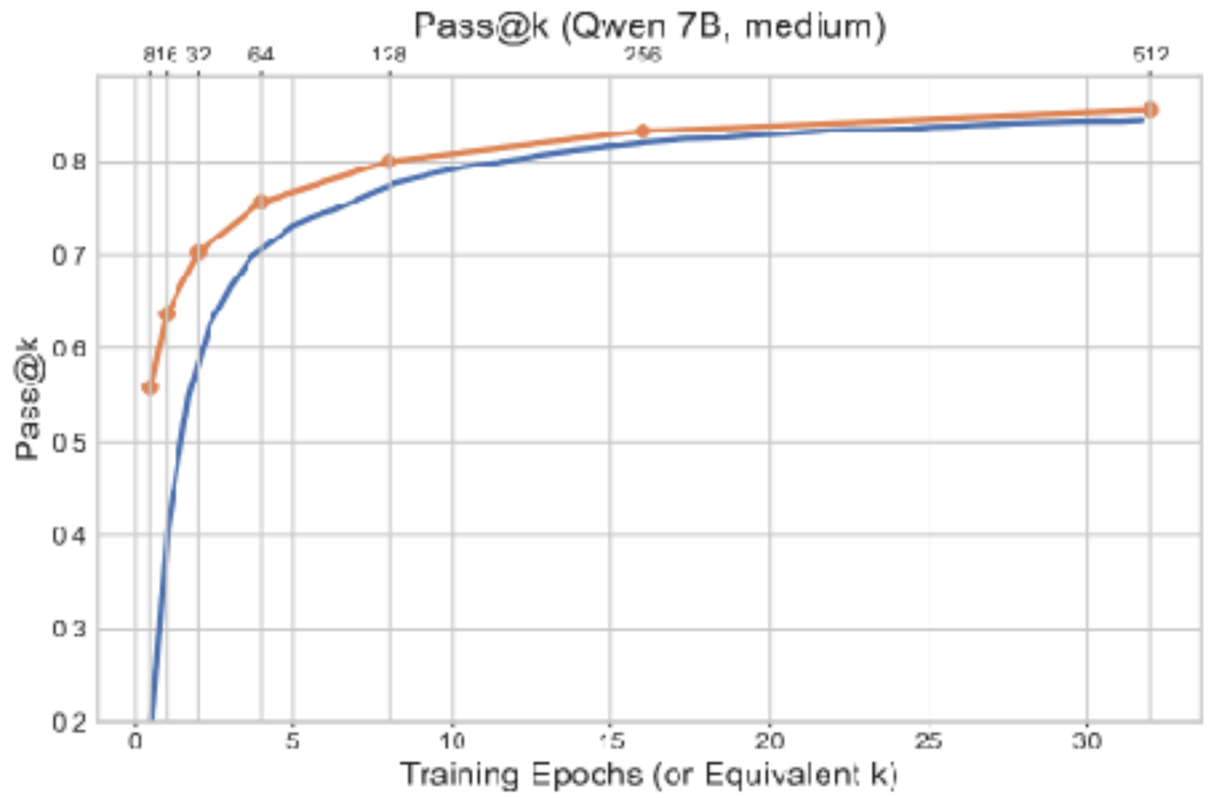


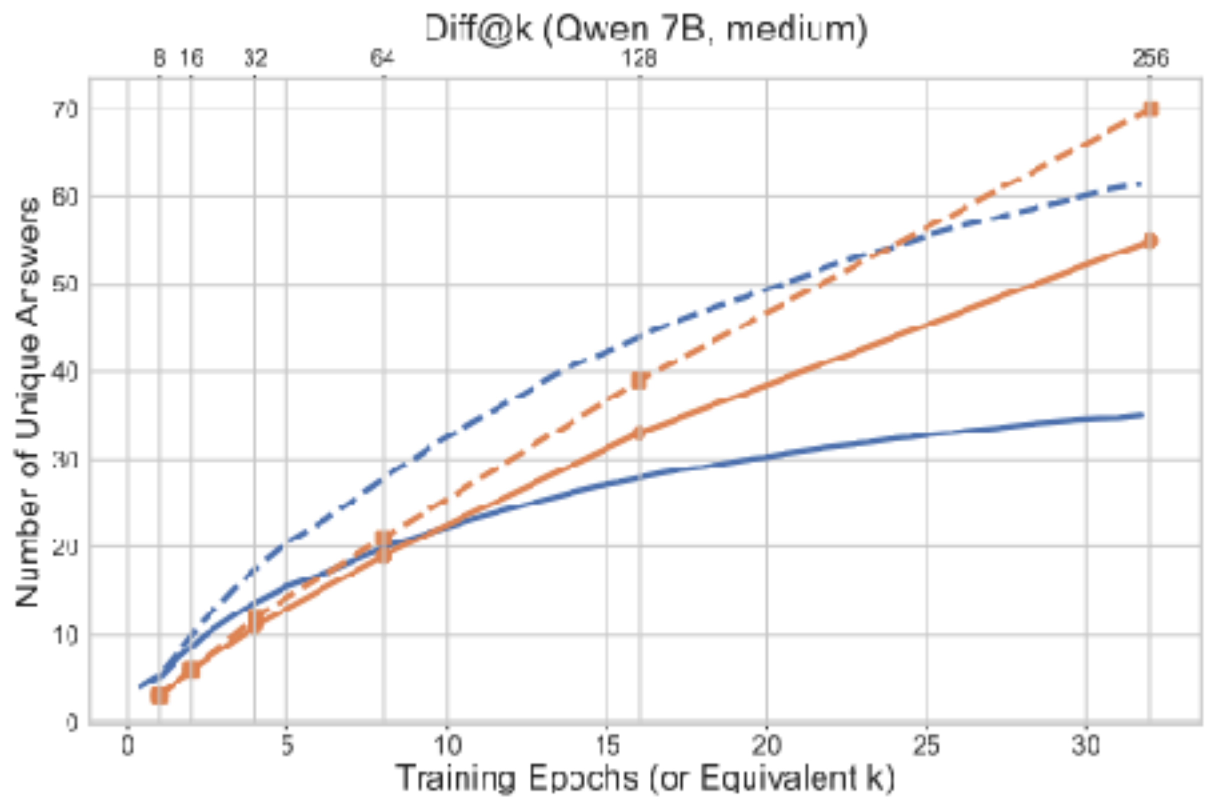
2.2.2. *Reward Modeling*

The reward is the source of the training signal, which decides the optimization direction of RL. To train DeepSeek-R1-Zero, we adopt a rule-based reward system that mainly consists of two types of rewards:

- **Accuracy rewards**: The accuracy reward model evaluates whether the response is correct. For example, in the case of math problems with deterministic results, the model is required to provide the final answer in a specified format (e.g., within a box), enabling reliable rule-based verification of correctness. Similarly, for LeetCode problems, a compiler can be used to generate feedback based on predefined test cases.
- **Format rewards**: In addition to the accuracy reward model, we employ a format reward model that enforces the model to put its thinking process between '<think>' and '</think>' tags.

$$R(x, y, a) = \mathbb{1}\{a = a_x^{\text{gold}}\}$$





Deepseek AI. "DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning"

Outcome-Based Exploration (OBE)

- In additional to the reward feedback, an exploration bonus that is inversely proportional to historical number of visits:

$$\text{bonus}(x, a) = \min\left\{1, \sqrt{\frac{1}{N(x, a)}}\right\}.$$

- Objective:

$$\widehat{\mathbb{E}}_{x,\{y_\kappa,a_\kappa\}_{\kappa=1}^k \sim \pi(\cdot|x)}\left[\frac{1}{k}\sum_{\kappa=1}^{k} \widehat{A}\left(x, \{y_{\kappa'}, a_{\kappa'}\}_{\kappa'=1}^{n}\right)_\kappa + c \cdot \text{bonus}(x, a_\kappa) - \beta\,\widehat{\text{KL}}\left(\pi(\cdot \mid x), \pi_{\text{base}}(\cdot \mid x)\right)\right]$$

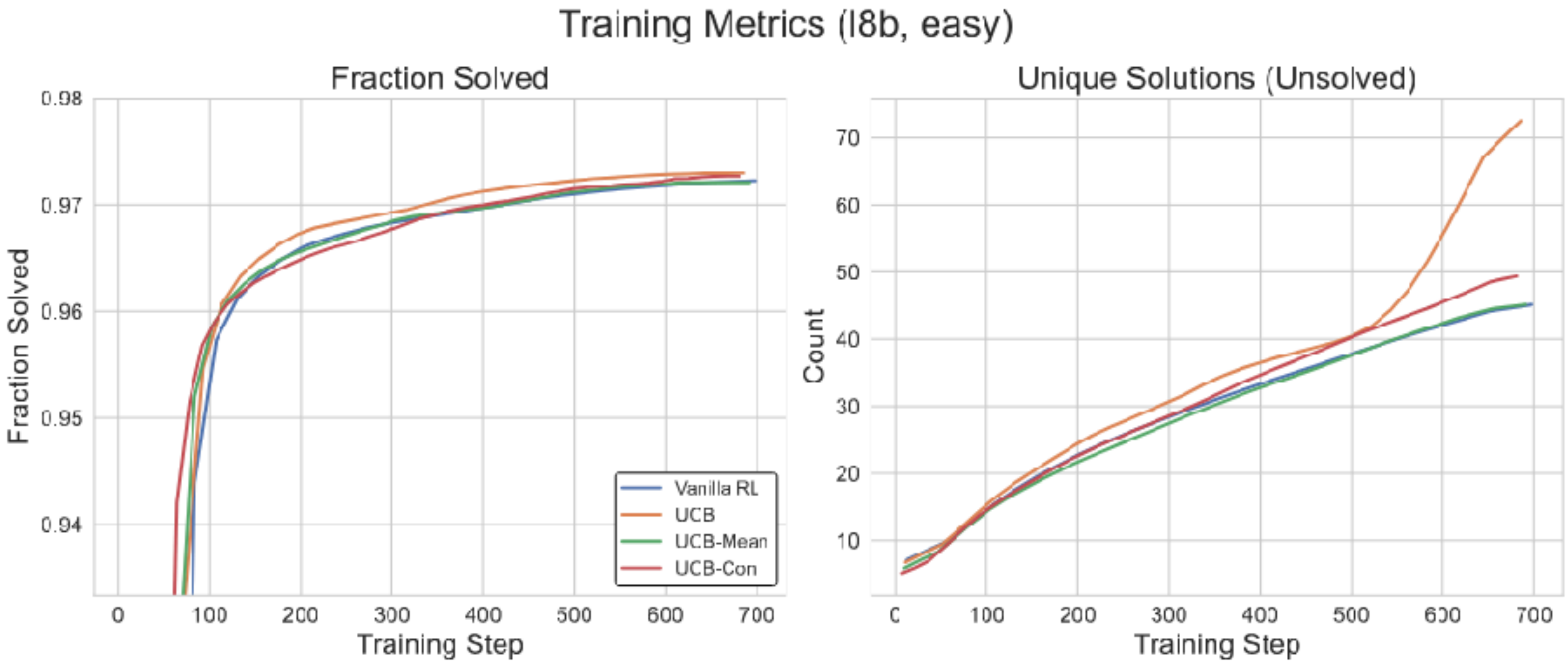
Why Bonus?

- The reason to use exploration bonus is to counter our estimation error
- But LLM reasoning is deterministic
- If we are greedy, we should not visit the same wrong answer twice
- Again prompts are not independent
- But still, should we keep rewarding wrong answers indefinitely?

Baselines

- In fact, naive OBE does not generalize well in test time
- Adding a baseline to tradeoff positive and negative bonus signal:
    - OBE-Mean: $B_{\text{mean}}(x, \{a_i\}_{\kappa=1}^{k})_i = \text{bonus}(x, a_i) - \frac{1}{k} \sum_{\kappa' \neq i}^{k} \text{bonus}(x, a_{\kappa'})$
    - OBE-Constant: $B_{\text{const}}(x, \{a_\kappa\}_{i=\kappa}^{k})_i = \text{bonus}(x, a_i) - b_0$

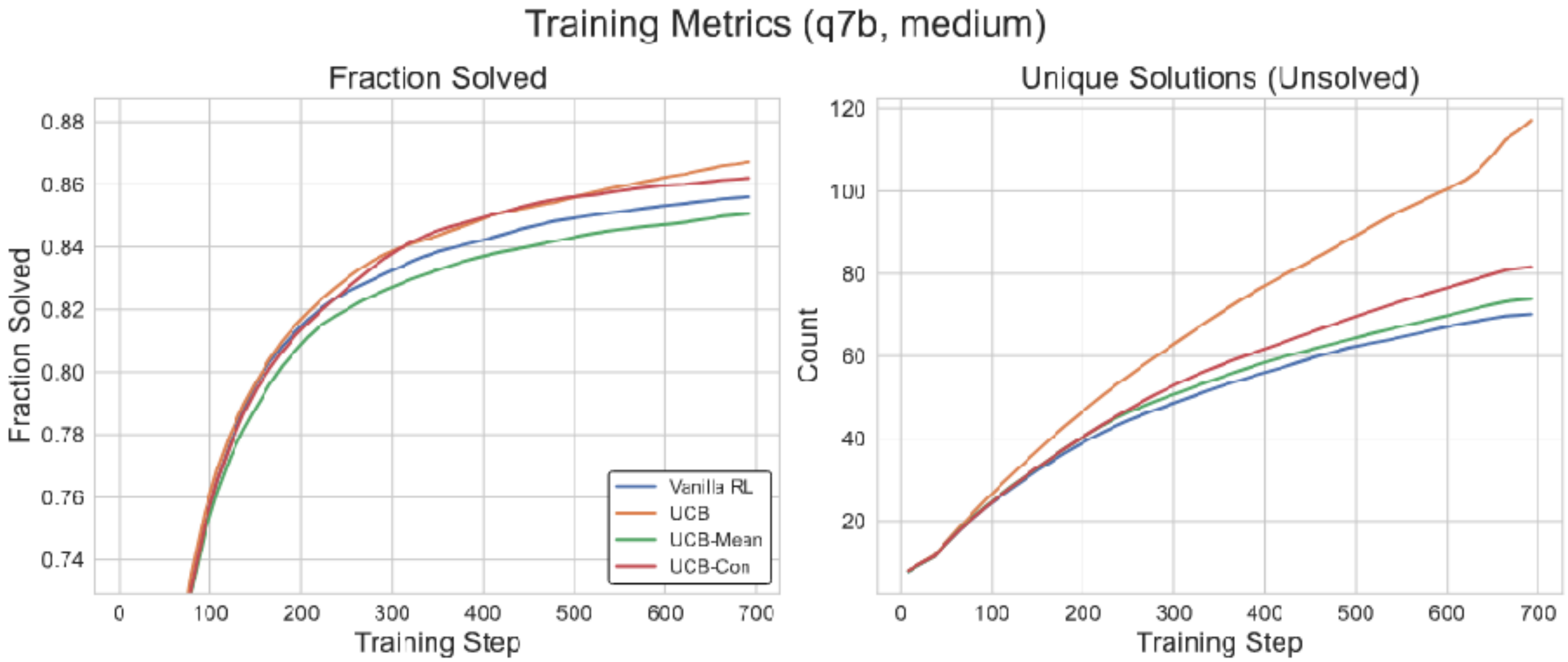
Training Results



Training Metrics (l8b, easy)

Fraction Solved

Unique Solutions (Unsolved)



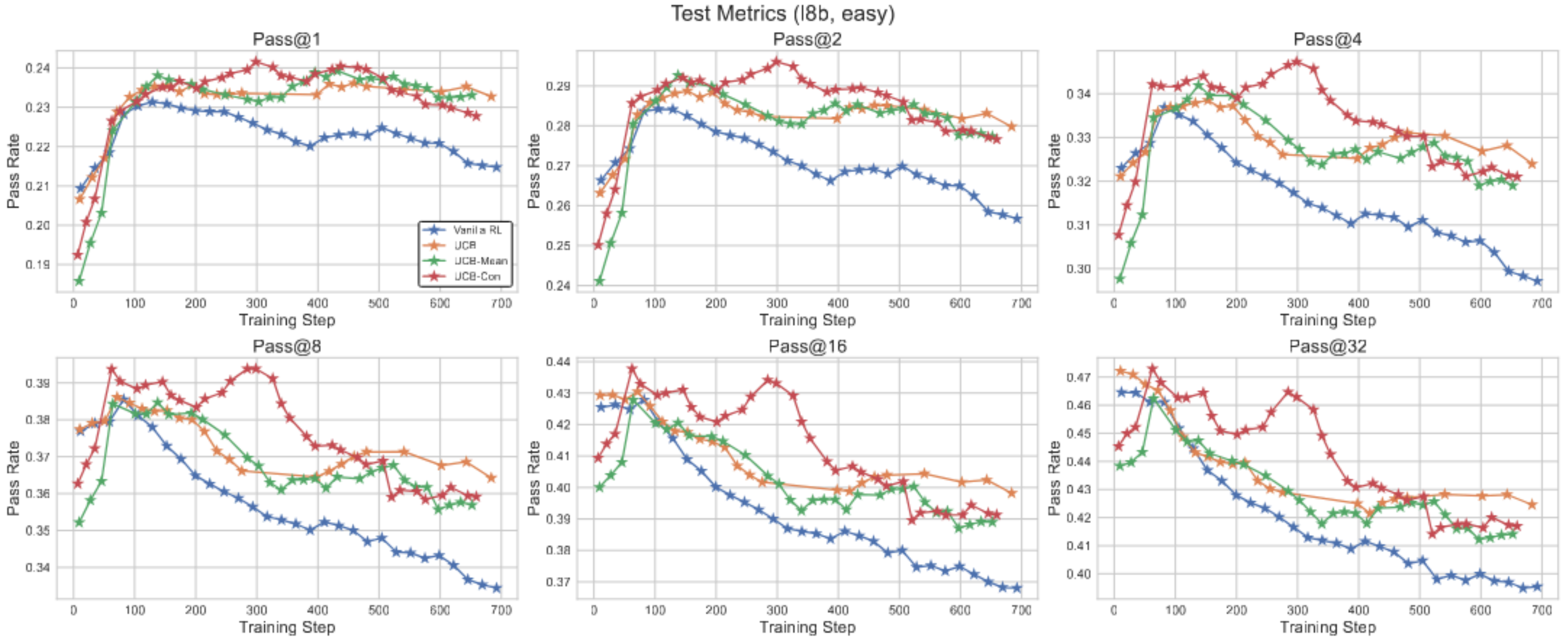
Training Metrics (q7b, medium)

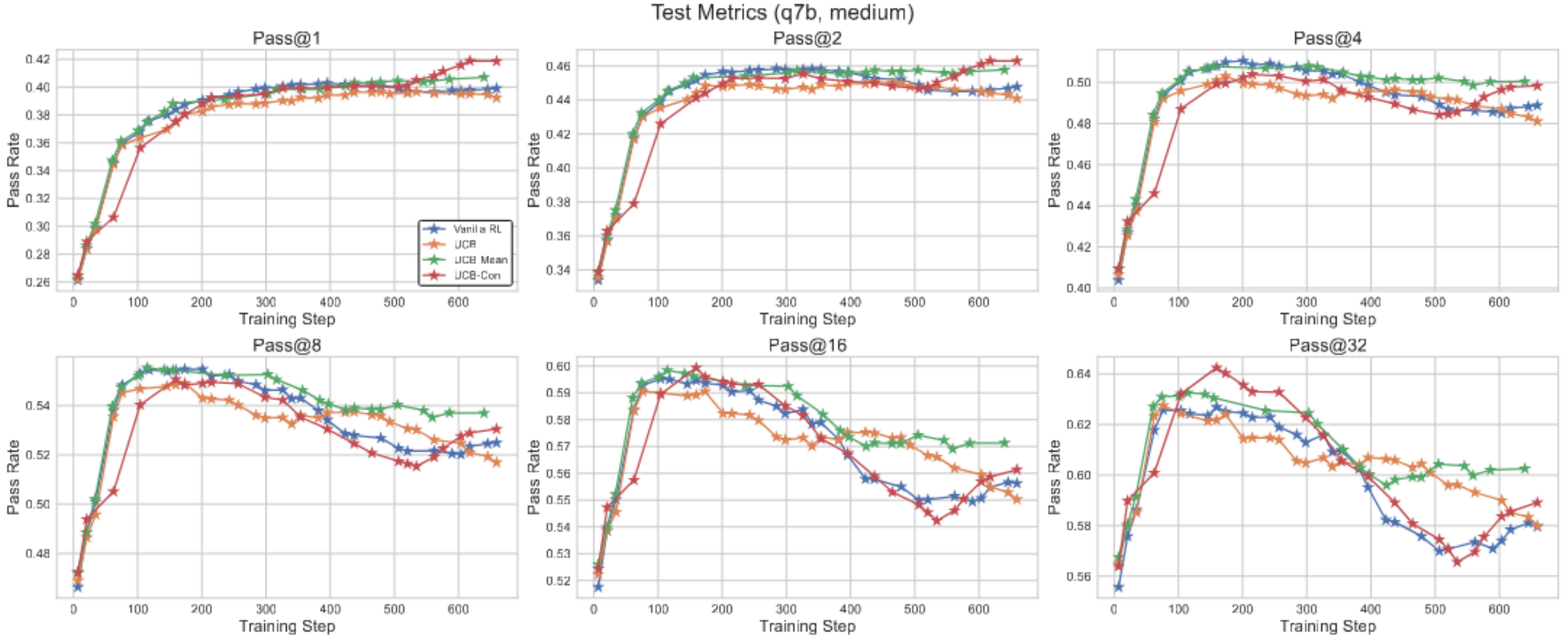
Fraction Solved

Unique Solutions (Unsolved)

Test Results

- Slower diversity degradation
- Better peak accuracy
- Mitigating over-optimization





What is Diversity

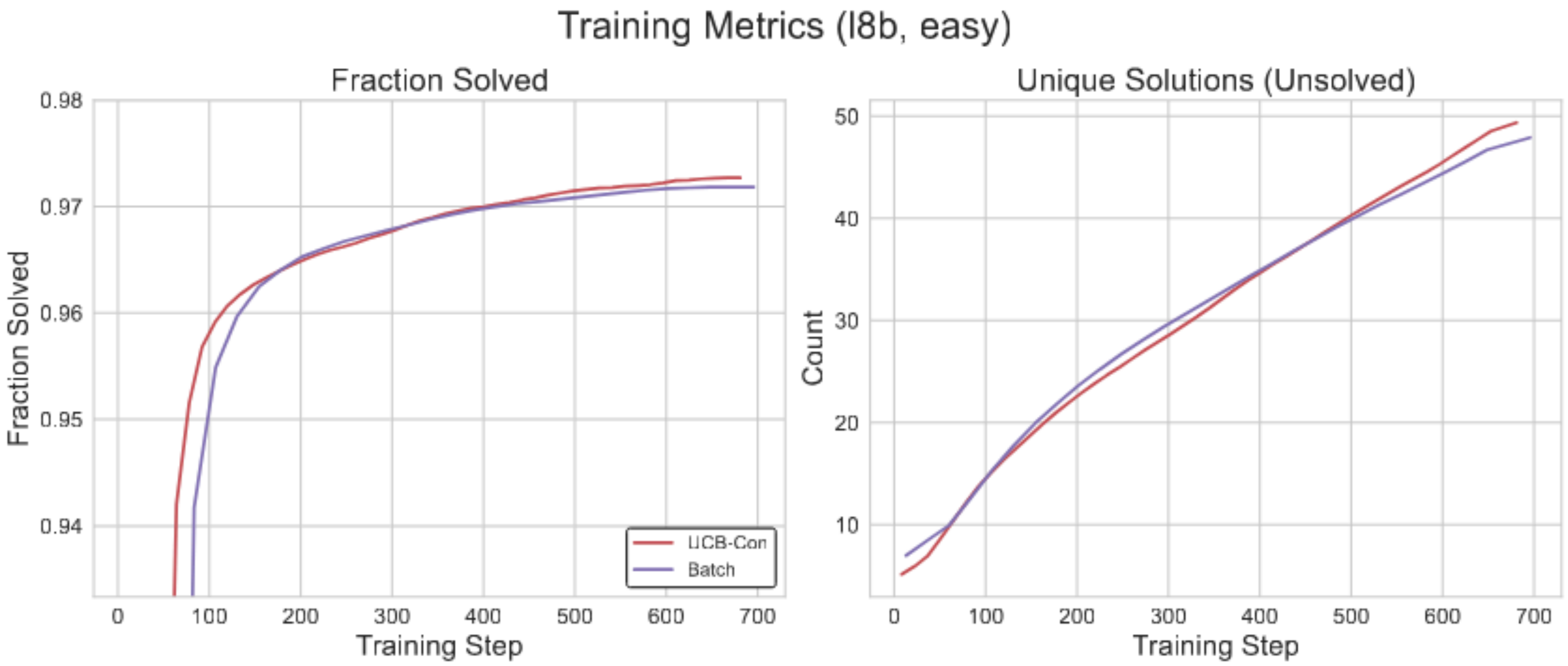
- Yes we explored during training time
- However, pass$@k$ is a test-time metric
- In fact, OBE theoretically can return a deterministic policy
- What if we just want a policy with diverse output?

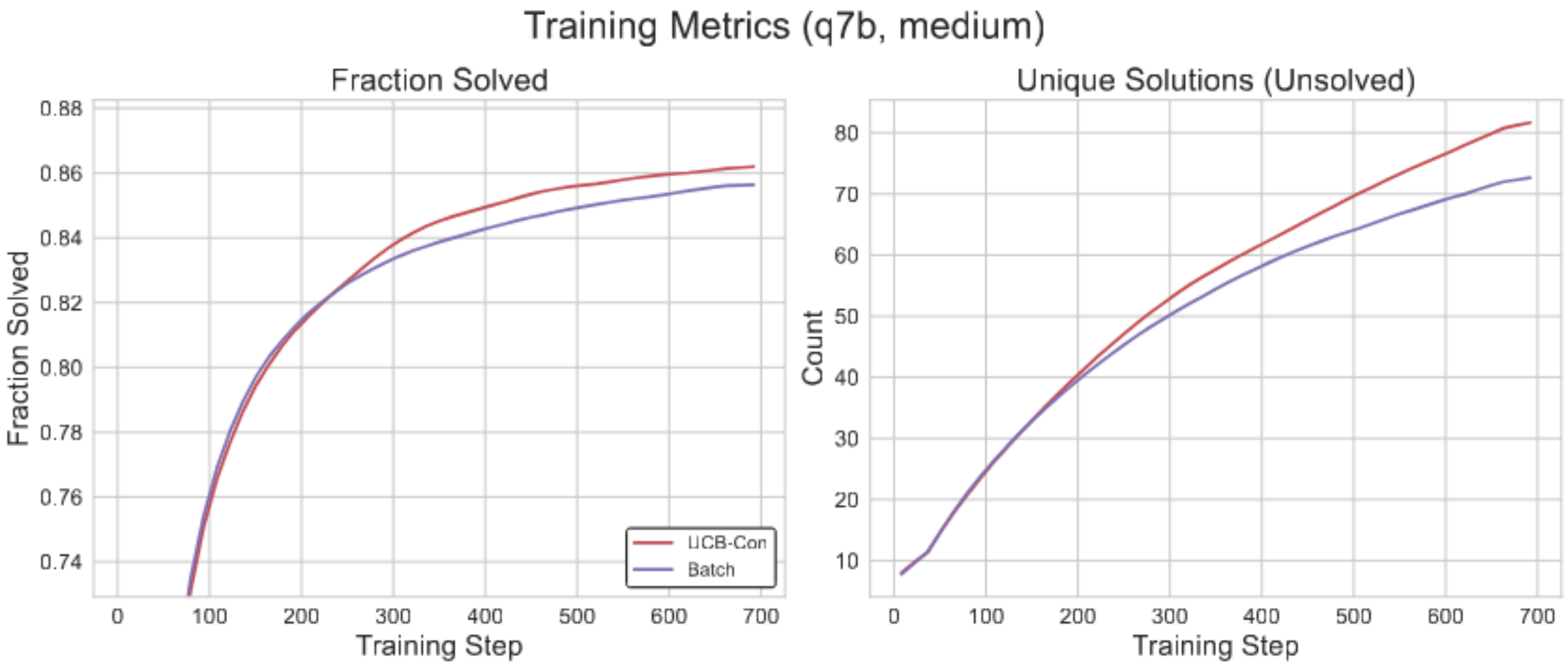
Batch Diversity

- Batch bonus:

$$\text{bonus}_{\text{batch}}(x, \{a_i\}_{\kappa=1}^{k})_i = -\frac{1}{k}\sum_{\kappa' \neq i}^{k} \mathbb{1}\{a_i = a_{\kappa'}\}$$

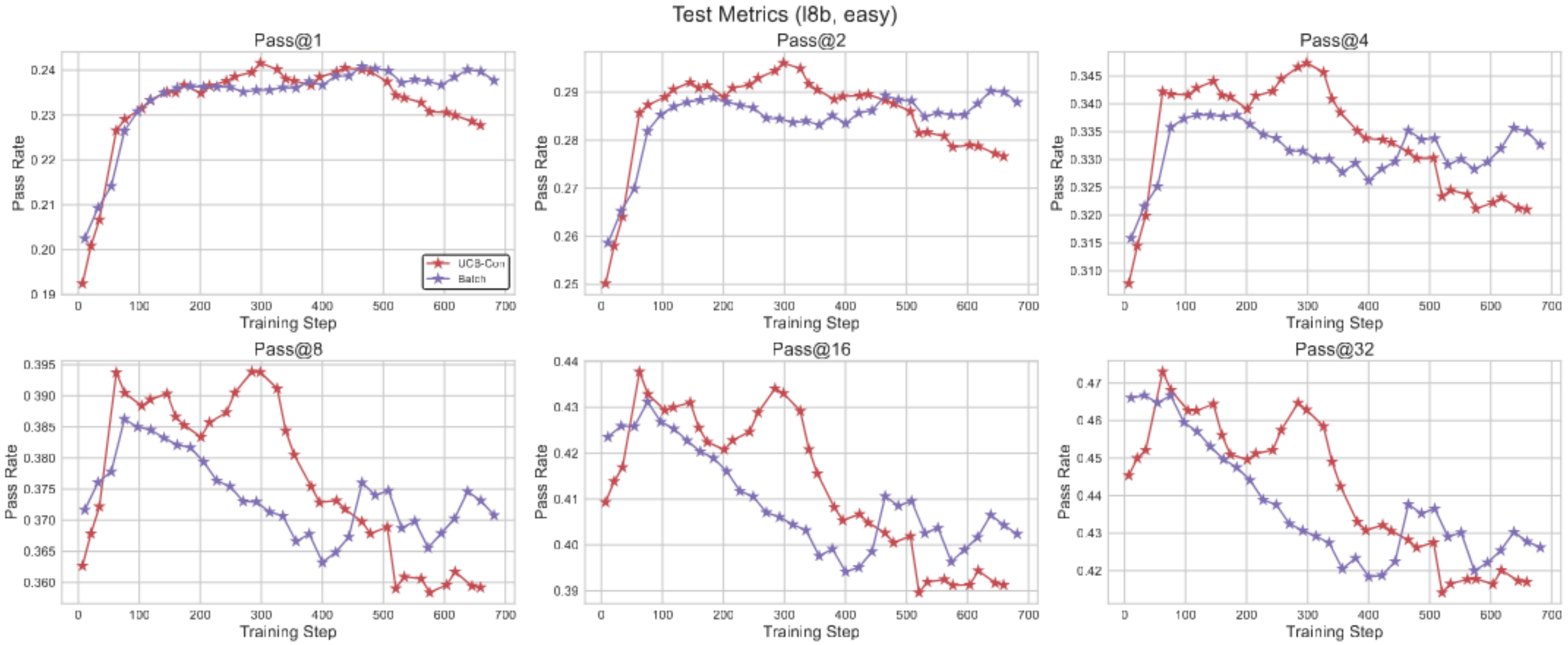
- Note that this is negative signal.



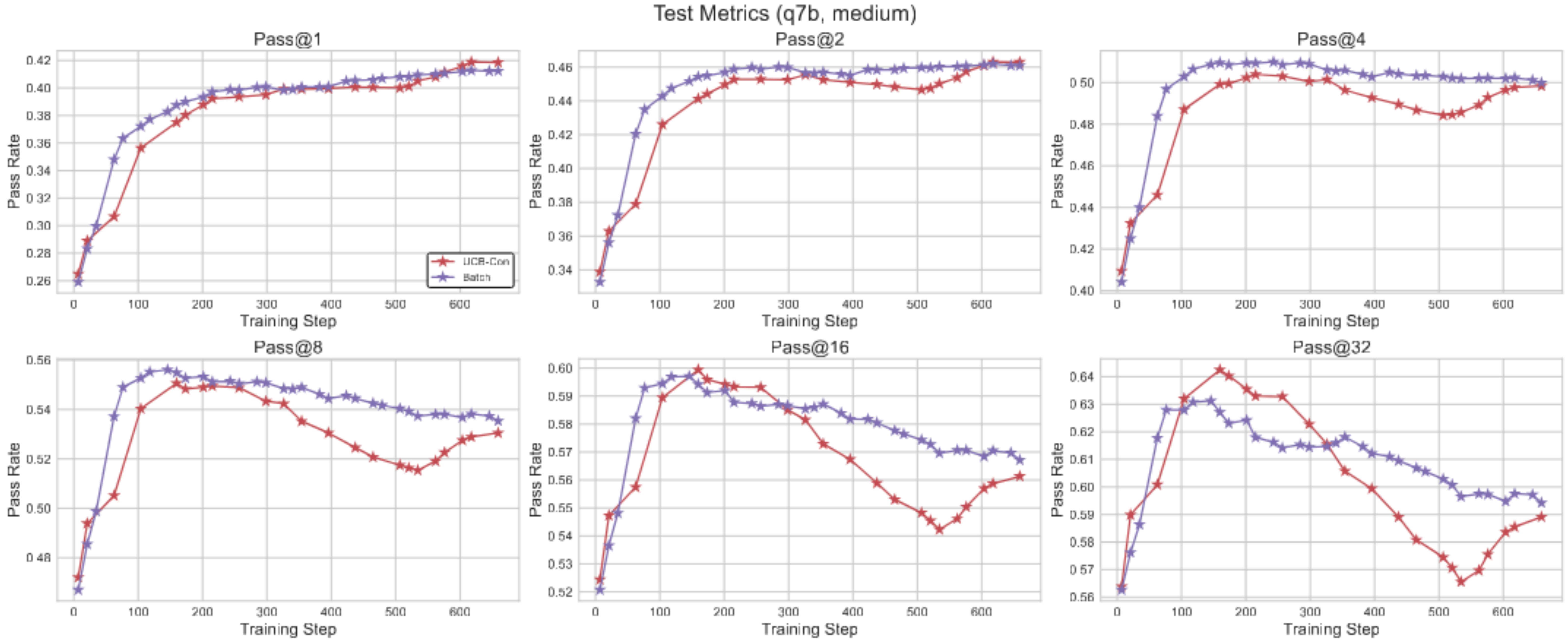


Test results

- Better diversity at the end of training
- Worse peak accuracy

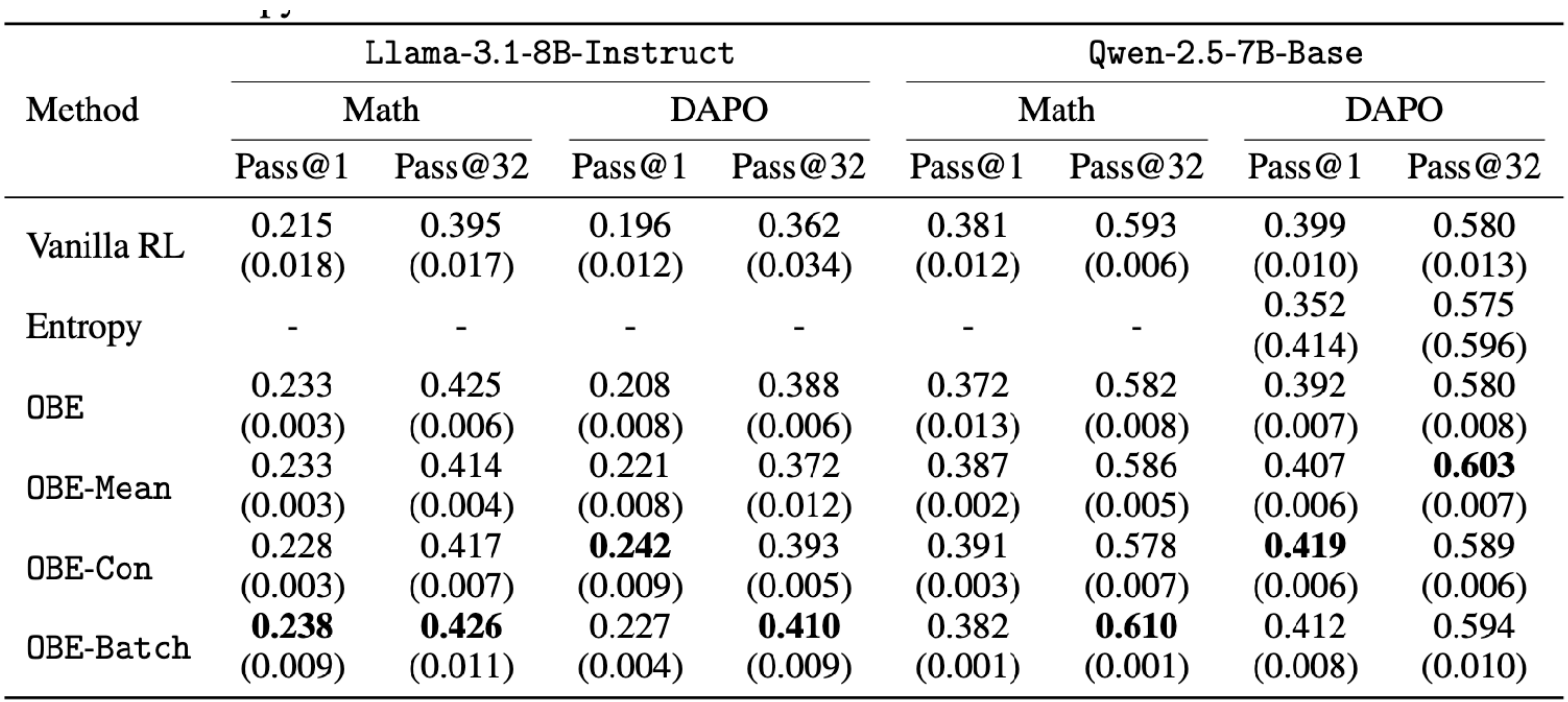


Test Metrics (l8b, easy)



Test Metrics (q7b, medium)

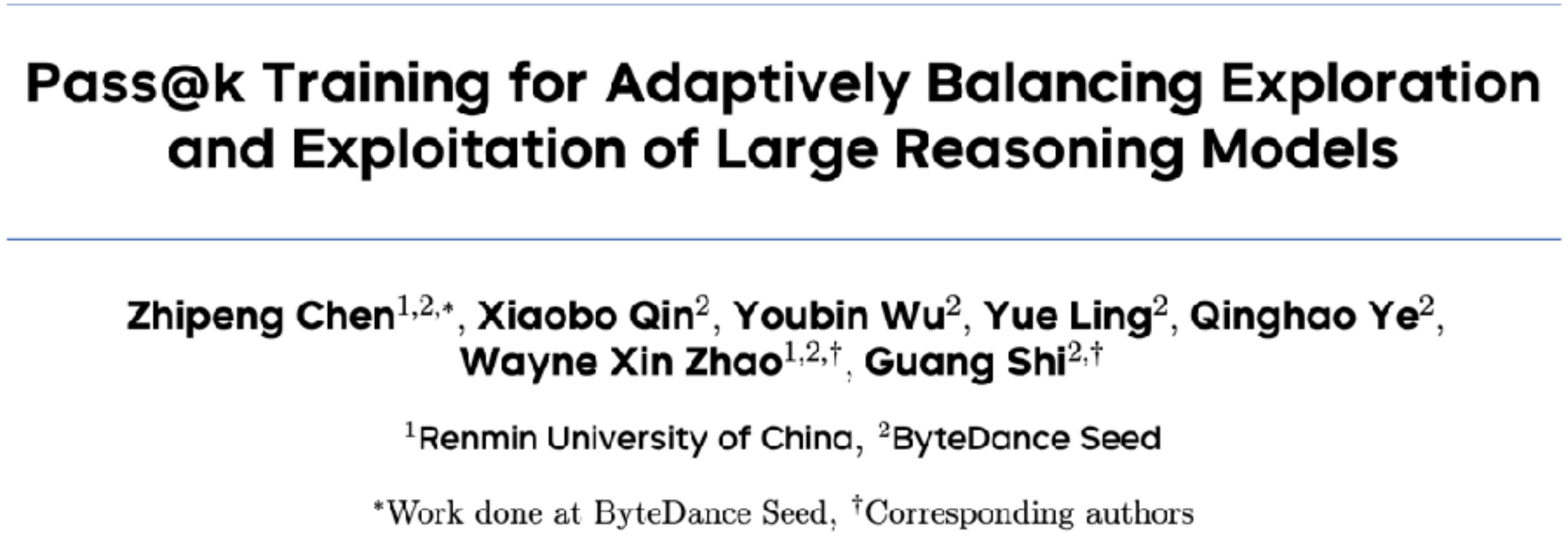
Quantitive Results



Method	Llama-3.1-8B-Instruct				Qwen-2.5-7B-Base			
	Math		DAPO		Math		DAPO	
	Pass@1	Pass@32	Pass@1	Pass@32	Pass@1	Pass@32	Pass@1	Pass@32
Vanilla RL	0.215 (0.018)	0.395 (0.017)	0.196 (0.012)	0.362 (0.034)	0.381 (0.012)	0.593 (0.006)	0.399 (0.010)	0.580 (0.013)
Entropy	-	-	-	-	-	-	0.352 (0.414)	0.575 (0.596)
OBE	0.233 (0.003)	0.425 (0.006)	0.208 (0.008)	0.388 (0.006)	0.372 (0.013)	0.582 (0.008)	0.392 (0.007)	0.580 (0.008)
OBE-Mean	0.233 (0.003)	0.414 (0.004)	0.221 (0.008)	0.372 (0.012)	0.387 (0.002)	0.586 (0.005)	0.407 (0.006)	**0.603** (0.007)
OBE-Con	0.228 (0.003)	0.417 (0.007)	**0.242** (0.009)	0.393 (0.005)	0.391 (0.003)	0.578 (0.007)	**0.419** (0.006)	0.589 (0.006)
OBE-Batch	**0.238** (0.009)	**0.426** (0.011)	0.227 (0.004)	**0.410** (0.009)	0.382 (0.001)	**0.610** (0.001)	0.412 (0.008)	0.594 (0.010)

What’s next?

- Multi-turn exploration
- Beyond verifiable rewards
- Fundamental paradigm shift
    - More steerable models

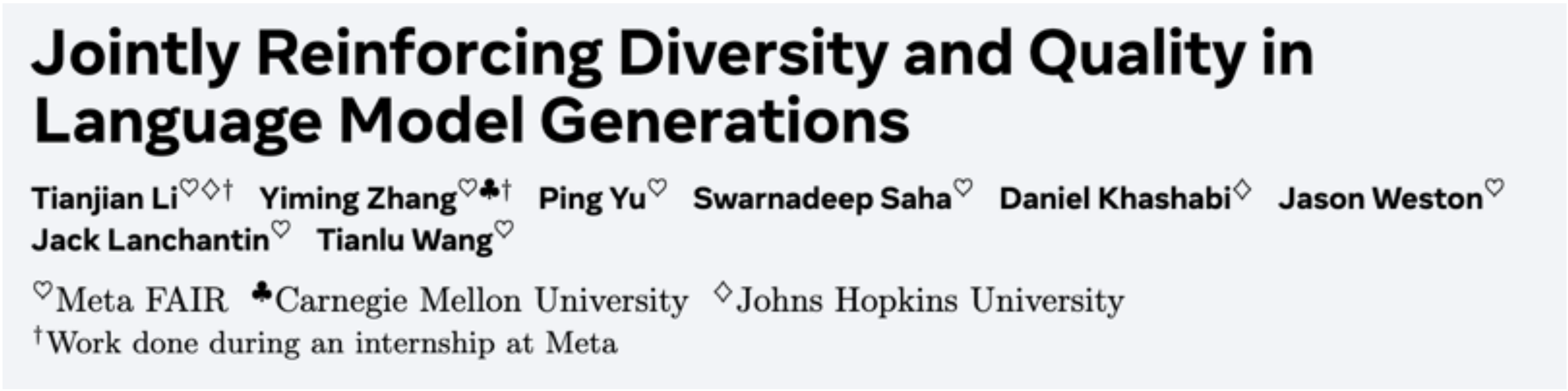


Pass@k Training for Adaptively Balancing Exploration and Exploitation of Large Reasoning Models

Zhipeng Chen[1,2,*], Xiaobo Qin[2], Youbin Wu[2], Yue Ling[2], Qinghao Ye[2], Wayne Xin Zhao[1,2,†], Guang Shi[2,†]

[1]Renmin University of China, [2]ByteDance Seed

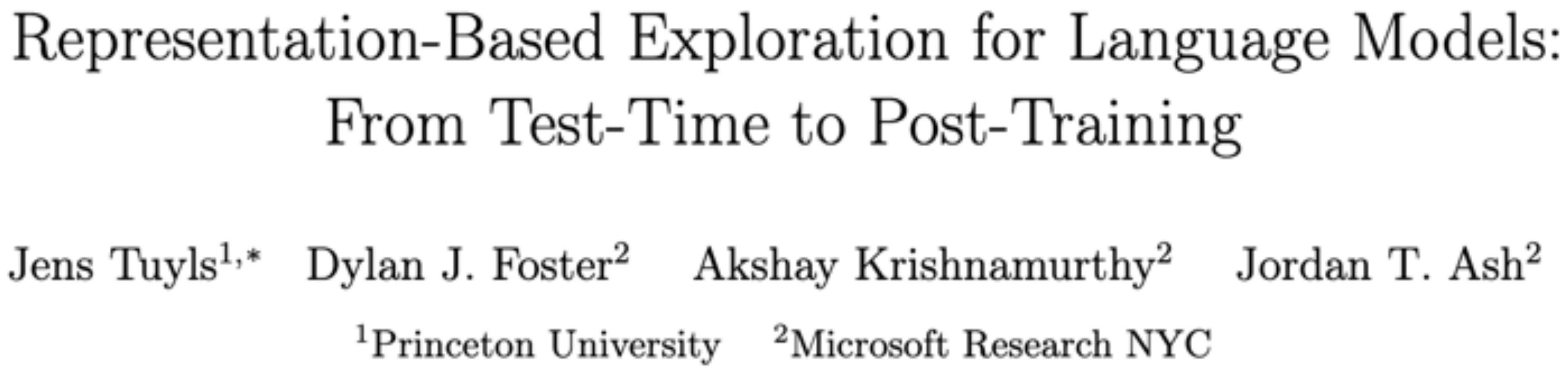
[*]Work done at ByteDance Seed, [†]Corresponding authors



Jointly Reinforcing Diversity and Quality in Language Model Generations

Tianjian Li[♡◇†] Yiming Zhang[♡♣†] Ping Yu[♡] Swarnadeep Saha[♡] Daniel Khashabi[◇] Jason Weston[♡] Jack Lanchantin[♡] Tianlu Wang[♡]

[♡]Meta FAIR [♣]Carnegie Mellon University [◇]Johns Hopkins University
[†]Work done during an internship at Meta



Representation-Based Exploration for Language Models: From Test-Time to Post-Training

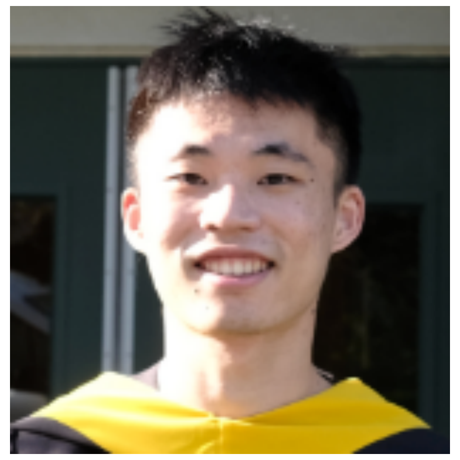
Jens Tuyls[1,*] Dylan J. Foster[2] Akshay Krishnamurthy[2] Jordan T. Ash[2]

[1]Princeton University [2]Microsoft Research NYC

1. How do we get feedback when the reward is unreliable?
2. How do we get additional signal when there is a reliable reward?
3. How do we efficiently query feedback (with help from synthetic feedback) when it is expensive?

Accelerating Unbiased LLM Evaluation via Synthetic Feedback

ICML 2025



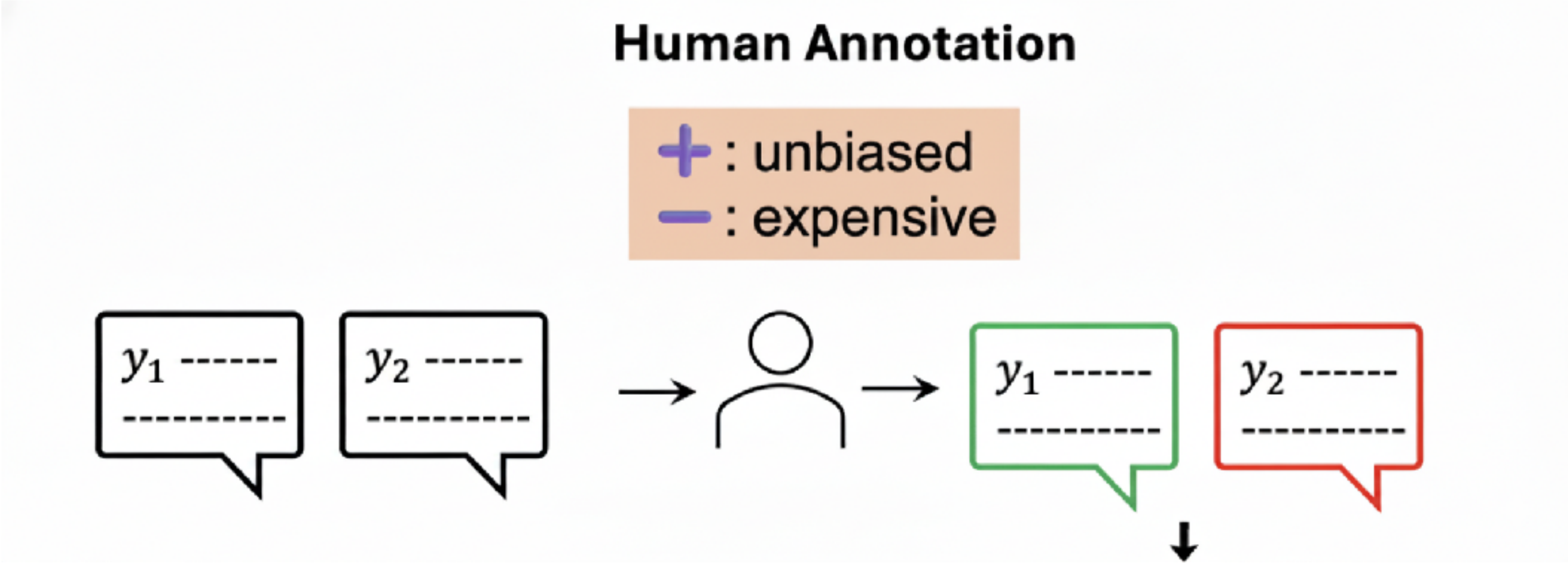
Zhaoyi Zhou, Yuda Song, Andrea Zenatte

How to evaluate my model?

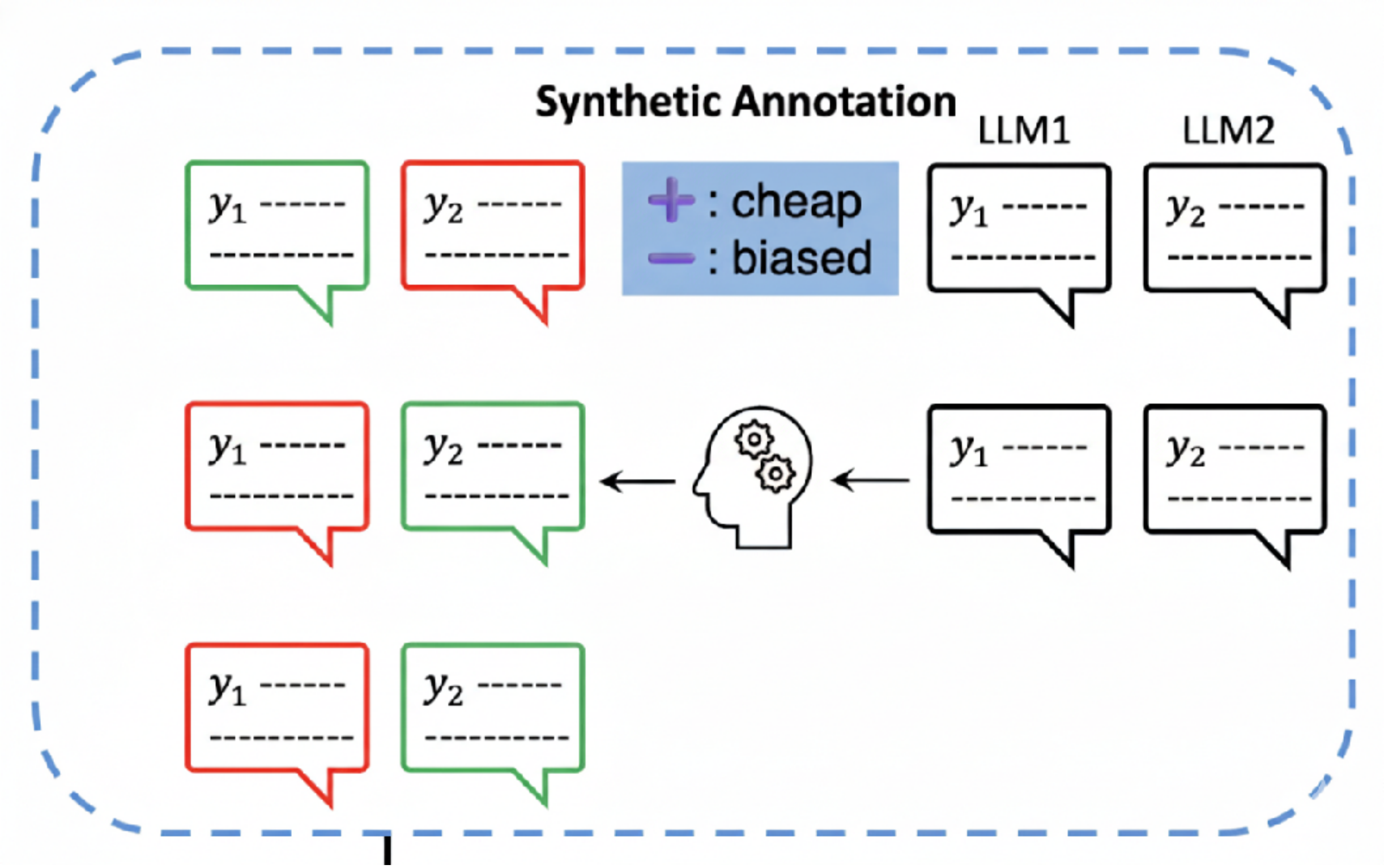
- LLM Arena
- Designing 100 different benchmarks

Querying Human Feedback is Expensive

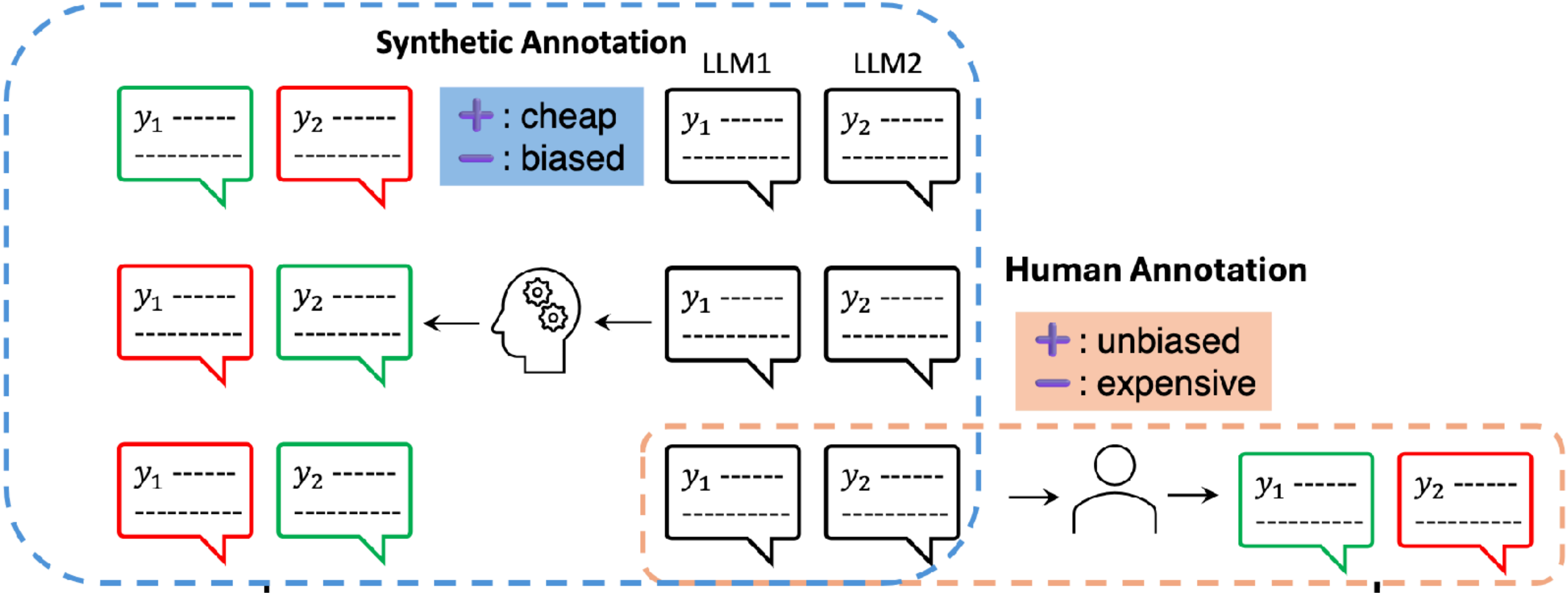
- Unless you are OAI, you need large community efforts (e.g. ChatBotArena)



Synthetic Feedback is Cheap



How about…


Synthetic Annotation
LLM1
LLM2
y_1 ------
y_2 ------
+ : cheap
− : biased
Human Annotation
+ : unbiased
− : expensive

Control Variates

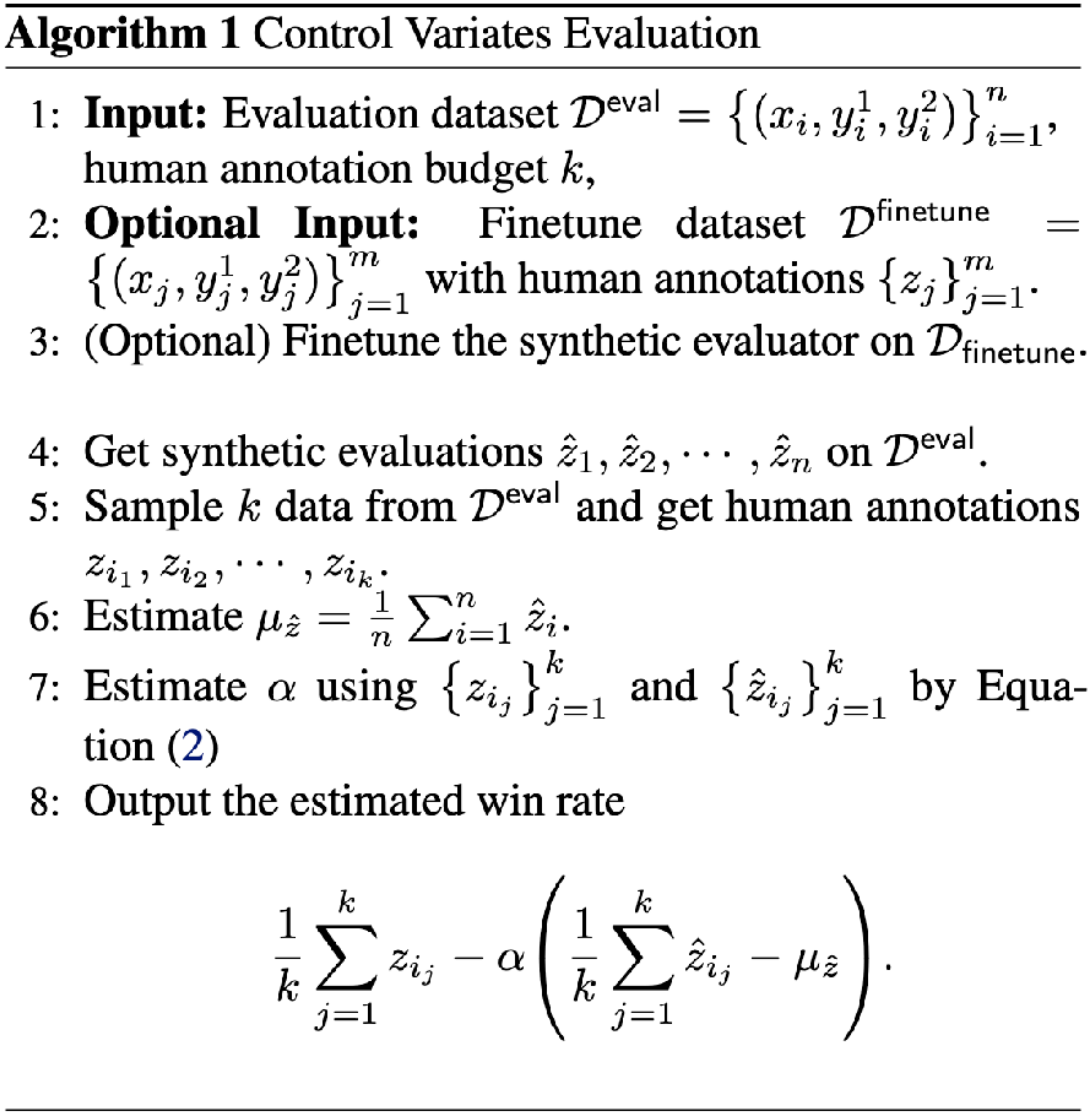
- Random variable $X, \mathbb{E}[X] = \mu_X$.
- Another random variable $Y, \mathbb{E}[X] = \mu_Y$.
- $X^\alpha = X - \alpha(Y - \mu_Y)$.
- $\alpha* = \mathrm{Cov}(X, Y)/\mathrm{Var}(Y)$.
- $\mathrm{Var}(X^{\alpha*}) = (1 - \rho^2)\mathrm{Var}(X)$.

Algorithm

- Unbiased
- Variance Reduction

Saving by $1 - \rho^2$

$$\rho^2 = \left(\mathrm{Corr}_{x,y^1,y^2}[z(y^1 \succ y^2), \hat{z}(y^1 \succ y^2)]\right)^2 .$$



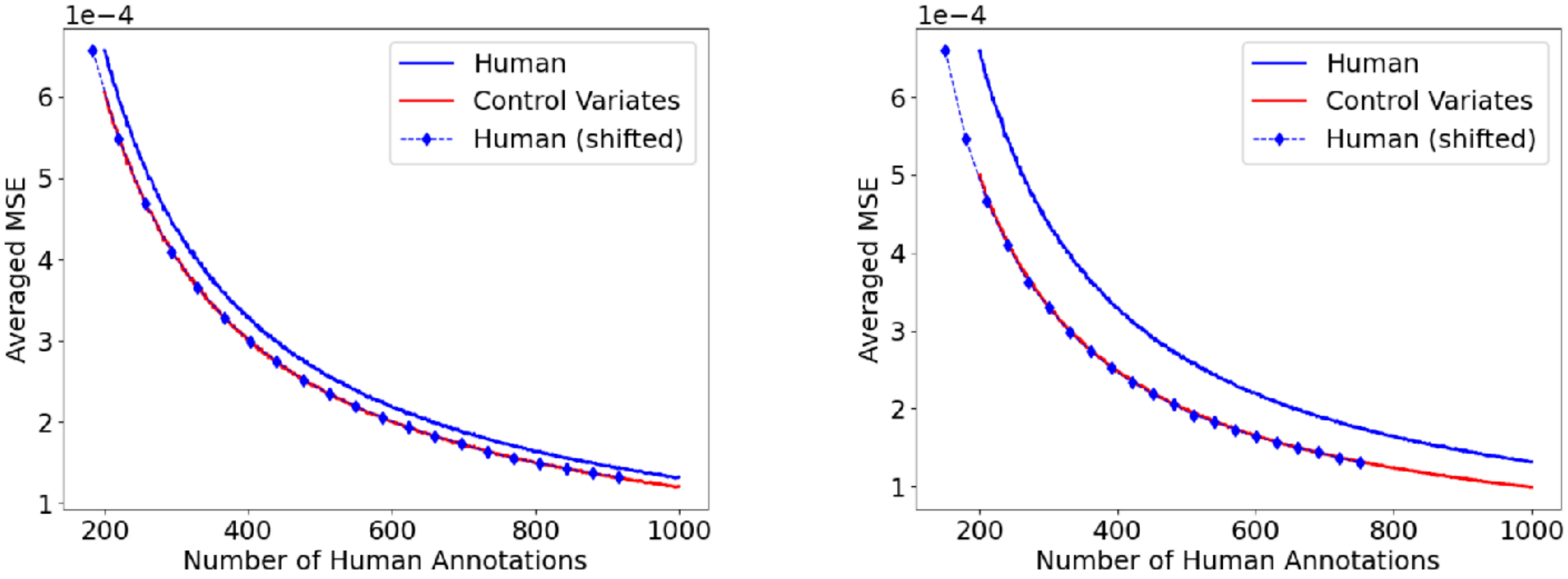
Algorithm 1 Control Variates Evaluation

```
Input: Evaluation dataset $\mathcal{D}^{\text{eval}} = \{(x_i, y_i^1, y_i^2)\}_{i=1}^n$,
    human annotation budget $k$,
Optional Input: Finetune dataset $\mathcal{D}^{\text{finetune}} =
    \{(x_j, y_j^1, y_j^2)\}_{j=1}^m$ with human annotations $\{z_j\}_{j=1}^m$.
(Optional) Finetune the synthetic evaluator on $\mathcal{D}_{\text{finetune}}$.

Get synthetic evaluations $\hat{z}_1, \hat{z}_2, \cdots, \hat{z}_n$ on $\mathcal{D}^{\text{eval}}$.
Sample $k$ data from $\mathcal{D}^{\text{eval}}$ and get human annotations
    $z_{i_1}, z_{i_2}, \cdots, z_{i_k}$.
Estimate $\mu_{\hat{z}} = \frac{1}{n}\sum_{i=1}^n \hat{z}_i$.
Estimate $\alpha$ using $\{z_{i_j}\}_{j=1}^k$ and $\{\hat{z}_{i_j}\}_{j=1}^k$ by Equation (2)
Output the estimated win rate

    $$\frac{1}{k}\sum_{j=1}^k z_{i_j} - \alpha\left(\frac{1}{k}\sum_{j=1}^k \hat{z}_{i_j} - \mu_{\hat{z}}\right).$$
```

Results


1e−4
6
5
4
3
2
1
Averaged MSE
200
400
600
800
1000
Number of Human Annotations
Human
Control Variates
Human (shifted)
1e−4
6
5
4
3
2
1
Averaged MSE
200
400
600
800
1000
Number of Human Annotations
Human
Control Variates
Human (shifted)